

---

# AWS Elastic Beanstalk

## Manuel du développeur



## AWS Elastic Beanstalk: Manuel du développeur

Copyright © Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Les marques et l'habillage commerciaux d'Amazon ne peuvent pas être utilisés en connexion avec un produit ou un service qui n'est pas celui d'Amazon, d'une manière susceptible de causer de la confusion chez les clients ou d'une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d'Amazon sont la propriété de leurs propriétaires respectifs, qui peuvent ou non être affiliés, connectés ou sponsorisés par Amazon.

## Table of Contents

Qu'est-ce que AWS Elastic Beanstalk ? .....	1
Tarification .....	1
Étapes suivantes .....	2
Mise en route .....	3
Configuration : Création d'un compte AWS .....	3
Étape 1 : Création .....	3
Création d'une application et d'un environnement .....	3
Ressources AWS créées pour l'exemple d'application .....	5
Étape 2 : Exploration .....	5
Étape 3 : Déploiement d'une nouvelle version .....	7
Étape 4 : Configuration .....	8
Réalisation d'une modification de la configuration .....	9
Vérification de la modification de la configuration .....	9
Étape 5 : Nettoyage .....	10
Étapes suivantes .....	11
Concepts .....	14
Application .....	14
Version de l'application .....	14
Environnement .....	14
Niveau de l'environnement .....	14
Configuration de l'environnement .....	14
Configuration enregistrée .....	15
Plateforme .....	15
Environnements de serveur web .....	15
Environnements de travail .....	16
Considérations relatives à la conception .....	17
Evolutivité .....	17
Sécurité .....	18
Stockage permanent .....	18
Tolérance aux pannes .....	19
Diffusion de contenu .....	19
Correctifs et mises à jour de logiciels .....	19
Connectivité .....	20
Autorisations .....	21
Rôle de service .....	21
Profil d'instance .....	22
Stratégie utilisateur .....	24
Plates-formes .....	25
Glossaire des plateformes .....	25
Modèle de responsabilité partagée .....	27
Stratégie de prise en charge de la plateforme .....	28
Calendrier de mise hors service des branches de plateforme .....	29
Branches de plate-forme mises hors service .....	30
Plateformes prises en charge .....	31
Versions de plateforme prises en charge .....	32
Plateformes Linux .....	32
Versions de plateforme Linux .....	33
Liste des plateformes Linux Elastic Beanstalk .....	33
Extension des plateformes Linux .....	34
Utilisation de Docker .....	46
Famille de plateformes Docker .....	47
La plateforme Docker .....	48
Docker multiconteneurs (AMI Amazon Linux) .....	64
Conteneurs préconfigurés .....	82

Configuration de l'environnement .....	85
Exécution de conteneurs localement .....	93
Utilisation de Go .....	98
Mise en route .....	98
Environnement de développement .....	100
Plateforme Go .....	101
Didacticiel de Go .....	106
Travail avec Java .....	110
Mise en route .....	111
Environnement de développement .....	116
Plateforme Tomcat .....	118
Plateforme Java SE .....	129
Ajout d'une base de données .....	136
Toolkit for Eclipse .....	142
Resources .....	157
Utilisation de .NET Core sous Linux .....	157
Mise en route .....	158
Environnement de développement .....	160
.NET Core sur la plateforme Linux .....	161
Didacticiel - .NET core sous Linux .....	165
. AWS Toolkit for Visual Studio .....	171
Migration de Windows vers Linux .....	191
Utilisation de .NET .....	192
Mise en route .....	192
Environnement de développement .....	195
Plateforme .NET .....	196
Didacticiel - ASP.NET MVC5 .....	206
Didacticiel - .NET Core .....	212
Ajout d'une base de données .....	221
. AWS Toolkit for Visual Studio .....	224
Migration de l'application sur site .....	250
Resources .....	251
Utilisation de Node.js .....	251
Mise en route .....	252
Environnement de développement .....	253
Plateforme Node.js .....	255
Didacticiel - Express .....	265
Didacticiel - Express avec mise en cluster .....	269
Tutoriel - Node.js avec DynamoDB .....	278
Ajout d'une base de données .....	287
Resources .....	290
Utilisation de PHP .....	290
Environnement de développement .....	291
Plateforme PHP .....	293
Didacticiel - Laravel .....	298
Didacticiel - CakePHP .....	305
Didacticiel - Symfony .....	311
Didacticiel - Production haute disponibilité .....	316
Didacticiel - WordPress haute disponibilité .....	325
Didacticiel - Drupal haute disponibilité .....	339
Ajout d'une base de données .....	353
Travail avec Python .....	356
Environnement de développement .....	357
Plateforme Python .....	359
Didacticiel - Flask .....	365
Didacticiel - Django .....	371
Ajout d'une base de données .....	382

Ressources .....	384
Utilisation de Ruby .....	384
Environnement de développement .....	384
La plateforme Ruby .....	387
Didacticiel - Rails .....	391
Didacticiel - Sinatra .....	397
Ajout d'une base de données .....	401
Didacticiels et exemples .....	404
Gestion d'applications .....	406
Console de gestion d'application .....	408
Gestion des versions d'application .....	409
Cycle de vie des versions .....	411
Balisage des versions d'application .....	413
Création d'une solution groupée source .....	415
Création d'une solution groupée source à partir de la ligne de commande .....	416
Création d'une solution groupée source avec Git .....	416
Compression de fichiers dans le Finder de Mac OS X ou l'Explorateur Windows .....	416
Création d'une solution groupée source pour une application .NET .....	421
Test de votre solution groupée source .....	422
Balisage des ressources .....	423
Ressources que vous pouvez baliser .....	424
Balisage des applications .....	424
Gestion des environnements .....	428
Console de gestion de l'environnement .....	429
Présentation de l'environnement .....	430
Actions dans l'environnement .....	432
Configuration .....	433
Logs .....	435
Health .....	435
Monitoring .....	436
Alarms .....	436
Mises à jour gérées .....	437
Events .....	437
Tags .....	438
Création d'environnements .....	439
Assistant de création d'un environnement .....	442
Clonage d'un environnement .....	461
Résiliation d'un environnement .....	464
Avec l'AWS CLI .....	465
Via l'API .....	466
URL Launch Now .....	469
API Compose Environments .....	474
Déploiements .....	476
Choix d'une stratégie de déploiement .....	477
Déploiement d'une nouvelle version de l'application .....	478
Redéploiement d'une version précédente .....	479
Autres méthodes de déploiement de votre application .....	479
Options de déploiement .....	479
Déploiements bleu/vert .....	486
Configuration changes .....	488
Mises à jour propagées .....	489
Mises à jour immuables .....	493
Mises à jour de plateforme .....	496
Méthode 1 – Mettre à jour la version de la plateforme de votre environnement .....	498
Méthode 2 – Effectuer un déploiement bleu/vert .....	500
Mises à jour gérées .....	501
Mise à niveau d'un environnement hérité .....	507

Mise à niveau vers Amazon Linux 2 .....	508
Annulation d'une mise à jour .....	516
Reconstruction d'un environnement .....	517
Reconstruction d'un environnement en cours d'exécution .....	517
Reconstruction d'un environnement suspendu .....	517
Types d'environnement .....	519
Environnement évolutif et équilibré en charge .....	519
Environnement à instance unique .....	520
Changement de type d'environnement .....	520
Environnements de travail .....	521
Démon SQS d'environnement de travail .....	524
Files d'attente de lettres mortes .....	525
Tâches périodiques .....	525
Utilisation d'Amazon CloudWatch pour la mise à l'échelle automatique dans les niveaux d'environnement de travail .....	526
Configuration des environnements de travail .....	527
Liens entre environnements .....	530
Configuration des environnements .....	532
Configuration via la console .....	533
Page de présentation de la configuration .....	534
Page de vérification des modifications .....	537
Instances Amazon EC2 .....	538
Configuration des instances Amazon EC2 de votre environnement .....	539
Espace de noms aws:autoscaling:launchconfiguration .....	544
IMDS .....	545
Groupe Auto Scaling .....	547
Prise en charge d'une instance Spot .....	548
Configuration du groupe Auto Scaling à l'aide de la console Elastic Beanstalk .....	550
Configuration du groupe Auto Scaling à l'aide de l'interface de ligne de commande EB .....	554
Options de configuration .....	555
Déclencheurs .....	555
Actions planifiées .....	558
Paramètres de vérification de l'état .....	561
ÉquilibrEUR de charge .....	562
ÉquilibrEUR de charge classique .....	564
Application Load Balancer .....	574
ÉquilibrEUR de charge Application Load Balancer partagé .....	591
ÉquilibrEUR de charge du réseau .....	607
Configuration des journaux d'accès .....	617
Base de données .....	617
Cycle de vie de base de données .....	618
Ajout d'une instance de base de données Amazon RDS à votre environnement à l'aide de la console .....	619
Connexion à la base de données .....	620
Configuration d'une instance de base de données RDS intégrée à l'aide de la console .....	621
Configuration d'une instance de base de données RDS intégrée à l'aide des fichiers de configuration .....	621
Découplage d'une instance de base de données RDS à l'aide de la console .....	622
Découplage d'une instance de base de données RDS à l'aide de fichiers de configuration .....	625
Sécurité .....	626
Configuration de la sécurité de votre environnement .....	626
Espaces de noms de configuration de la sécurité de l'environnement .....	628
Balisage des environnements .....	629
Ajout de balises lors de la création de l'environnement .....	629
Gestion des balises d'un environnement existant .....	630
Paramètres du logiciel .....	632
Configurer les paramètres spécifiques à la plateforme .....	633

Configuration des propriétés de l'environnement .....	634
Espaces de noms des paramètres de logiciel .....	636
Accès aux propriétés de l'environnement .....	637
Débogage .....	638
Affichage des journaux .....	641
Notifications .....	644
Configuration des notifications à l'aide de la console Elastic Beanstalk .....	645
Configuration des notifications à l'aide des options de configuration .....	646
Configuration des autorisations d'envoi de notifications .....	648
Amazon VPC .....	649
Configuration des paramètres VPC dans la console Elastic Beanstalk .....	649
Espace de noms aws:ec2:vpc .....	652
Migration d'EC2-Classic vers un VPC .....	653
Nom de domaine .....	656
Configuration d'environnements (niveau avancé) .....	658
Options de configuration .....	658
Precedence .....	659
Valeurs recommandées .....	660
Avant la création de l'environnement .....	661
Lors de la création .....	665
Après la création .....	671
Options générales .....	679
Options spécifiques à une plateforme .....	727
Options personnalisées .....	736
.Ebextensions .....	737
Paramètres d'option .....	738
Serveur Linux .....	740
Serveur Windows .....	753
Ressources personnalisées .....	759
Configurations enregistrées .....	779
Balisage de configurations enregistrées .....	783
.env.yaml .....	785
Image personnalisée .....	787
Création d'une AMI personnalisée .....	787
Nettoyage d'une AMI personnalisée .....	790
Fichiers statiques .....	790
Configurer les fichiers statiques à l'aide de la console .....	791
Configurer des fichiers statiques à l'aide des options de configuration .....	792
HTTPS .....	792
Créer un certificat .....	794
Chargement d'un certificat .....	796
Arrêtez la connexion au niveau de l'équilibrEUR de charge .....	797
Suspension sur l'instance .....	799
Chiffrement de bout en bout .....	824
TCP Passthrough .....	827
Stockage sécurisé des clés .....	828
Redirection HTTP vers HTTPS .....	829
Surveillance d'un environnement .....	830
Console de surveillance .....	830
Présentation .....	831
Surveillance des graphiques .....	831
Personnalisation de la console de surveillance .....	832
Création de rapports d'intégrité de base .....	834
Couleurs de l'intégrité .....	835
Vérifications de l'état Elastic Load Balancing .....	835
Vérifications de l'état d'un environnement à instance unique et d'un environnement de travail .....	836
Contrôles supplémentaires .....	836

Métriques Amazon CloudWatch .....	836
Surveillance et création de rapports d'intégrité améliorée .....	837
Agent de vérification de l'état Elastic Beanstalk .....	840
Facteurs de détermination de l'intégrité de l'environnement et de l'instance .....	840
Personnalisation d'une règle de vérification de l'état .....	842
Rôles d'intégrité améliorée .....	843
Autorisation de santé améliorée .....	843
Événements d'intégrité améliorée .....	844
Comportement de la création de rapports d'intégrité améliorée au cours des mises à jour, des déploiements et de la mise à l'échelle .....	845
Activation des rapports d'intégrité améliorée .....	845
Console de surveillance de l'état .....	849
Couleurs et états utilisés dans les rapports d'intégrité .....	854
Métriques des instances .....	856
Règles d'intégrité améliorée .....	859
CloudWatch .....	862
Utilisateurs de l'API .....	869
Format de journal d'intégrité améliorée .....	870
Notifications et dépannage .....	873
Gestion des alarmes .....	874
Afficher l'historique des modifications .....	877
Affichage des événements .....	879
Surveillance des instances .....	881
Affichage des journaux d'instance .....	884
Emplacement des journaux sur les instances Amazon EC2 .....	886
Emplacement des journaux dans Amazon S3 .....	886
Paramètres de rotation des journaux sous Linux .....	887
Extension de la configuration de tâche de journal par défaut .....	887
Diffusion de fichiers journaux vers les Amazon CloudWatch Logs .....	889
Intégration des services AWS .....	891
Présentation de l'architecture .....	891
CloudFront .....	892
CloudTrail .....	893
Informations Elastic Beanstalk dans CloudTrail .....	893
Présentation des entrées du fichier journal Elastic Beanstalk .....	894
CloudWatch .....	894
CloudWatch Logs .....	895
Conditions préalables pour la diffusion des journaux d'instance vers CloudWatch Logs .....	897
Méthode de configuration de CloudWatch Logs par Elastic Beanstalk .....	898
Diffusion de journaux d'instance vers CloudWatch Logs .....	901
Résolution des problèmes liés à l'intégration de CloudWatch Logs .....	903
Diffusion des informations d'intégrité de l'environnement .....	903
EventBridge .....	906
Surveiller une ressource Elastic Beanstalk avec EventBridge .....	907
Exemple de modèles d'événements Elastic Beanstalk .....	908
Exemple d'événement Elastic Beanstalk .....	910
Mappage des champs d'événement Elastic Beanstalk .....	911
AWS Config .....	913
Configuration d AWS Config .....	913
Configuration de AWS Config pour enregistrer les ressources Elastic Beanstalk .....	913
Affichage des détails de configuration Elastic Beanstalk dans la console AWS Config .....	914
Évaluation des ressources Elastic Beanstalk à l'aide des règles AWS Config .....	917
DynamoDB .....	917
ElastiCache .....	918
Amazon EFS .....	918
Fichiers de configuration .....	919
Systèmes de fichiers chiffrés .....	919

Exemples d'applications .....	919
Nettoyage de systèmes de fichiers .....	920
IAM .....	920
Profils d'instance .....	921
Rôles de service .....	926
Utilisation des rôles liés à un service .....	935
Stratégies utilisateur .....	946
Format ARN .....	951
Ressources et conditions .....	952
Contrôle d'accès basé sur les balises .....	978
Exemples de stratégies gérées .....	981
Exemple de stratégies spécifiques aux ressources .....	984
Amazon RDS .....	991
Amazon RDS dans un VPC par défaut .....	992
Amazon RDS dans EC2-Classic .....	997
Chaîne de connexion dans Amazon S3 .....	1001
Nettoyage d'une instance Amazon RDS externe .....	1003
Amazon S3 .....	1003
Contenu du compartiment Elastic Beanstalk Amazon S3 .....	1003
Suppression d'objets dans le compartiment Elastic Beanstalk Amazon S3 .....	1004
Suppression du compartiment Elastic Beanstalk Amazon S3 .....	1005
Amazon VPC .....	1006
VPC public .....	1007
VPC public/privé .....	1008
VPC privé .....	1008
Hôtes bastions .....	1010
Amazon RDS .....	1015
Points de terminaison d'un VPC .....	1021
Configuration de votre machine de développement .....	1024
Création d'un dossier de projet .....	1024
Configuration du contrôle de code source .....	1024
Configuration d'un référentiel distant .....	1025
Installation de l'interface de ligne de commande EB .....	1025
Installation de l'AWS CLI .....	1026
Interface de ligne de commande EB .....	1027
Installation de l'interface de ligne de commande EB .....	1028
Installation de l'interface de ligne de commande EB à l'aide de scripts .....	1028
Installation manuelle .....	1028
Configuration de l'interface de ligne de commande EB .....	1036
Utilisation d'un fichier .ebignore pour ignorer des fichiers .....	1038
Utilisation de profils nommés .....	1038
Déploiement d'un artefact à la place du dossier de projet .....	1039
Configuration des paramètres et des priorités .....	1039
Métadonnées de l'instance .....	1039
Principes de base de l'interface de ligne de commande EB .....	1040
Eb create .....	1040
Eb status .....	1041
Eb health .....	1041
Eb events .....	1042
Eb logs .....	1042
Eb open .....	1042
Eb deploy .....	1043
Eb config .....	1043
Eb terminate .....	1044
CodeBuild .....	1044
Création d'une application .....	1045
Génération et déploiement du code de votre application .....	1045

Utilisation de l'interface de ligne de commande EB avec Git .....	1046
Associer des environnements Elastic Beanstalk à des branches Git .....	1047
Déploiement des modifications .....	1047
Utilisation des sous-modules Git .....	1047
Affectation de balises Git à votre version de l'application .....	1048
CodeCommit .....	1048
Prérequis .....	1049
Création d'un référentiel CodeCommit avec l'interface de ligne de commande EB .....	1049
Déploiement à partir de votre référentiel CodeCommit .....	1050
Configuration de branches et environnements supplémentaires .....	1051
Utilisation d'un référentiel CodeCommit existant .....	1052
Surveillance de l'intégrité .....	1053
Lecture du résultat .....	1055
Vue d'intégrité interactive .....	1057
Options d'affichage d'intégrité interactif .....	1058
Composition des environnements .....	1058
Dépannage .....	1060
Résolution de problèmes de déploiement .....	1060
Commandes de l'interface de ligne de commande (CLI) EB .....	1062
eb abort .....	1063
eb appversion .....	1064
eb clone .....	1068
eb codesource .....	1070
eb config .....	1071
eb console .....	1078
eb create .....	1078
eb deploy .....	1089
eb events .....	1090
eb health .....	1092
eb init .....	1093
eb labs .....	1096
eb list .....	1096
eb local .....	1097
eb logs .....	1100
eb open .....	1102
eb platform .....	1103
eb printenv .....	1110
eb restore .....	1111
eb scale .....	1112
eb setenv .....	1112
eb ssh .....	1113
eb status .....	1115
eb swap .....	1117
eb tags .....	1118
eb terminate .....	1120
eb upgrade .....	1122
eb use .....	1123
Options courantes .....	1123
Interface de ligne de commande EB 2.6 (retirée) .....	1124
Différences par rapport à la version 3 de l'interface de ligne de commande EB .....	1124
Migration vers l'interface de ligne de commande Elastic Beanstalk 3 et CodeCommit .....	1125
Interface de ligne de commande de l'API EB (mis hors service) .....	1125
Conversion des scripts d'interface de ligne de commande de l'API Elastic Beanstalk .....	1125
Sécurité .....	1128
Protection des données .....	1128
Chiffrement des données .....	1129
Trafic inter-réseaux .....	1130

Identity and Access Management .....	1130
Stratégies gérées par AWS .....	1131
Journalisation et surveillance .....	1134
Création de rapports d'intégrité améliorée .....	1135
Journaux des instances Amazon EC2 .....	1135
Notifications de l'environnement .....	1135
Alarmes Amazon CloudWatch .....	1135
Journaux AWS CloudTrail .....	1135
Débogage AWS X-Ray .....	1135
Validation de la conformité .....	1136
Résilience .....	1136
Sécurité de l'infrastructure .....	1137
Modèle de responsabilité partagée .....	1137
Bonnes pratiques de sécurité .....	1137
Bonnes pratiques de sécurité préventive .....	1137
Bonnes pratiques de sécurité de détection .....	1138
Dépannage .....	1140
Connectivity .....	1140
Création de l'environnement .....	1141
Déploiements .....	1142
Santé .....	1142
Configuration .....	1142
Docker .....	1143
FAQ .....	1144
Ressources .....	1145
Exemples d'applications .....	1145
Historique de la plateforme .....	1147
Plateformes personnalisées .....	1147
Création d'une plateforme personnalisée .....	1148
Utilisation d'un exemple de plateforme personnalisée .....	1148
Contenu de l'archive de définition de plateforme .....	1153
Hooks de plateforme personnalisée .....	1153
Nettoyage des instances Packer .....	1154
Format platform.yaml .....	1155
Balisage des versions de plateforme personnalisée .....	1157

# Qu'est-ce que AWS Elastic Beanstalk ?

Amazon Web Services (AWS) comprend plus d'une centaine de services, chacun d'entre eux étant spécialisé dans un ensemble de fonctionnalités spécifique. Même si les différents services offrent une certaine flexibilité pour vous permettre de gérer votre infrastructure AWS, il est parfois difficile d'identifier les services à utiliser et de comprendre comment les allouer.

Avec Elastic Beanstalk, vous pouvez déployer et gérer rapidement les applications sur le cloud AWS, sans vous préoccuper de l'infrastructure qui les exécute. Elastic Beanstalk réduit la complexité inhérente à la gestion sans pour autant sacrifier le choix ou le niveau de contrôle. Vous téléchargez simplement votre application, et Elastic Beanstalk gère automatiquement les détails du dimensionnement des capacités, de la répartition de la charge, de la mise à l'échelle et de la surveillance de l'état de l'application.

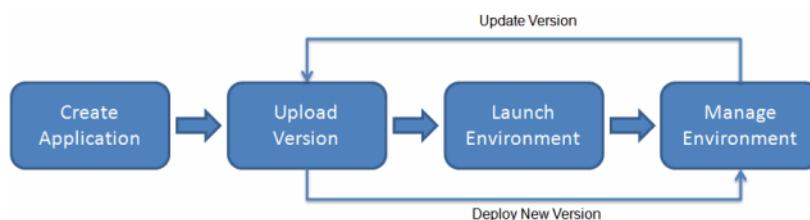
Elastic Beanstalk prend en charge les applications développées dans Go, Java, .NET, Node.js, PHP, Python et Ruby. Lorsque vous déployez votre application, Elastic Beanstalk crée la version de plateforme prise en charge qui a été sélectionnée et alloue une ou plusieurs ressources AWS, telles que des instances Amazon EC2, pour l'exécution de votre application.

Vous pouvez interagir avec Elastic Beanstalk à l'aide de la console Elastic Beanstalk, de l'AWS Command Line Interface (AWS CLI) ou d'eb, interface de ligne de commande de haut niveau conçue spécifiquement pour Elastic Beanstalk.

Pour en savoir plus sur le déploiement d'un exemple d'application web avec Elastic Beanstalk, veuillez consulter [Mise en route avec Démarrer avec AWS : déploiement d'une application web](#).

Vous pouvez également effectuer directement la plupart de vos tâches de déploiement, telles que modifier la taille de votre flotte d'instances Amazon EC2 ou surveiller votre application, depuis l'interface web Elastic Beanstalk (console).

Pour utiliser Elastic Beanstalk, vous devez créer une application, charger une version d'application sous la forme d'un bundle de fichiers source d'application (par exemple, un fichier Java .war) sur Elastic Beanstalk, puis fournir des informations sur l'application. Elastic Beanstalk lance automatiquement un environnement, et crée et configure les ressources AWS nécessaires pour exécuter votre code. Une fois le lancement de votre environnement effectué, vous pouvez le gérer et déployer de nouvelles versions d'application. Le diagramme suivant illustre le flux de travail d'Elastic Beanstalk.



Une fois que vous avez créé et déployé votre application, vous pouvez accéder aux informations sur l'application (telles que les métriques, les événements et l'état de l'environnement) via la console Elastic Beanstalk, les API ou les interfaces de ligne de commande, dont l'AWS CLI unifiée.

## Tarification

Il n'y a pas frais supplémentaires pour AWS Beanstalk. Vous ne payez que pour les ressources AWS sous-jacentes utilisées par votre application. Pour de plus amples informations sur la tarification, veuillez consulter la [page détaillée du service Elastic Beanstalk](#).

## Étapes suivantes

Ce manuel contient des informations conceptuelles sur le service web Elastic Beanstalk, ainsi que des informations sur la façon d'utiliser le service pour déployer les applications web. Des sections distinctes expliquent comment utiliser la console Elastic Beanstalk, les outils d'interface de ligne de commande et l'API pour déployer et gérer vos environnements Elastic Beanstalk. Ce manuel explique également comment Elastic Beanstalk s'intègre à d'autres services fournis par Amazon Web Services.

Nous vous recommandons de commencer par lire la page [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour savoir comment utiliser Elastic Beanstalk. La mise en route vous aide à créer, afficher et mettre à jour votre application Elastic Beanstalk, ainsi qu'à modifier ou arrêter votre environnement Elastic Beanstalk. En outre, ce manuel de mise en route décrit les différentes façons vous permettant d'accéder à Elastic Beanstalk.

Pour en savoir plus sur une application Elastic Beanstalk et ses composants, veuillez consulter les pages suivantes.

- [Concepts Elastic Beanstalk \(p. 14\)](#)
- [Glossaire des plateformes Elastic Beanstalk \(p. 25\)](#)
- [Modèle de responsabilité partagée pour la maintenance de la plateforme Elastic Beanstalk \(p. 27\)](#)
- [Stratégie de prise en charge de la plateforme Elastic Beanstalk \(p. 28\)](#)

# Mise en route avec Elastic Beanstalk

Pour vous aider à comprendre le fonctionnement d'AWS Elastic Beanstalk, ce didacticiel vous guide tout au long de la création, l'exploration, la mise à jour et la suppression d'une application Elastic Beanstalk. Son exécution complète prend moins d'une heure.

L'utilisation d'Elastic Beanstalk n'entraîne aucun coût, mais les ressources AWS qu'il crée pour ce tutoriel sont actives (et ne s'exécutent pas dans un environnement de test (sandbox)). Les frais d'utilisation standards pour ces ressources vous sont facturés jusqu'à ce que vous les résiliez à la fin de ce didacticiel. Les frais totaux sont généralement inférieurs à un dollar. Pour de plus amples informations sur la façon de réduire les frais, veuillez consulter [Offre gratuite AWS](#).

## Rubriques

- Configuration : Crédation d'un compte AWS (p. 3)
- Étape 1 : Crédation d'un exemple d'application (p. 3)
- Étape 2 : Exploration de votre environnement (p. 5)
- Étape 3 : Déploiement d'une nouvelle version de votre application (p. 7)
- Étape 4 : Configuration de votre environnement (p. 8)
- Étape 5 : Nettoyage (p. 10)
- Étapes suivantes (p. 11)

## Configuration : Crédation d'un compte AWS

Si vous n'êtes pas déjà un client AWS, vous devez créer un compte AWS. L'inscription vous permet d'accéder à Elastic Beanstalk et aux autres services AWS dont vous avez besoin.

### Pour créer un compte AWS

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Suivez les instructions affichées.

## Étape 1 : Crédation d'un exemple d'application

Dans cette étape, vous créez une nouvelle application à partir d'un exemple d'application préexistante. Elastic Beanstalk prend en charge les plateformes pour différents langages de programmation, serveurs d'applications et conteneurs Docker. Vous choisissez une plateforme lorsque vous créez l'application.

### Création d'une application et d'un environnement

Pour créer votre exemple d'application, vous allez utiliser l'assistant de console Crédier une application web. Il crée une application Elastic Beanstalk et y lance un environnement. Un environnement est l'ensemble des ressources AWS requises pour exécuter votre code d'application.

### Pour créer un exemple d'application

1. Ouvrez la console Elastic Beanstalk en utilisant ce lien : <https://console.aws.amazon.com/elasticbeanstalk/home#/gettingStarted?applicationName=getting-started-app>
2. Ajoutez éventuellement des balises d'application (p. 424).

3. Pour Platform (Plateforme), choisissez une plateforme, puis Create application (Créer une application).

Pour exécuter l'exemple d'application sur des ressources AWS, Elastic Beanstalk effectue les actions suivantes. L'exécution de celles-ci prend environ cinq minutes.

1. Création d'une application Elastic Beanstalk nommée getting-started-app.
2. Lancement d'un environnement nommé GettingStartedApp-env avec les ressources AWS suivantes :
  - Une instance Amazon Elastic Compute Cloud (Amazon EC2) (machine virtuelle)
  - Un groupe de sécurité Amazon EC2
  - Un compartiment Amazon Simple Storage Service (Amazon S3)
  - Des alarmes Amazon CloudWatch
  - Une pile AWS CloudFormation
  - Un nom de domaine
3. Pour plus d'informations sur ces ressources AWS, consultez [the section called "Ressources AWS créées pour l'exemple d'application" \(p. 5\)](#).
4. Crédit d'une nouvelle version d'application nommée Exemple d'application. Il s'agit de l'exemple de fichier d'application Elastic Beanstalk par défaut.
5. Déploiement du code de l'exemple d'application dans l'environnement GettingStartedApp-env.

Pendant le processus de création de l'environnement, la console suit la progression et affiche les événements.



### Creating GettingStarted-env

This will take a few minutes....

```
8:40pm Successfully launched environment: GettingStarted-env
8:39pm Environment health has transitioned from Pending to Ok. Initialization completed 16 seconds ago and took 5 m
8:36pm Added instance [i-045eb69a24818d1d4] to your environment.
8:36pm Waiting for EC2 instances to launch. This may take a few minutes.
8:35pm Created EIP: 34.230.236.246
8:34pm Created security group named:
eb-dv-e-sbj4gzb2dm-stack-AWSEBSecurityGroup-KATGTR06V1J9
8:34pm Environment health has transitioned to Pending. Initialization in progress (running for 8 seconds). There are no
8:34pm Using elasticbeanstalk-us-east-1-270205402845 as Amazon S3 storage bucket for environment data.
8:34pm createEnvironment is starting.
```

Lorsque toutes les ressources sont lancées et que les instances EC2 exécutant l'application réussissent les vérifications de l'état, l'état de l'environnement devient ok. Vous pouvez maintenant utiliser le site de votre application web.

## Ressources AWS créées pour l'exemple d'application

Lorsque vous créez l'exemple d'application, Elastic Beanstalk crée les ressources AWS suivantes :

- Instance EC2 – Une machine virtuelle Amazon EC2 configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble distinct de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse qui traite le trafic web devant votre application web, lui transmet les demandes, traite les ressources statiques et génère des journaux d'accès et d'erreurs.

- Groupe de sécurité de l'instance – Un groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibrEUR de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlENT la charge sur les instances de votre environnement et se déclenchENT si la charge est trop élevEE ou trop faible. Lorsqu'une alarme est déclenchEE, votre groupe Auto Scaling s'adapTE en fonction, à la hausSE ou à la baisSE.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définIES dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme [\*sous-domaine.région.elasticbeanstalk.com\*](#).

## Étape 2 : Exploration de votre environnement

Pour afficher une présentation de l'environnement de votre application Elastic Beanstalk, utilisez la page d'environnement dans la console Elastic Beanstalk.

Pour afficher la présentation de l'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

Le volet de présentation de l'environnement affiche des informations de niveau supérieur sur votre environnement. Cela inclut son nom, son URL, son état de santé actuel, le nom de la version de l'application actuellement déployée et la version de la plateforme sur laquelle l'application s'exécute. Sous le volet de présentation, vous pouvez voir les cinq événements d'environnement les plus récents.

Pour en savoir plus sur les niveaux d'environnement, les plateformes, les versions de l'application et d'autres concepts, veuillez consulter [Concepts \(p. 14\)](#).

The screenshot shows the AWS Elastic Beanstalk console interface. On the left, a sidebar menu lists 'Environments', 'Applications', 'Change history', and a expanded section for 'GettingStartedApp' containing 'Application versions' and 'Saved configurations'. Below this is another expanded section for 'GettingStartedApp-env' containing 'Go to environment' (with a link), 'Configuration', 'Logs', 'Health', 'Monitoring', 'Alarms', 'Managed updates', 'Events', and 'Tags'. At the bottom of this sidebar is a section for 'Recent environments'. The main content area is titled 'GettingStartedApp-env' and displays the URL 'GettingStartedApp-env.eba-nvfhyh93.us-west-2.elasticbeanstalk.com' and the application name 'GettingStartedApp'. It features a 'Health' section with a green circle containing a white checkmark and the status 'Ok', and a 'Recent events' table.

Time	Type	Details
2020-10-15 21:24:42 UTC-0400	INFO	Successfully launched environment.
2020-10-15 21:24:41 UTC-0400	INFO	Application available at <a href="#">http://GettingStartedApp-env.eba-nvfhyh93.us-west-2.elasticbeanstalk.com</a> .
2020-10-15 21:24:41 UTC-0400	INFO	Added instance [i-0efc51f]
2020-10-15 21:24:41 UTC-0400	INFO	Environment health has transitioned to OK.
2020-10-15 21:24:10 UTC-0400	INFO	Instance deployment completed.

Pendant qu'Elastic Beanstalk crée vos ressources AWS et lance votre application, l'environnement est à l'état Pending. Des messages d'état sur les événements de lancement sont ajoutés en permanence à la présentation.

L'URL de l'environnement se trouve en haut de la présentation, sous le nom de l'environnement. Il s'agit de l'URL de l'application web que l'environnement exécute. Choisissez cette URL pour accéder à la page Félicitations de l'exemple d'application.

La page de navigation située sur le côté gauche de la console inclut des liens vers d'autres pages qui contiennent des informations plus détaillées sur votre environnement et fournissent l'accès à des fonctionnalités supplémentaires :

- Configuration – Affiche les ressources provisionnées pour cet environnement, telles que les instances Amazon Elastic Compute Cloud (Amazon EC2) qui hébergent votre application. Vous pouvez configurer certaines des ressources allouées sur cette page.

- État – Affiche le statut et les informations détaillées sur l'état des instances Amazon EC2 qui exécutent votre application.
- Surveillance – Affiche les statistiques relatives à l'environnement, telles que la latence moyenne et l'utilisation de l'UC. Cette page vous permet de créer des alarmes pour les métriques que vous surveillez.
- Événements – Affiche les informations ou les messages d'erreur provenant du service Elastic Beanstalk et des autres services dont cet environnement utilise les ressources.
- Balises – Affiche les balises de l'environnement et vous permet de les gérer. Les balises sont des paires clé-valeur qui sont appliquées à votre environnement.

## Étape 3 : Déploiement d'une nouvelle version de votre application

Vous devrez peut-être déployer régulièrement une nouvelle version de votre application. Vous pouvez déployer une nouvelle version à tout moment, tant qu'aucune autre opération de mise à jour n'est en cours sur votre environnement.

La version de l'application avec laquelle vous avez démarré ce didacticiel est nommée Exemple d'application.

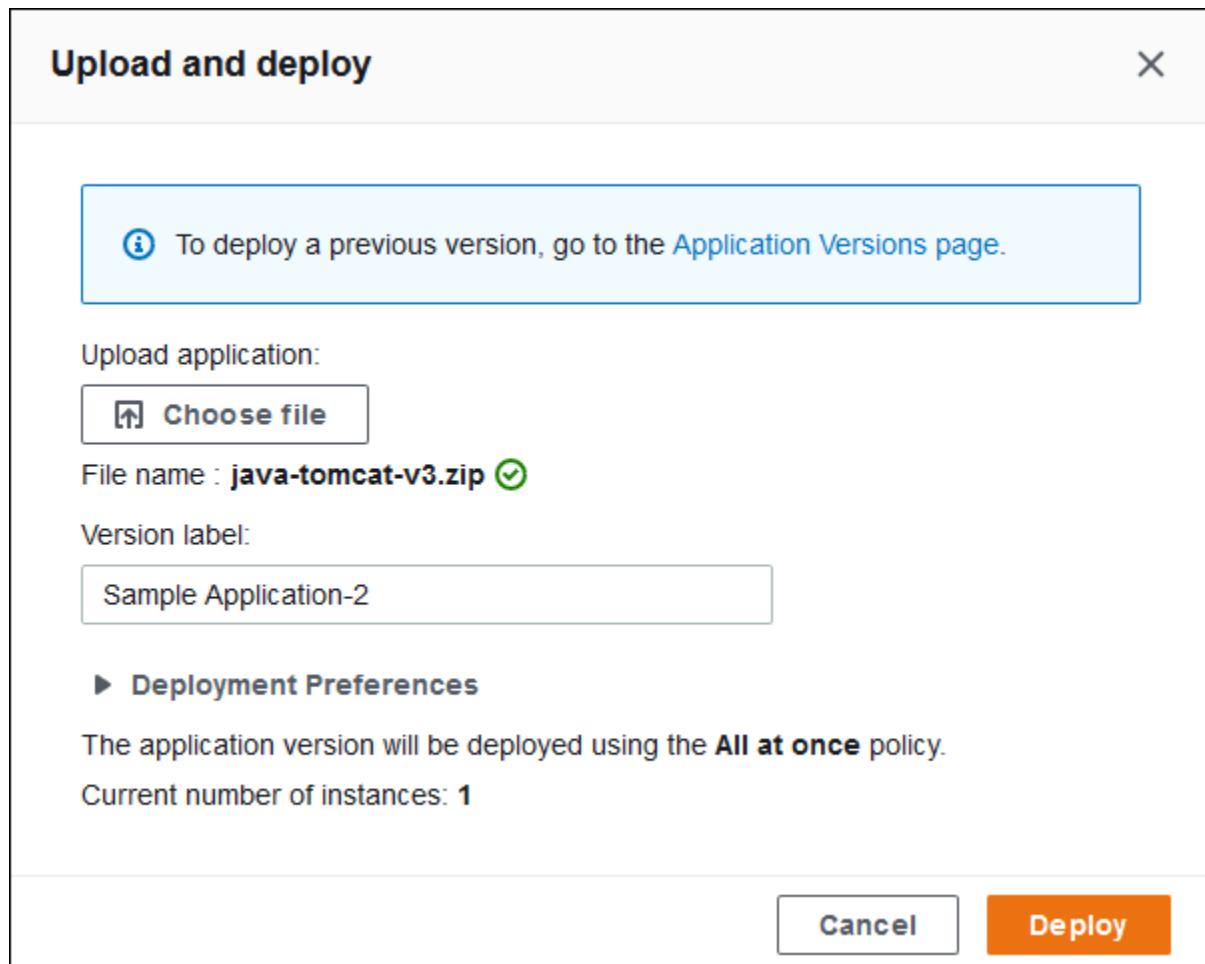
Pour mettre à jour la version de votre application

1. Téléchargez l'exemple d'application qui correspond à la plateforme de votre environnement. Utilisez l'une des applications suivantes.
  - Docker – [docker.zip](#)
  - Docker multi-conteneurs – [docker-multicontainer-v2.zip](#)
  - Docker avec plateforme préconfigurée (Glassfish) – [docker-glassfish-v1.zip](#)
  - Go – [go.zip](#)
  - Corretto – [corretto.zip](#)
  - Tomcat – [tomcat.zip](#)
  - .NET Core sous Linux – [dotnet-core-linux.zip](#)
  - .NET – [dotnet-asp-v1.zip](#)
  - Node.js – [nodejs.zip](#)
  - PHP – [php.zip](#)
  - Python – [python.zip](#)
  - Ruby – [ruby.zip](#)
2. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
3. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

4. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
5. Sélectionnez Choose file (Choisir un fichier), puis chargez le groupe source de l'exemple d'application que vous avez téléchargé.



La console remplit automatiquement le champ Étiquette de version avec une nouvelle étiquette unique. Si vous saisissez votre propre étiquette de version, assurez-vous qu'elle est unique.

6. Choisissez Deploy (Déployer).

Pendant qu'Elastic Beanstalk déploie votre fichier sur vos instances Amazon EC2, vous pouvez afficher l'état du déploiement dans la présentation de l'environnement. Pendant la mise à jour de la version de l'application, le statut Environment Health (État de l'environnement) s'affiche en gris. Une fois le déploiement terminé, Elastic Beanstalk effectue une vérification de l'état de l'application. Lorsque l'application répond à la vérification de l'état, elle est considérée comme saine et le statut redevient vert. La présentation de l'environnement affiche la nouvelle Running version (Version en cours d'exécution), c'est-à-dire le nom que vous avez fourni dans le champ Version label (Étiquette de version).

Elastic Beanstalk télécharge également la nouvelle version de votre application et l'ajoute au tableau des versions de l'application. Pour afficher le tableau, choisissez Application versions (Versions de l'application) sous getting-started-app dans le volet de navigation.

## Étape 4 : Configuration de votre environnement

Vous pouvez configurer votre environnement afin de mieux répondre aux besoins de votre application. Par exemple, si vous avez une application gourmande en calcul, vous pouvez modifier le type d'instance

Amazon Elastic Compute Cloud (Amazon EC2) qui exécute votre application. Pour appliquer des modifications de configuration, Elastic Beanstalk effectue une mise à jour de l'environnement.

Certains changements de configuration sont simples et rapidement effectifs. Certaines modifications nécessitent de supprimer et de recréer des ressources AWS, ce qui peut prendre plusieurs minutes. Lorsque vous modifiez les paramètres de configuration, Elastic Beanstalk vous avertit des temps d'arrêt éventuels de l'application.

## Réalisation d'une modification de la configuration

Dans cet exemple de modification de la configuration, vous modifiez les paramètres de capacité de votre environnement. Vous configurez un environnement évolutif à la charge équilibrée qui a entre deux et quatre instances Amazon EC2 dans son groupe Auto Scaling, puis vous vérifiez que la modification s'est produite. Elastic Beanstalk crée une instance Amazon EC2 supplémentaire, qui s'ajoute à l'instance unique qu'il a initialement créée. Elastic Beanstalk associe ensuite les deux instances à l'équilibrage de charge de l'environnement. Par conséquent, la réactivité de votre application est améliorée et sa disponibilité est augmentée.

Pour modifier la capacité de votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling), remplacez Environment type (Type d'environnement) par Load balanced (Équilibrage de charge).
6. Sur la ligne Instances, définissez la valeur Max sur **4**, puis définissez la valeur Min sur **2**.
7. Choisissez Apply.
8. Un avertissement vous indique que cette mise à jour remplace toutes vos instances actuelles. Choisissez Confirm (Confirmer).
9. Dans le panneau de navigation, sélectionnez Événements.

La mise à jour de l'environnement peut prendre quelques minutes. Pour savoir si elle est terminée, recherchez l'événement Successfully deployed new configuration to environment (Nouvelle configuration déployée avec succès dans l'environnement) dans la liste des événements. Cela confirme que le nombre minimum d'instances Auto Scaling a été défini sur 2. Elastic Beanstalk lance automatiquement la deuxième instance.

## Vérification de la modification de la configuration

Lorsque la mise à jour de l'environnement est terminée et que l'environnement est prêt, vérifiez votre modification.

Pour vérifier l'augmentation de la capacité

1. Dans le panneau de navigation, sélectionnez Santé.

2. Consultez la page Enhanced health overview (Présentation améliorée de l'état).

Vous pouvez voir que deux instances Amazon EC2 sont répertoriées après la ligne Overall (Globale). La capacité de votre environnement est passée à deux instances.

The screenshot shows the Enhanced Health Overview page for the 'GettingStartedApp-env' environment. The top navigation bar includes 'Elastic Beanstalk > Environments > GettingStartedApp-env > Health'. The main title is 'Enhanced Health Overview' with the subtitle 'Instances: 2 Total, 2 Ok'. A link 'Learn more about enhanced health.' is present. Below is a table with columns: Instance ID, Status, Running, and Deployment ID. The table has four rows: 'Overall' (Status: Ok, Running: N/A, Deployment ID: N/A), and two individual instances (Status: Ok, Running: 10 minutes, Deployment ID: 1). The two individual instances are highlighted with a yellow border.

Instance ID	Status	Running	Deployment ID
Overall	Ok	N/A	N/A
i-0867c82b5baab7ef1	Ok	10 minutes	1
i-0106bcf1fb76efdc4	Ok	10 minutes	1

## Étape 5 : Nettoyage

Félicitations ! Vous avez déployé correctement un exemple d'application dans le cloud AWS, téléchargé une nouvelle version et modifié sa configuration pour ajouter une deuxième instance Auto Scaling. Pour vous assurer pour les services que vous n'utilisez pas ne vous seront pas facturés, supprimez toutes les versions de l'application et résiliez l'environnement. Cela supprime également les ressources AWS que l'environnement a créées pour vous.

Pour supprimer l'application et toutes les ressources associées

1. Supprimez toutes les versions de l'application.
  - Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
  - Dans le volet de navigation, choisissez Applications, puis choisissez getting-started-app.
  - Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
  - Sur la page Application versions (Versions de l'application), sélectionnez toutes les versions de l'application que vous souhaitez supprimer.
  - Choisissez Actions, puis choisissez Delete.

- f. Activez Delete versions from Amazon S3 (Supprimer les versions Amazon S3).
  - g. Choisissez Delete (Supprimer), puis Done (Terminé).
2. Terminer l'environnement.
    - a. Dans le volet de navigation, choisissez getting-started-app, puis GettingStartedApp-env dans la liste de l'environnement.
    - b. Choisissez Environment actions (Actions d'environnement), puis Terminate Environment (Arrêter l'environnement).
    - c. Confirmez que vous souhaitez résilier GettingStartedApp-env en saisissant le nom de l'environnement, puis choisissez Résilier.
  3. Supprimez l'application getting-started-app.
    - a. Dans le volet de navigation, choisissez getting-started-app
    - b. Choisissez Actions, puis Delete application (Supprimer l'application).
    - c. Confirmez que vous souhaitez supprimer getting-started-app en saisissant le nom de l'application, puis choisissez Supprimer.

## Étapes suivantes

Maintenant que vous savez comment créer une application Elastic Beanstalk et un environnement, nous vous recommandons de lire [Concepts \(p. 14\)](#). Cette rubrique fournit des informations sur les composants et l'architecture Elastic Beanstalk, et décrit les considérations de conception importantes pour votre application Elastic Beanstalk.

Outre la console Elastic Beanstalk, vous pouvez utiliser les outils suivants pour créer et gérer des environnements Elastic Beanstalk.

### Interface de ligne de commande EB

L'interface de ligne de commande EB est un outil de ligne de commande qui permet de créer et de gérer des environnements. Consultez [Utilisation de l'interface de ligne de commande Elastic Beanstalk \(EB\) \(p. 1027\)](#) pour plus de détails.

### AWS SDK for Java

Le kit AWS SDK for Java fournit une API Java que vous pouvez utiliser pour créer des applications qui utilisent les services d'infrastructure AWS. Avec le kit AWS SDK for Java, vous pouvez démarrer en quelques minutes avec un package unique, téléchargeable et complet, qui inclut la bibliothèque AWS Java, des exemples de codes et de la documentation.

Le kit AWS SDK for Java nécessite le kit de développement J2SE 5.0 ou version ultérieure. Vous pouvez télécharger la dernière version du logiciel Java sur la page <http://developers.sun.com/downloads/>. Le SDK nécessite également Apache Commons (Codec, HttpClient et Logging) et les packages tiers Saxon-HE, qui sont inclus dans le répertoire tiers du SDK.

Pour plus d'informations, consultez [Kit SDK AWS pour Java](#).

### AWS Toolkit for Eclipse

AWS Toolkit for Eclipse est un plug-in open source pour l'IDE Eclipse Java. Vous pouvez l'utiliser pour créer des projets web AWS Java préconfigurés avec le kit AWS SDK for Java, puis déployer les applications web sur Elastic Beanstalk. Le plug-in Elastic Beanstalk s'ajoute à la plateforme WTP (Eclipse Web Tools Platform). La boîte à outils fournit un modèle d'exemple d'application web Travel Log, qui illustre l'utilisation d'Amazon S3 et d'Amazon SNS.

Pour vous assurer que vous disposez de toutes les dépendances WTP, nous vous recommandons de commencer par la distribution Java EE d'Eclipse. Vous pouvez la télécharger à partir de l'adresse <http://eclipse.org/downloads/>.

Pour de plus amples informations sur l'utilisation du plug-in Elastic Beanstalk pour Eclipse, veuillez consulter [AWS Toolkit for Eclipse](#). Pour commencer à créer votre application Elastic Beanstalk à l'aide d'Eclipse, veuillez consulter [Création et déploiement d'applications Java sur Elastic Beanstalk \(p. 110\)](#).

## AWS SDK for .NET

Le kit AWS SDK for .NET vous permet de créer des applications qui utilisent les services d'infrastructure AWS. Avec le kit AWS SDK for .NET, vous pouvez démarrer en quelques minutes avec un package unique, téléchargeable et complet, qui inclut la bibliothèque AWS .NET, des exemples de codes et de la documentation.

Pour plus d'informations, consultez [Kit SDK AWS pour .NET](#). Pour connaître les versions de .NET Framework et de Visual Studio prises en charge, consultez le [Guide du développeur AWS SDK for .NET](#).

## AWS Toolkit for Visual Studio

Le plug-in AWS Toolkit for Visual Studio vous permet de déployer une application .NET existante dans Elastic Beanstalk. Vous pouvez également créer des projets à l'aide des modèles AWS qui sont préconfigurés avec le kit AWS SDK for .NET.

Pour plus d'informations sur les prérequis et l'installation, consultez [AWS Toolkit for Visual Studio](#). Pour commencer à créer votre application Elastic Beanstalk à l'aide de Visual Studio, veuillez consulter [Création et déploiement d'applications .NET sur Elastic Beanstalk \(p. 192\)](#).

## Kit SDK AWS pour JavaScript dans Node.js

Le kit SDK AWS pour JavaScript dans Node.js vous permet de créer des applications basées sur les services d'infrastructure AWS. Avec le kit SDK AWS pour JavaScript dans Node.js, vous pouvez commencer à travailler en quelques minutes : il suffit de télécharger un seul package comprenant la bibliothèque AWS Node.js, des exemples de codes et une documentation.

Pour plus d'informations, consultez [Kit SDK AWS pour JavaScript dans Node.js](#).

## AWS SDK for PHP

Le kit AWS SDK for PHP vous permet de créer des applications basées sur les services d'infrastructure AWS. Avec le kit AWS SDK for PHP, vous pouvez démarrer en quelques minutes avec un package unique, téléchargeable et complet, qui inclut la bibliothèque AWS PHP, des exemples de codes et de la documentation.

Le kit AWS SDK for PHP nécessite PHP version 5.2 ou ultérieure. Pour plus d'informations sur le téléchargement, consultez <http://php.net/>.

Pour plus d'informations, consultez [Kit SDK AWS pour PHP](#).

## AWS SDK for Python (Boto)

Avec le kit AWS SDK for Python (Boto), vous pouvez démarrer en quelques minutes avec un package unique, téléchargeable et complet, qui inclut la bibliothèque AWS Python, des exemples de codes et de la documentation. Vous pouvez développer des applications Python sur des API qui éliminent la complexité liée au codage direct dans une interface de service web.

La bibliothèque tout-en-un offre des API Python conviviales pour le développeur, qui cachent une grande partie des tâches de niveau inférieur associées à la programmation pour le cloud AWS, y compris

L'authentification, les tentatives de demande et la gestion des erreurs. Le kit SDK fournit des exemples pratiques en Python sur la façon d'utiliser les bibliothèques pour créer des applications.

Pour obtenir des informations sur Boto, des exemples de code, de la documentation, des outils et des ressources supplémentaires, veuillez consulter le [Centre pour développeurs Python](#).

## AWS SDK for Ruby

Vous pouvez démarrer en quelques minutes avec un package unique, téléchargeable et complet, qui inclut la bibliothèque AWS Ruby, des exemples de code et de la documentation. Vous pouvez développer des applications Ruby sur des API qui éliminent la complexité liée au codage direct dans une interface de service Web.

La bibliothèque tout-en-un offre des API Ruby conviviales pour le développeur, qui cachent une grande partie des tâches de niveau inférieur associées à la programmation pour le cloud AWS, y compris l'authentification, les tentatives de demande et la gestion des erreurs. Le kit SDK fournit des exemples pratiques en Ruby sur la façon d'utiliser les bibliothèques pour créer des applications.

Pour obtenir des informations sur le kit SDK, des exemples de code, de la documentation, des outils et des ressources supplémentaires, veuillez consulter le [Centre pour développeurs Ruby](#).

# Concepts Elastic Beanstalk

AWS Elastic Beanstalk vous permet de gérer toutes les ressources qui exécutent votre application en tant qu'environnements. Voici quelques concepts clés Elastic Beanstalk.

## Application

Une application Elastic Beanstalk est un ensemble logique de composants Elastic Beanstalk, y compris des environnements, des versions, et des configurations d'environnement. Sur le plan conceptuel, une application est semblable à un dossier dans Elastic Beanstalk.

## Version de l'application

Dans Elastic Beanstalk, une version de l'application fait référence à une itération étiquetée et spécifique du code à déployer pour une application web. Une version de l'application pointe vers un objet Amazon Simple Storage Service (Amazon S3) qui contient le code déployable tel qu'un fichier WAR Java. Une version de l'application fait partie d'une application. Les applications peuvent avoir de nombreuses versions et chaque version de l'application est unique. Dans un environnement d'exploitation, vous pouvez déployer toute version de l'application que vous avez déjà chargée dans l'application ou vous pouvez charger et déployer immédiatement une nouvelle version de l'application. Vous pouvez charger plusieurs versions de l'application pour tester les différences entre une version de votre application web et une autre.

## Environnement

Un environnement est une collection de ressources AWS qui exécutent une version de l'application. Chaque environnement exécute une seule version d'application à la fois, cependant, vous pouvez exécuter la même version d'application ou différentes versions d'application dans de nombreux environnements simultanément. Lorsque vous créez un environnement, Elastic Beanstalk alloue les ressources nécessaires pour exécuter la version de l'application que vous avez spécifiée.

## Niveau de l'environnement

Lorsque vous lancez un environnement Elastic Beanstalk, vous commencez par choisir un niveau d'environnement. Le niveau d'environnement désigne le type d'application qui s'exécute sur l'environnement, et détermine quelles ressources sont mises en service par Elastic Beanstalk pour le prendre en charge. Une application qui traite des demandes HTTP s'exécute dans une [couche d'environnement de serveur Web \(p. 15\)](#). Un environnement backend qui extrait des tâches à partir d'une file d'attente Amazon Simple Queue Service (Amazon SQS) s'exécute dans une [couche d'environnement de travail \(p. 16\)](#).

## Configuration de l'environnement

Une configuration de l'environnement identifie un ensemble de paramètres et des réglages qui définissent le comportement d'un environnement et de ses ressources associées. Lorsque vous mettez à jour les paramètres de configuration d'un environnement, Elastic Beanstalk applique automatiquement les modifications aux ressources existantes ou supprime et déploie de nouvelles ressources (en fonction du type de modification).

# Configuration enregistrée

Une configuration enregistrée est un modèle que vous pouvez utiliser comme point de départ pour créer des configurations d'environnement uniques. Vous pouvez créer et modifier les configurations enregistrées et les appliquer à des environnements, à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande EB, de l'AWS CLI ou de l'API. L'API et l'AWS CLI font référence aux configurations enregistrées en tant que modèles de configuration.

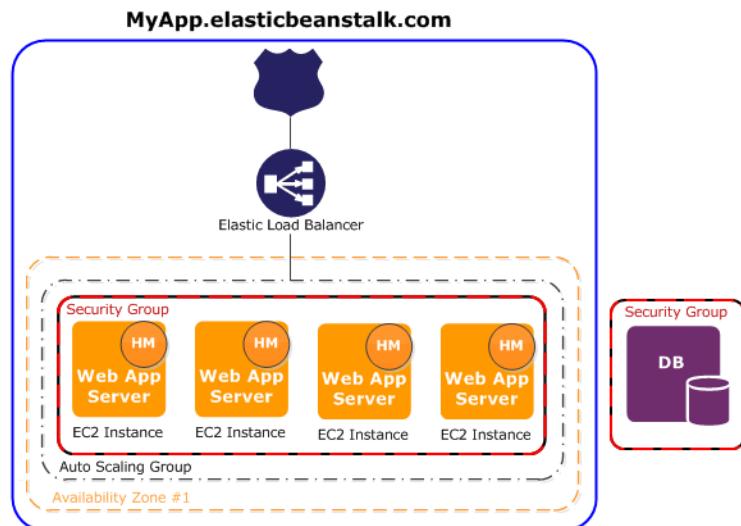
## Plateforme

Une plateforme combine un système d'exploitation, un environnement d'exécution de langage de programmation, un serveur web, un serveur d'applications et des composants Elastic Beanstalk. Vous concevez et ciblez votre application web sur une plateforme. Elastic Beanstalk fournit une grande variété de plateformes sur lesquelles vous pouvez créer vos applications.

Pour plus d'informations, consultez [Plateformes Elastic Beanstalk \(p. 25\)](#).

## Environnements de serveur web

Le schéma suivant illustre un exemple d'architecture Elastic Beanstalk pour un niveau d'environnement de serveur web et montre comment les composants de ce type de niveau d'environnement travaillent ensemble.



L'environnement est le cœur de l'application. Dans le schéma, l'environnement est affiché dans la ligne unie de niveau supérieur. Lorsque vous créez un environnement, Elastic Beanstalk alloue les ressources nécessaires à l'exécution de votre application. Des ressources AWS créées pour un environnement comprennent un équilibrEUR de charge élastique (ELB dans le diagramme), un groupe Auto Scaling et une ou plusieurs instances Amazon Elastic Compute Cloud (Amazon EC2).

Chaque environnement dispose d'un CNAME (URL) qui pointe vers un équilibrEUR de charge. L'environnement dispose d'une URL telle que `myapp.us-west-2.elasticbeanstalk.com`. Cette URL a un alias dans [Amazon Route 53](#) vers une URL Elastic Load Balancing (similaire à `abcdef-123456.us-west-2.elb.amazonaws.com`) en utilisant un enregistrement CNAME. [Amazon Route 53](#) est un

service web de système de noms de domaine (DNS) hautement disponible et évolutif. Cela fournit un acheminement sûr et fiable vers votre infrastructure. Votre nom de domaine que vous avez enregistré avec votre fournisseur DNS réachemine les requêtes vers le CNAME.

L'équilibrer de charge se trouve devant les instances Amazon EC2, qui font partie d'un groupe Auto Scaling. Amazon EC2 Auto Scaling démarre automatiquement les instances Amazon EC2 supplémentaires pour vous adapter à une charge croissante sur votre application. Si la charge sur votre application diminue, Amazon EC2 Auto Scaling arrête des instances, mais laisse toujours au moins une instance en cours d'exécution.

La pile de logiciels en cours d'exécution sur les instances Amazon EC2 dépend du type de conteneur. Un type de conteneur définit la topologie de l'infrastructure et la pile logicielle à utiliser pour cet environnement. Par exemple, un environnement Elastic Beanstalk avec un conteneur Apache Tomcat utilise le système d'exploitation Amazon Linux, le serveur web Apache et le logiciel Apache Tomcat. Pour afficher la liste des types de conteneurs pris en charge, consultez [Plateformes prises en charge par Elastic Beanstalk \(p. 31\)](#). Chaque instance Amazon EC2 qui exécute votre application utilise l'un de ces types de conteneurs. En outre, un composant logiciel appelé le gestionnaire hôte (HM) s'exécute sur chaque instance Amazon EC2. Le gestionnaire hôte est chargé des opérations suivantes :

- Déploiement de l'application
- Regroupement des événements et des métriques pour une récupération via la console, l'API ou la ligne de commande
- Génération d'événements niveau instance
- Surveillance des fichiers journaux de l'application pour les erreurs critiques
- Surveillance du serveur d'applications
- Mise à jour corrective de composants de l'instance
- Rotation des fichiers journaux de votre application et publication de ces derniers sur Amazon S3

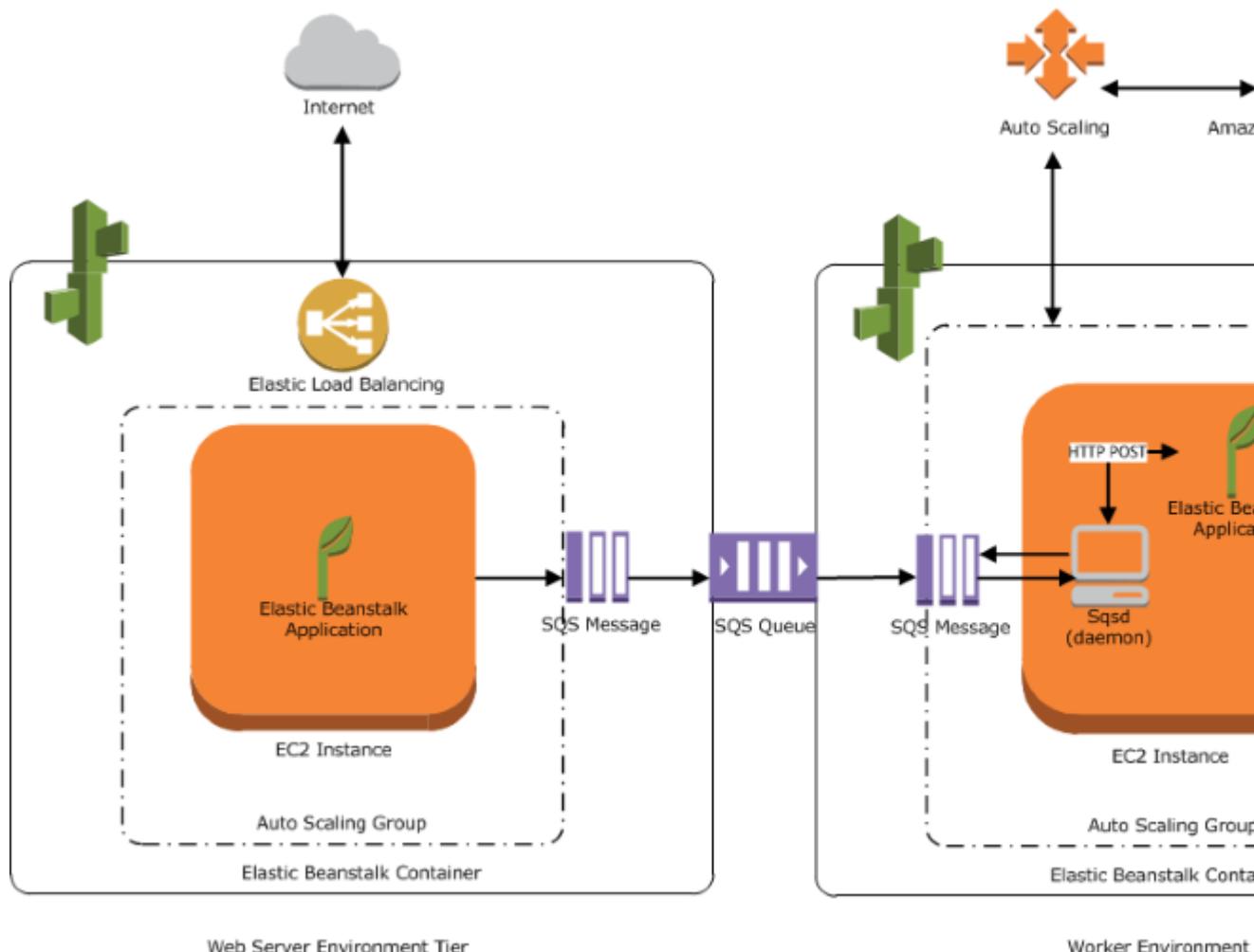
Le gestionnaire hôte rapporte des métriques, des erreurs et des événements, ainsi que le statut de l'instance du serveur, qui sont disponibles via la console Elastic Beanstalk, les API et les interfaces de ligne de commande.

Les instances Amazon EC2 illustrées dans le schéma font partie d'un groupe de sécurité. Un groupe de sécurité définit les règles de pare-feu pour vos instances. Par défaut, Elastic Beanstalk définit un groupe de sécurité, ce qui permet à tout le monde de se connecter via le port 80 (HTTP). Vous pouvez définir plus d'un groupe de sécurité. Par exemple, vous pouvez définir un groupe de sécurité pour votre serveur de base de données. Pour de plus amples informations sur les groupes de sécurité Amazon EC2 et sur la manière de les configurer pour votre application Elastic Beanstalk, veuillez consulter [Groupes de sécurité \(p. 542\)](#).

## Environnements de travail

Les ressources AWS créées pour un niveau d'environnement de travail incluent un groupe Auto Scaling, une ou plusieurs instances Amazon EC2 et un rôle IAM. Pour le niveau d'environnement de travail, Elastic Beanstalk crée et met en service une file d'attente Amazon SQS si vous n'en avez pas déjà. Lorsque vous lancez un niveau d'environnement de travail, Elastic Beanstalk installe les fichiers de prise en charge nécessaires pour le langage de programmation de votre choix et un démon sur chaque instance EC2 du groupe Auto Scaling. Le démon lit les messages d'une file d'attente Amazon SQS. Le démon envoie les données à partir de chaque message lu à l'application web exécutée dans l'environnement de travail en vue de leur traitement. Si votre environnement de travail contient plusieurs instances, chacune d'elles a son propre démon, mais toutes sont lues depuis la même file d'attente Amazon SQS.

Le schéma suivant illustre les différents composants et leurs interactions entre les environnements et les services AWS.



Amazon CloudWatch est utilisé pour la surveillance des alarmes et de l'intégrité. Pour plus d'informations, consultez la section concernant [Création de rapports d'intégrité de base \(p. 834\)](#).

Pour de plus amples informations sur le fonctionnement du niveau d'environnement de travail, veuillez consulter [Environnements de travail Elastic Beanstalk \(p. 521\)](#).

## Considérations relatives à la conception

Comme les applications déployées à l'aide d'Elastic Beanstalk s'exécutent sur des ressources du cloud Amazon, vous devez conserver plusieurs éléments à l'esprit lors de la conception de votre application : scalabilité, sécurité, stockage permanent, tolérance aux pannes, diffusion de contenu, correctifs et mises à jour de logiciels et connectivité. Pour obtenir la liste complète des livres blancs techniques AWS, couvrant des sujets tels que l'architecture, la sécurité et l'économie, accédez à [Livres blancs AWS Cloud Computing](#).

### Evolutivité

Lorsque vous travaillez dans un environnement matériel physique, par opposition à un environnement de cloud, vous pouvez aborder l'évolutivité de deux façons : vous pouvez monter en puissance (mise à l'échelle verticale) ou monter en charge (mise à l'échelle horizontale). L'approche de montée en

puissance nécessite un investissement dans du matériel puissant du fait de l'augmentation des demandes sur l'activité. En revanche, l'approche de montée en charge nécessite de suivre un modèle distribué d'investissement, ainsi le matériel et les acquisitions d'application sont plus ciblés, les ensembles de données sont fédérés et la conception est axée sur le service. L'approche de montée en charge peut devenir très coûteuse, et il y a toujours le risque que la demande vienne à dépasser la capacité. Bien que l'approche de montée en charge soit généralement plus efficace, elle nécessite de prévoir la demande à intervalles réguliers et de déployer l'infrastructure par blocs afin de répondre à la demande. Cette approche entraîne souvent une capacité inutilisée et nécessite une surveillance attentive.

En optant pour le cloud, vous pouvez apporter l'utilisation de votre infrastructure en vous alignant étroitement sur la demande en tirant parti de l'élasticité du cloud. L'élasticité est la rationalisation de l'acquisition et de la libération de ressources, afin que votre infrastructure puisse rapidement diminuer et augmenter la taille des instances en fonction de l'évolution à la demande. Pour implémenter l'élasticité, configurez vos paramètres Auto Scaling pour monter ou descendre en puissance en fonction de métriques issues des ressources dans votre environnement (utilisation des serveurs ou de l'E/S réseau, par exemple). Vous pouvez utiliser Auto Scaling pour ajouter automatiquement de la capacité de calcul lorsque l'utilisation augmente et la supprimer lorsque l'utilisation diminue. Publiez des métriques du système (UC, mémoire, E/S disque, E/S réseau) sur Amazon CloudWatch et configurez des alarmes pour déclencher des actions Auto Scaling ou envoyer des notifications. Pour en savoir plus sur la configuration d'Auto Scaling, consultez [Groupe Auto Scaling pour votre environnement Elastic Beanstalk \(p. 547\)](#).

Les applications Elastic Beanstalk doivent également être aussi sans état que possible, utilisant des composants à tolérance de panne et à couplage souple pouvant monter en puissance en fonction des besoins. Pour plus d'informations sur la conception d'architectures d'applications évolutives pour AWS, lisez le livre blanc [Architecting for the Cloud: Best Practices](#).

## Sécurité

La sécurité sur AWS est une [responsabilité partagée](#). AWS protège les ressources physiques dans votre environnement et assure que le cloud est un endroit sûr où vous pouvez exécuter des applications. Vous êtes responsable de la sécurité des données entrant et sortant de votre environnement Elastic Beanstalk et de la sécurité de votre application.

Pour protéger les informations circulant entre votre application et les clients, configurez SSL. Pour ce faire, vous avez besoin d'un certificat gratuit fourni par AWS Certificate Manager (ACM). Si vous possédez déjà un certificat d'une autorité de certification externe (CA), vous pouvez utiliser ACM pour importer ce certificat par programmation ou à l'aide de l'interface de ligne de commande (CLI) AWS.

Si ACM n'est pas [disponible dans votre région](#), vous pouvez acheter un certificat émis par une autorité de certification externe telle que VeriSign ou Entrust. Vous pouvez, ensuite, charger une clé privée et un certificat tiers ou auto-signé sur AWS Identity and Access Management (IAM) en utilisant l'AWS Command Line Interface. La clé publique du certificat authentifie votre serveur dans le navigateur. Elle sert également de base pour créer la clé de session partagée qui chiffre les données dans les deux directions. Pour obtenir les instructions sur la création, le téléchargement et l'attribution d'un certificat SSL à votre environnement, veuillez consulter [Configuration de HTTPS pour votre environnement Elastic Beanstalk \(p. 792\)](#).

Lorsque vous configurez un certificat SSL pour votre environnement, les données sont chiffrées entre le client et l'équilibreur de charge Elastic Load Balancing de votre environnement. Par défaut, le chiffrement s'arrête au niveau de l'équilibreur de charge, et le trafic entre l'équilibreur de charge et les instances Amazon EC2 n'est pas chiffré.

## Stockage permanent

Les applications Elastic Beanstalk s'exécutent sur des instances Amazon EC2 qui n'ont pas de stockage local permanent. Lorsque les instances Amazon EC2 prennent fin, le système de fichiers local n'est pas enregistré et de nouvelles instances Amazon EC2 démarrent avec un système de fichiers par défaut. Vous devez concevoir votre application pour stocker des données dans une source de données persistante.

Amazon Web Services offre un certain nombre de services de stockage permanent que vous pouvez utiliser pour votre application. Le tableau suivant les répertorie.

Service de stockage	Documentation du service	Intégration d'Elastic Beanstalk
Amazon S3	Documentation Amazon Simple Storage Service	Utilisation d'Elastic Beanstalk avec Amazon S3 (p. 1003)
Amazon Elastic File System	Documentation Amazon Elastic File System	Utilisation d'Elastic Beanstalk avec Amazon Elastic File System (p. 918)
Amazon Elastic Block Store	Amazon Elastic Block Store  Feature Guide: Elastic Block Store	
Amazon DynamoDB	Documentation Amazon DynamoDB	Utilisation d'Elastic Beanstalk avec Amazon DynamoDB (p. 917)
Amazon Relational Database Service (RDS)	Documentation Amazon Relational Database Service	Utilisation d'Elastic Beanstalk avec Amazon RDS (p. 991)

## Tolérance aux pannes

En règle générale, vous devez être pessimiste lors de la conception d'une architecture pour le cloud. Vous devez toujours concevoir, mettre en œuvre et déployer en prévoyant une reprise automatique après une défaillance. Utilisez plusieurs zones de disponibilité pour vos instances Amazon EC2 et pour Amazon RDS. Les zones de disponibilité sont sur le plan conceptuel comme des centres de données logiques. Utilisez Amazon CloudWatch pour obtenir plus de visibilité concernant l'intégrité de votre application Elastic Beanstalk et prendre les mesures appropriées en cas de panne de matériel ou de dégradation des performances. Configurez vos paramètres Auto Scaling pour maintenir votre flotte d'instances Amazon EC2 à une taille fixe afin que les instances Amazon EC2 défectueuses soient remplacées par de nouvelles. Si vous utilisez Amazon RDS, définissez alors la période de conservation des sauvegardes, afin qu'Amazon RDS puisse effectuer des sauvegardes automatiques.

## Diffusion de contenu

Lorsque des utilisateurs se connectent à votre site web, leurs demandes peuvent être acheminées via un certain nombre de réseaux individuels. Ainsi les utilisateurs peuvent subir des performances médiocres en raison d'une latence élevée. Amazon CloudFront peut aider à améliorer les problèmes de latence en répartissant votre contenu web (par exemple, des images, des vidéos, etc) sur un réseau d'emplacements périphériques dans le monde entier. Les utilisateurs finaux sont acheminés vers l'emplacement périphérique le plus proche, afin de diffuser le contenu de manière optimale. CloudFront fonctionne de façon transparente avec Amazon S3, qui stocke durablement les versions originales et définitives de vos fichiers. Pour en savoir plus sur Amazon CloudFront, veuillez consulter <http://aws.amazon.com/cloudfront>.

## Correctifs et mises à jour de logiciels

Elastic Beanstalk met régulièrement à jour ses plateformes avec de nouveaux logiciels et correctifs. Elastic Beanstalk ne met pas automatiquement à niveau les environnements en cours d'exécution vers de nouvelles versions de plateforme, mais vous pouvez lancer une [mise à jour de plate-forme \(p. 496\)](#) pour mettre à jour votre environnement en cours d'exécution. Les mises à jour de plateforme utilisent des [mises à jour propagées \(p. 489\)](#) pour maintenir votre application disponible en appliquant des modifications par lots.

## Connectivité

Elastic Beanstalk doit être capable de se connecter aux instances de votre environnement pour compléter les déploiements. Lorsque vous déployez une application Elastic Beanstalk à l'intérieur d'un Amazon VPC, la configuration requise pour activer la connectivité dépend du type d'environnement Amazon VPC que vous créez :

- Pour les environnements à instance unique, aucune configuration supplémentaire n'est requise, car Elastic Beanstalk alloue à chaque instance Amazon EC2 une adresse IP Elastic publique qui permet à l'instance de communiquer directement avec Internet.
- Pour les environnements évolutifs et à équilibrage de charge dans un Amazon VPC avec des sous-réseaux publics et privés, vous devez procéder comme suit :
  - Créez un équilibrEUR de charge dans le sous-réseau public pour acheminer le trafic entrant à partir d'Internet vers les instances Amazon EC2.
  - Créez un périphérique de traduction d'adresses réseau (NAT) pour acheminer le trafic sortant des instances Amazon EC2 des sous-réseaux privés vers Internet.
  - Créez des règles de routage de trafic sortant et entrant pour les instances Amazon EC2 à l'intérieur du sous-réseau privé.
  - Si vous utilisez une instance NAT, configurez les groupes de sécurité pour l'instance NAT et les instances Amazon EC2 pour permettre la communication Internet.
- Pour un environnement évolutif et d'équilibrage de charge dans un Amazon VPC qui dispose d'un sous-réseau public, aucune configuration supplémentaire n'est obligatoire car les instances Amazon EC2 sont configurées avec une adresse IP publique qui permet aux instances de communiquer avec Internet.

Pour de plus amples informations sur l'utilisation d'Elastic Beanstalk avec Amazon VPC, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#).

# Rôles de service, profils d'instance et stratégies utilisateur

Lorsque vous créez un environnement, AWS Elastic Beanstalk vous invite à spécifier deux fonctions AWS Identity and Access Management (IAM), une fonction de service et un profil d'instance. La [fonction du service \(p. 21\)](#) est exercée par Elastic Beanstalk pour utiliser d'autres services AWS en votre nom. Le [profil d'instance \(p. 22\)](#) s'applique aux instances de votre environnement et leur permet de récupérer les [versions d'application \(p. 14\)](#) depuis Amazon Simple Storage Service (Amazon S3), de charger les journaux dans Amazon S3 et d'effectuer d'autres tâches qui varient en fonction du type d'environnement et de la plateforme.

Le meilleur moyen d'obtenir un profil d'instance et un rôle de service correctement configurés consiste à [créer un environnement exécutant un exemple d'application \(p. 439\)](#) dans la console Elastic Beanstalk ou via l'interface de ligne de commande Elastic Beanstalk. Lorsque vous créez un environnement, les clients créent les rôles nécessaires et leur attribuent des [stratégies gérées \(p. 946\)](#) qui incluent toutes les autorisations nécessaires.

Outre les deux rôles que vous affectez à votre environnement, vous pouvez créer des [stratégies utilisateur \(p. 24\)](#) et les appliquer aux utilisateurs et aux groupes IAM de votre compte pour permettre aux utilisateurs de créer et de gérer des environnements et des applications Elastic Beanstalk. Elastic Beanstalk fournit des stratégies gérées pour un accès complet et un accès en lecture seule.

Vous pouvez créer vos propres profils d'instance et stratégies utilisateur pour des scénarios avancés. Si vos instances ont besoin d'accéder à des services qui ne sont pas inclus dans les stratégies par défaut, vous pouvez ajouter des stratégies supplémentaires à l'élément par défaut ou en créer une autre. Vous pouvez également créer des stratégies utilisateur plus restrictives si la stratégie gérée est trop permissive. Consultez le [AWS Identity and Access Management Guide de l'utilisateur](#) pour un examen approfondi des autorisations AWS.

## Rubriques

- [Rôle de service Elastic Beanstalk \(p. 21\)](#)
- [Profil d'instance Elastic Beanstalk \(p. 22\)](#)
- [Stratégie utilisateur Elastic Beanstalk \(p. 24\)](#)

## Rôle de service Elastic Beanstalk

Un rôle de service correspond au rôle IAM exercé par Elastic Beanstalk lorsqu'il appelle d'autres services en votre nom. Par exemple, Elastic Beanstalk utilise la fonction du service que vous spécifiez lors de la création d'un environnement Elastic Beanstalk lorsqu'il appelle les API Amazon Elastic Compute Cloud (Amazon EC2), Elastic Load Balancing et Amazon EC2 Auto Scaling pour recueillir des informations sur l'état de ses ressources AWS dans le cadre de la [surveillance améliorée de l'état \(p. 837\)](#).

La stratégie gérée `AWSElasticBeanstalkEnhancedHealth` contient toutes les autorisations dont Elastic Beanstalk a besoin pour surveiller l'état de l'environnement :

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [
```

```
        "elasticloadbalancing:DescribeInstanceHealth",
        "elasticloadbalancing:DescribeLoadBalancers",
        "elasticloadbalancing:DescribeTargetHealth",
        "ec2:DescribeInstances",
        "ec2:DescribeInstanceStatus",
        "ec2:GetConsoleOutput",
        "ec2:AssociateAddress",
        "ec2:DescribeAddresses",
        "ec2:DescribeSecurityGroups",
        "sns:GetQueueAttributes",
        "sns:GetQueueUrl",
        "autoscaling:DescribeAutoScalingGroups",
        "autoscaling:DescribeAutoScalingInstances",
        "autoscaling:DescribeScalingActivities",
        "autoscaling:DescribeNotificationConfigurations",
        "sns:Publish"
    ],
    "Resource": [
        "*"
    ]
}
]
```

Cette stratégie inclut également des actions Amazon SQS qui permettent à Elastic Beanstalk de surveiller l'activité de file d'attente pour les environnements de travail.

Lorsque vous créez un environnement à l'aide la console Elastic Beanstalk, Elastic Beanstalk vous invite à créer un rôle de service nommé `aws-elasticbeanstalk-service-role` avec l'ensemble d'autorisations par défaut et une stratégie d'approbation qui permet à Elastic Beanstalk d'exercer le rôle de service. Si vous activez les [mises à jour de la plateforme gérée \(p. 501\)](#), Elastic Beanstalk attache une autre stratégie disposant des autorisations d'activation de cette fonctionnalité.

De la même manière, lorsque vous créez un environnement à l'aide de la commande [the section called “eb create” \(p. 1078\)](#) de l'interface de ligne de commande Elastic Beanstalk, et que vous ne spécifiez pas de rôle de service par l'intermédiaire de l'option `--service-role`, Elastic Beanstalk crée le rôle de service par défaut `aws-elasticbeanstalk-service-role`. Si le rôle de service par défaut existe déjà, Elastic Beanstalk l'utilise pour le nouvel environnement.

Lorsque vous créez un environnement en utilisant l'action `CreateEnvironment` de l'API Elastic Beanstalk et que vous ne spécifiez pas de rôle de service, Elastic Beanstalk crée un rôle lié à un service de surveillance. Il s'agit d'un type de fonction de service unique prédéfini par Elastic Beanstalk pour inclure toutes les autorisations nécessaires afin que le service appelle d'autres services AWS en votre nom. Le rôle lié à un service est associé à votre compte. Elastic Beanstalk le crée une fois, puis le réutilise lors de la création d'environnements supplémentaires. Vous pouvez également utiliser IAM pour créer par avance le rôle lié à un service de surveillance de votre compte. Lorsque votre compte dispose d'un rôle lié à un service de surveillance, vous pouvez l'utiliser pour créer un environnement à l'aide de l'API Elastic Beanstalk, de la console Elastic Beanstalk ou de l'interface de ligne de commande EB. Pour de plus amples informations sur l'utilisation des rôles liés à un service avec les environnements Elastic Beanstalk, veuillez consulter [Utilisation des rôles liés à un service pour Elastic Beanstalk \(p. 935\)](#).

Pour de plus amples informations sur les rôles de service, veuillez consulter [Gestion des rôles de service Elastic Beanstalk \(p. 926\)](#).

## Profil d'instance Elastic Beanstalk

Un profil d'instance est un rôle IAM qui est appliqué aux instances lancées dans votre environnement Elastic Beanstalk. Lorsque vous créez un environnement Elastic Beanstalk, vous spécifiez le profil d'instance utilisé lorsque vos instances :

- récupèrent les [versions de l'application \(p. 14\)](#) depuis Amazon Simple Storage Service (Amazon S3) ;
- écrivent des journaux dans Amazon S3 ;
- dans des [environnements intégrés AWS X-Ray \(p. 638\)](#), chargent des données de débogage dans X-Ray
- coordonnent les déploiements de conteneurs avec Amazon Elastic Container Service (dans les environnements Docker à conteneurs multiples) ;
- lisent à partir d'une file d'attente Amazon Simple Queue Service (Amazon SQS) (dans les environnements de travail) ;
- effectuent des choix principaux avec Amazon DynamoDB (dans les environnements de travail) ;
- publient des métriques de vérification de l'état des instances dans Amazon CloudWatch (dans les environnements de travail).

La stratégie gérée `AWSElasticBeanstalkWebTier` contient des instructions permettant aux instances de votre environnement de charger des journaux dans Amazon S3 et d'envoyer des informations de débogage à X-Ray :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "BucketAccess",
      "Action": [
        "s3:Get*",
        "s3>List*",
        "s3:PutObject"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:s3:::elasticbeanstalk-*",
        "arn:aws:s3:::elasticbeanstalk-*/*"
      ]
    },
    {
      "Sid": "XRayAccess",
      "Action": [
        "xray:PutTraceSegments",
        "xray:PutTelemetryRecords",
        "xray:GetSamplingRules",
        "xray:GetSamplingTargets",
        "xray:GetSamplingStatisticSummaries"
      ],
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Sid": "CloudWatchLogsAccess",
      "Action": [
        "logs:PutLogEvents",
        "logs>CreateLogStream",
        "logs:DescribeLogStreams",
        "logs:DescribeLogGroups"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:logs:*log-group:/aws/elasticbeanstalk*"
      ]
    },
    {
      "Sid": "ElasticBeanstalkHealthAccess",
      "Action": [
        "elasticbeanstalk:Ping"
      ]
    }
  ]
}
```

```
        "elasticbeanstalk:PutInstanceStatistics"
    ],
    "Effect": "Allow",
    "Resource": [
        "arn:aws:elasticbeanstalk::::application/*",
        "arn:aws:elasticbeanstalk::::environment/*"
    ]
}
}
```

Elastic Beanstalk fournit également des stratégies gérées nommées `AWSElasticBeanstalkWorkerTier` et `AWSElasticBeanstalkMulticontainerDocker` pour les autres cas d'utilisation. Elastic Beanstalk attache toutes ces stratégies au profil d'instance par défaut, `aws-elasticbeanstalk-ec2-role`, lorsque vous créez un environnement avec la console ou l'interface de ligne de commande EB.

Si votre application web a besoin d'un accès à d'autres services AWS, ajoutez des instructions ou des stratégies gérées au profil d'instance autorisant l'accès à ces services.

Pour de plus amples informations sur les profils d'instance, veuillez consulter [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#).

## Stratégie utilisateur Elastic Beanstalk

Pour éviter d'utiliser votre compte racine ou de partager les informations d'identification, créez des utilisateurs IAM pour chaque personne utilisant Elastic Beanstalk. Pour optimiser la sécurité, accordez uniquement à ces utilisateurs l'autorisation d'accéder aux services et aux fonctionnalités dont ils ont besoin.

Elastic Beanstalk exige des autorisations non seulement pour les actions de ses propres API, mais aussi pour plusieurs autres services AWS. Elastic Beanstalk utilise des autorisations utilisateur pour lancer toutes les ressources d'un environnement, y compris les instances EC2, un équilibrer de charge Elastic Load Balancing et un groupe Auto Scaling. Elastic Beanstalk utilise également des autorisations utilisateur pour enregistrer les journaux et les modèles dans Amazon Simple Storage Service (Amazon S3), envoyer des notifications à Amazon SNS, attribuer des profils d'instance et publier des métriques dans CloudWatch. Elastic Beanstalk a besoin d'autorisations AWS CloudFormation pour organiser les mises à jour et les déploiements des ressources. Ce service a également besoin d'autorisations Amazon RDS pour créer des bases de données si nécessaire, et d'autorisations Amazon SQS pour créer des files d'attente pour les environnements de travail.

Pour de plus amples informations sur les stratégies utilisateur, veuillez consulter [Gestion des stratégies utilisateur Elastic Beanstalk \(p. 946\)](#).

# Plateformes Elastic Beanstalk

AWS Elastic Beanstalk fournit une grande variété de plateformes sur lesquelles vous pouvez créer vos applications. Vous concevez votre application web sur l'une de ces plateformes. Elastic Beanstalk déploie votre code sur la version de la plateforme que vous avez sélectionnée pour créer un environnement d'application actif.

Elastic Beanstalk propose des plateformes dans différents langages de programmation, serveurs d'applications, et conteneurs Docker. Certaines plateformes possèdent plusieurs versions prise en charge simultanément.

## Rubriques

- [Glossaire des plateformes Elastic Beanstalk \(p. 25\)](#)
- [Modèle de responsabilité partagée pour la maintenance de la plateforme Elastic Beanstalk \(p. 27\)](#)
- [Stratégie de prise en charge de la plateforme Elastic Beanstalk \(p. 28\)](#)
- [Plateformes prises en charge par Elastic Beanstalk \(p. 31\)](#)
- [Plateformes Linux Elastic Beanstalk \(p. 32\)](#)
- [Déploiement d'applications Elastic Beanstalk à partir de conteneurs Docker \(p. 46\)](#)
- [Création et déploiement d'applications Go sur Elastic Beanstalk \(p. 98\)](#)
- [Création et déploiement d'applications Java sur Elastic Beanstalk \(p. 110\)](#)
- [Utilisation de .NET Core sous Linux \(p. 157\)](#)
- [Création et déploiement d'applications .NET sur Elastic Beanstalk \(p. 192\)](#)
- [Déploiement d'applications Node.js sur Elastic Beanstalk \(p. 251\)](#)
- [Création et déploiement d'applications PHP sur Elastic Beanstalk \(p. 290\)](#)
- [Travail avec Python \(p. 356\)](#)
- [Création et déploiement d'applications Ruby sur Elastic Beanstalk \(p. 384\)](#)

## Glossaire des plateformes Elastic Beanstalk

Voici les principaux termes liés aux plateformes AWS Elastic Beanstalk et à leur cycle de vie.

### Exécution

Langage de programmation propre au logiciel d'environnement d'exécution (infrastructure, bibliothèques, interpréteur, VM, etc.) requis pour exécuter votre code d'application.

### Composants Elastic Beanstalk

Composants logiciels qu'Elastic Beanstalk ajoute à une plateforme pour activer la fonctionnalité Elastic Beanstalk. Par exemple, l'agent amélioré pour l'état de santé est nécessaire pour recueillir et rapporter les données d'état de santé.

### Plateforme

Combinaison entre un système d'exploitation (OS), un environnement d'exécution, un serveur web, un serveur d'applications et les composants Elastic Beanstalk. Les plateformes comportent les composants disponibles pour exécuter l'application.

### Version de plateforme

Combinaison entre les versions spécifiques d'un système d'exploitation (OS), un environnement d'exécution, un serveur web, un serveur d'applications et les composants Elastic Beanstalk. Vous créez un environnement Elastic Beanstalk basé sur une version de plateforme et déployez votre application sur cette dernière.

Une version de plateforme possède un numéro de version sémantique au format X.Y.Z, où X est la version majeure, Y est la version mineure et Z est la version de correctif.

Une version de plateforme peut se trouver dans l'un des états suivants :

- Prise en charge – Version de plateforme qui comprend uniquement des composants pris en charge. Tous les composants n'ont pas atteint leur fin de vie, selon les indications de leurs fournisseurs respectifs (propriétaires, AWS ou tiers, ou communautés). Ils reçoivent régulièrement des correctifs ou des mises à jour mineures de la part de leurs fournisseurs. Elastic Beanstalk met à votre disposition les versions de plateforme prises en charge pour la création d'environnements.
- Hors service – Version de plateforme avec un ou plusieurs composants hors service, qui ont atteint leur fin de vie, selon les indications de leurs fournisseurs. Les versions de plateforme hors service ne sont pas disponibles à l'utilisation dans les environnements Elastic Beanstalk pour les clients nouveaux ou existants.

Pour plus d'informations sur les composants retirés, consultez [the section called “Stratégie de prise en charge de la plateforme” \(p. 28\)](#).

#### Branche de plateforme

Une ligne de versions de plateforme partageant des versions spécifiques (généralement majeures) de certains de leurs composants, notamment le système d'exploitation (OS), l'exécution ou des composants Elastic Beanstalk. Par exemple : Python 3.6 s'exécute sur Amazon Linux 64 bits ; IIS 10.0 s'exécute sur Windows Server 2016 64 bits. Chaque version de plateforme successive dans la branche est une mise à jour par rapport à la précédente.

La dernière version de plateforme dans chaque branche de plateforme est disponible sans conditions pour vous permettre la création d'un environnement. Les versions précédentes de la plateforme dans la branche sont toujours prises en charge : vous pouvez créer un environnement basé sur une version précédente de la plateforme si vous l'avez utilisée dans un environnement au cours des 30 derniers jours. Cependant, ces versions précédentes de plateforme n'incluent pas les composants les plus récents et ne sont donc pas recommandées.

Une branche de plateforme peut se trouver dans l'un des états suivants :

- Prise en charge – Branche de plateforme actuelle. Elle comprend uniquement des composants pris en charge. Elle bénéficie de mises à jour continues de la plateforme et est recommandée pour une utilisation dans les environnements de production. Pour obtenir une liste de branches de plateformes prises en charge, consultez [Plateformes Elastic Beanstalk prises en charge](#) dans le guide Plateformes AWS Elastic Beanstalk.
- Bêta – Version préliminaire, pré-version de la branche de plateforme. Elle est de nature expérimentale. Elle peut bénéficier des mises à jour continues de la plateforme pendant un certain temps, mais ne dispose pas d'une prise en charge à long terme. Une branche de plateforme bêta n'est pas recommandée pour une utilisation dans des environnements de production. Utilisez-la uniquement à des fins d'évaluation. Pour obtenir la liste des branches de plateforme bêta, consultez [Versions de plateformes Elastic Beanstalk en bêta publique](#) dans le guide Plateformes AWS Elastic Beanstalk.
- Obsolète – Branche de plateforme avec un ou plusieurs composants obsolètes. Elle bénéficie de mises à jour continues de la plateforme, mais n'est pas recommandée pour une utilisation dans des environnements de production. Pour obtenir une liste des branches de plateforme obsolètes, consultez [Versions de plateformes Elastic Beanstalk Platform dont le retrait est programmé](#) dans le guide Plateformes AWS Elastic Beanstalk.
- Hors service – Branche de plateforme avec un ou plusieurs composants hors service. Elle ne bénéficie plus de mises à jour de la plateforme et n'est pas recommandée pour une utilisation dans des environnements de production. Les branches de plateforme retirées ne sont pas répertoriées dans le guide Plateformes AWS Elastic Beanstalk. Elastic Beanstalk ne met pas les versions de plateforme des branches de plateforme hors service à votre disposition pour la création d'environnement.

Un composant pris en charge n'a pas de date de retrait prévue par son fournisseur (propriétaire ou communauté). Le fournisseur peut être AWS ou une tierce partie. Un composant obsolète a une date de retrait planifiée par son fournisseur. Un composant hors service a atteint sa fin de vie et n'est plus pris en charge par son fournisseur. Pour plus d'informations sur les composants hors service, consultez [the section called "Stratégie de prise en charge de la plateforme" \(p. 28\)](#).

Si votre environnement utilise une branche de plateforme obsolète ou hors service, nous vous recommandons de la mettre à jour vers une version de plateforme dans une branche de plateforme prise en charge. Pour plus d'informations, consultez [the section called "Mises à jour de plateforme" \(p. 496\)](#).

#### Mise à jour de plateforme

Publication de nouvelles versions de plateforme qui contiennent les mises à jour de certains composants de la plateforme : système d'exploitation, environnement d'exécution, serveur web, serveur d'applications et composants Elastic Beanstalk. Les mises à jour de plateforme suivent la taxonomie de la sémantique des versions et peuvent contenir plusieurs niveaux :

- Mise à jour majeure – Mise à jour contenant des modifications incompatibles avec les versions de plateforme existantes. Vous devrez peut-être modifier votre application pour qu'elle s'exécute correctement sur une nouvelle version majeure. Une mise à jour majeure a un nouveau numéro de version majeure de plateforme.
- Mise à jour mineure – Mise à jour ajoutant des fonctionnalités rétrocompatibles avec une version de plateforme existante. Vous n'avez pas besoin de modifier votre application pour qu'elle s'exécute correctement sur une nouvelle version mineure. Une mise à jour mineure a un nouveau numéro de version mineure de plateforme.
- Mise à jour corrective – Mise à jour consistant en publications de maintenance (correctifs de bogues, mises à jour de sécurité et améliorations de performances) qui sont rétrocompatibles avec une version de plateforme existante. Une mise à jour de correctif a un nouveau numéro de correctif de version de plateforme.

#### Mises à jour gérées

Fonctionnalité Elastic Beanstalk qui applique automatiquement les mises à jour correctives et mineures au système d'exploitation (OS), à l'environnement d'exécution, au serveur web, au serveur d'applications et aux composants Elastic Beanstalk pour une version de plateforme prise en charge par Elastic Beanstalk. Une mise à jour gérée applique une version de plateforme plus récente dans la même branche de plateforme à votre environnement. Vous pouvez configurer les mises à jour gérées pour n'appliquer que les mises à jour correctives ou les mises à jour mineures et correctives. Vous pouvez également désactiver entièrement les mises à jour gérées.

Pour plus d'informations, consultez [Mises à jour gérées de la plateforme \(p. 501\)](#).

## Modèle de responsabilité partagée pour la maintenance de la plateforme Elastic Beanstalk

AWS et nos clients partagent la responsabilité d'obtenir un niveau élevé de sécurité et de conformité des composants logiciels. Ce modèle de responsabilité partagée permet de réduire votre charge opérationnelle.

Pour plus d'informations, consultez le [modèle de responsabilité partagée AWS](#).

AWS Elastic Beanstalk vous aide à exécuter votre part du modèle de responsabilité partagée en vous fournissant une fonction de mises à jour gérées. Cette fonction applique automatiquement les mises à jour correctives et mineures correspondant à une version de plateforme prise en charge par Elastic Beanstalk. Si une mise à jour gérée échoue, Elastic Beanstalk vous informe de l'échec pour que vous en teniez compte et que vous preniez des mesures immédiates.

Pour plus d'informations, consultez [Mises à jour gérées de la plateforme \(p. 501\)](#).

De plus, Elastic Beanstalk effectue les opérations suivantes :

- Publie sa [stratégie de prise en charge de la plateforme \(p. 28\)](#) et son calendrier de mise hors service pour les 12 mois prochains.
- Libère les mises à jour correctives, mineures et majeures des composants du système d'exploitation (OS), du moteur d'exécution, du serveur d'applications et du serveur Web généralement dans les 30 jours suivant leur disponibilité. Elastic Beanstalk est responsable de la création de mises à jour des composants Elastic Beanstalk présents sur ses versions de plateforme prises en charge. Toutes les autres mises à jour sont fournies directement par leurs fournisseurs (propriétaires ou communautés).

Vous êtes chargé d'effectuer les tâches suivantes :

- Mettre à jour tous les composants que vous contrôlez (identifiés comme Client dans le [modèle de responsabilité partagée AWS](#)). Il s'agit notamment d'assurer la sécurité de votre application, de vos données et de tous les composants dont votre application a besoin et que vous avez téléchargé.
- Assurez-vous que vos environnements Elastic Beanstalk s'exécutent sur une version de plateforme prise en charge et migrez tout environnement exécuté sur une version de plateforme hors service vers une version prise en charge.
- Résolvez tous les problèmes qui surviennent lors de l'échec des tentatives de mises à jour gérées et réessayez la mise à jour.
- Appliquez vous-même un correctif au système d'exploitation, à l'environnement d'exécution et au serveur web si vous n'avez pas choisi les mises à jour gérées par Elastic Beanstalk. Pour cela, vous pouvez [appliquer les mises à jour de plateforme manuellement \(p. 496\)](#) ou appliquer directement les correctifs aux composants sur toutes les ressources d'environnement concernées.
- Gérez la sécurité et la conformité de tous les services AWS que vous utilisez en dehors d'Elastic Beanstalk selon le [modèle de responsabilité partagée AWS](#).

## Stratégie de prise en charge de la plateforme Elastic Beanstalk

AWS Elastic Beanstalk fournit une variété de plateformes pour exécuter des applications sur AWS. Elastic Beanstalk prend en charge les succursales de plateformes qui continuent de recevoir des mises à jour mineures et correctives de la part de leurs fournisseurs (propriétaires ou communauté). Pour obtenir une définition complète des termes associés, veuillez consulter [Glossaire des plateformes Elastic Beanstalk \(p. 25\)](#).

Lorsqu'un composant (système d'exploitation [SE], environnement d'exécution, serveur d'applications ou serveur web) d'une branche de plateforme prise en charge est marqué EOL (End of Life, fin de vie) par son fournisseur, Elastic Beanstalk marque cette branche de plateforme comme retirée. Lorsqu'une branche de plateforme est marquée comme retirée, Elastic Beanstalk ne la met plus à disposition des clients Elastic Beanstalk existants et nouveaux pour les déploiements dans les nouveaux environnements. Les branches de plateforme retirées sont disponibles pour les environnements client existants pour une durée de 90 jours à compter de la date de retrait publiée.

Elastic Beanstalk n'est pas en mesure de fournir des mises à jour de sécurité, un support technique ou des correctifs logiciels pour les branches de plateforme retirées une fois que le fournisseur a marqué comme EOL (End of Live, Fin de vie) leur composant. Pour les clients existants exécutant un environnement Elastic Beanstalk sur une version de plateforme retirée au-delà de la période de 90 jours, Elastic Beanstalk devra peut-être supprimer automatiquement les composants Elastic Beanstalk et transférer au client la prise en charge de la responsabilité de la gestion et du support continu de l'application exécutée et des ressources AWS associées. Pour continuer à profiter des améliorations importantes proposées par les fournisseurs des composants en matière de sécurité, de performances et de fonctionnalités, nous vous encourageons

vivement à mettre à jour tous vos environnements Elastic Beanstalk vers une version de plateforme prise en charge.

## Calendrier de mise hors service des branches de plateforme

Les tableaux suivants répertorient les composants de plateforme existants qui sont marqués comme hors service ou dont les dates de mise hors service sont comprises dans les 12 mois prochains. Les tableaux indiquent la date de fin de disponibilité pour les branches de plateforme Elastic Beanstalk qui contiennent ces composants.

Pour obtenir une liste des branches de plateforme mises hors service d'Elastic Beanstalk connexes, consultez les [versions de plateformes dont le retrait est programmé](#) dans le guide Plateformes AWS Elastic Beanstalk.

### Versions de système d'exploitation (OS)

Version du système d'exploitation	Date de fin de disponibilité		
Amazon Linux AMI (AL1)	30 juin 2022		
Windows Server 2012 R2	1 mai 2022		

### Versions d'environnement d'exécution

Version d'environnement d'exécution	Date de fin de disponibilité		
Corretto 8 avec Tomcat 7	30 avril 2022		
Corretto 11 avec Tomcat 7	30 avril 2022		
Docker préconfiguré – GlassFish 5.0 avec Java 8	30 juin 2022		
Java 7 SE	30 juin 2022		
Java 7 avec Tomcat 7	30 juin 2022		
Node.js 10.x	30 avril 2022 (Amazon Linux 2)/30 juin 2022 (Amazon Linux AMI)		
PHP 7.2-7.3	30 avril 2022 (Amazon Linux 2)/30 juin 2022 (Amazon Linux AMI)		
Python 3.6	30 juin 2022		

Version d'environnement d'exécution	Date de fin de disponibilité		
Ruby 2.4, Ruby 2.6	30 juin 2022		
Ruby 2.5	30 avril 2022 (Amazon Linux 2)/30 juin 2022 (Amazon Linux AMI)		

#### Versions de serveur d'applications

Version du serveur d'applications	Date de fin de disponibilité		
Tomcat 7	30 avril 2022 (Amazon Linux 2)/30 juin 2022 (Amazon Linux AMI)		

## Branches de plate-forme mises hors service

Les tableaux suivants répertorient les composants de plateforme qui ont été marqués comme étant hors service dans le passé. Les tableaux indiquent la date à laquelle Elastic Beanstalk a retiré les branches de plateforme qui contenaient ces composants.

#### Versions de système d'exploitation (OS)

Version du système d'exploitation	Date de mise hors service de la plateforme		
Windows Server 2008R2	2 octobre 2019		

#### Versions de serveur web

Version du serveur web	Date de fin de disponibilité		
Serveur HTTP Apache 2.2	31 octobre 2020		
Nginx 1.12.2	31 octobre 2020		

#### Versions d'environnement d'exécution

Version d'environnement d'exécution	Date de fin de disponibilité		
Go 1.3–1.10	31 octobre 2020		
Java 6	31 octobre 2020		
Node.js 4.x–8.x	31 octobre 2020		

Version d'environnement d'exécution	Date de fin de disponibilité		
PHP 5.4–5.6	31 octobre 2020		
PHP 7.0–7.1	31 octobre 2020		
Python 2.6, 2.7, 3.4	31 octobre 2020		
Ruby 1.9.3	31 octobre 2020		
Ruby 2.0–2.3	31 octobre 2020		

#### Versions de serveur d'applications

Version du serveur d'applications	Date de fin de disponibilité		
Tomcat 6	31 octobre 2020		
Tomcat 8	31 octobre 2020		

## Plateformes prises en charge par Elastic Beanstalk

AWS Elastic Beanstalk fournit une grande variété de plateformes sur lesquelles vous pouvez créer vos applications. Vous concevez votre application web sur l'une de ces plateformes. Elastic Beanstalk déploie votre code sur la version de la plateforme que vous avez sélectionnée pour créer un environnement d'application actif.

Elastic Beanstalk fournit des plateformes pour les langages de programmation (Go, Java, Node.js, PHP, Python, Ruby), serveurs d'applications (Tomcat, Passenger, Puma) et des conteneurs Docker. Certaines plateformes possèdent plusieurs versions prise en charge simultanément.

Elastic Beanstalk fournit les ressources nécessaires à l'exécution de votre application, y compris une ou plusieurs instances Amazon EC2. La pile logicielle qui s'exécute sur les instances Amazon EC2 dépend de la version de plateforme spécifique que vous avez sélectionnée pour votre environnement.

Vous pouvez utiliser le nom de la pile de solutions répertorié sous le nom de la version de plateforme pour lancer un environnement avec l'[interface de ligne de commande EB \(p. 1027\)](#), l'[API Elastic Beanstalk](#) ou l'[interface de ligne de commande AWS](#). Vous pouvez également récupérer les noms de pile de solutions à partir du service avec l'API `ListAvailableSolutionStacks` (`aws elasticbeanstalk list-available-solution-stacks` dans l'interface de ligne de commande AWS). Cette opération renvoie toutes les piles de solutions que vous pouvez utiliser pour créer un environnement.

#### Note

Chaque plateforme comporte des versions de plateforme prises en charge ou hors service. Vous pouvez toujours créer un environnement basé sur une version de plateforme prise en charge. Les versions de plateforme hors service ne sont disponibles pour les environnements client que pendant un délai de 90 jours à compter de la date de mise hors service publiée. Pour obtenir la liste des dates de mise hors service des versions de plateforme, veuillez consulter [Calendrier de mise hors service des branches de plateforme \(p. 29\)](#).

Lorsqu'Elastic Beanstalk met à jour une plateforme, les versions de plateforme antérieures continuent d'être prises en charge, mais elles ne comportent pas les composants mis à jour et leur utilisation n'est pas recommandée. Nous vous recommandons de passer à la version de

plateforme la plus récente. Vous pouvez encore créer un environnement basé sur une version de plateforme antérieure si vous avez utilisé celle-ci dans les 30 derniers jours (avec le même compte et dans la même région).

Vous pouvez personnaliser et configurer le logiciel dont dépend votre application dans votre plateforme. Pour en savoir plus, veuillez consulter [Personnalisation du logiciel sur des serveurs Linux \(p. 740\)](#) et [Personnalisation du logiciel sur des serveurs Windows \(p. 753\)](#). Des notes de mise à jour détaillées sont disponibles pour les mises à jour récentes dans le document suivant : [Notes de mise à jour AWS Elastic Beanstalk](#).

## Versions de plateforme prises en charge

Toutes les versions de plateforme actuelles sont répertoriées dans [Plateformes prises en charge par Elastic Beanstalk](#) dans le guide Plateformes AWS Elastic Beanstalk. Chaque section spécifique à la plateforme pointe également vers l'historique de la plateforme, une liste des versions précédentes de cette dernière. Pour accéder directement à la liste des versions d'une plateforme spécifique, utilisez l'un des liens suivants.

- [Docker](#)
- [Docker multiconteneurs](#)
- [Docker préconfigurée](#)
- [Go](#)
- [Java SE](#)
- [Tomcat](#)
- [.NET Core sous Linux](#)
- [.NET sous Windows Server](#)
- [Node.js](#)
- [PHP](#)
- [Python](#)
- [Ruby](#)

## Plateformes Linux Elastic Beanstalk

AWS Elastic Beanstalk fournit une grande variété de plateformes sur lesquelles vous pouvez créer vos applications. Vous concevez votre application web sur l'une de ces plateformes. Elastic Beanstalk déploie votre code sur la version de la plateforme que vous avez sélectionnée pour créer un environnement d'application actif.

Elastic Beanstalk propose des plateformes dans différents langages de programmation, serveurs d'applications, et conteneurs Docker. Certaines plateformes possèdent plusieurs versions prise en charge simultanément.

Pour en savoir plus sur les plateformes Elastic Beanstalk, veuillez consulter [Plateformes Elastic Beanstalk \(p. 25\)](#).

La plupart des plateformes prises en charge par Elastic Beanstalk se basent sur le système d'exploitation Linux. Elles se basent plus précisément sur Amazon Linux, une distribution Linux fournie par AWS. Les plateformes Linux Elastic Beanstalk utilisent des instances Amazon Elastic Compute Cloud (Amazon EC2) qui exécutent Amazon Linux. Pour en savoir plus, veuillez consulter [Amazon Linux](#) dans le Guide de l'utilisateur Amazon EC2 pour instances Linux.

Les plateformes Linux Elastic Beanstalk fournissent de nombreuses fonctionnalités prêtes à l'emploi. Vous pouvez étendre les plateformes de plusieurs façons pour prendre en charge votre application. Pour plus d'informations, consultez [the section called “Extension des plateformes Linux” \(p. 34\)](#).

## Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation

Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

### Rubriques

- [Versions de plateforme Linux \(p. 33\)](#)
- [Liste des plateformes Linux Elastic Beanstalk \(p. 33\)](#)
- [Extension des plateformes Linux Elastic Beanstalk \(p. 34\)](#)

## Versions de plateforme Linux

AWS fournit deux versions d'Amazon Linux : [Amazon Linux 2](#) et [AMI Amazon Linux](#). Voici quelques améliorations importantes d'Amazon Linux 2 par rapport à l'AMI Amazon Linux :

- Amazon Linux 2 offre une prise en charge à long terme.
- Amazon Linux 2 est disponible en tant qu'images de machine virtuelle pour le développement et les tests sur site.
- Amazon Linux 2 est livré avec des composants mis à jour : le noyau Linux, la bibliothèque C, le compilateur et les outils. Il utilise également le service systemd et le gestionnaire de systèmes par opposition au système d'initialisation System V de l'AMI Amazon Linux.

Elastic Beanstalk gère des versions de plateforme avec les deux versions d'Amazon Linux. Pour de plus amples informations sur les versions de plateforme prises en charge, veuillez consulter [Plateformes prises en charge par Elastic Beanstalk \(p. 31\)](#).

### Note

Les versions de plateforme Amazon Linux 2 sont incompatibles avec les versions de plateforme précédentes de l'AMI Amazon Linux. Si vous migrez votre application Elastic Beanstalk vers Amazon Linux 2, veuillez consulter [the section called "Mise à niveau vers Amazon Linux 2" \(p. 508\)](#).

## Liste des plateformes Linux Elastic Beanstalk

La liste suivante répertorie les plateformes Linux prises en charge par Elastic Beanstalk pour les différents langages de programmation et les conteneurs Docker, ainsi que des liens vers des chapitres les concernant dans le présent manuel du développeur.

- [Docker \(p. 46\)](#)
- [Go \(p. 98\)](#)
- [Java \(p. 110\)](#)

- .NET Core sous Linux (p. 157)
- Node.js (p. 251)
- PHP (p. 290)
- Python (p. 356)
- Ruby (p. 384)

## Extension des plateformes Linux Elastic Beanstalk

Les [plateformes Linux AWS Elastic Beanstalk \(p. 32\)](#) fournissent de nombreuses fonctionnalités pour prendre en charge le développement et l'exécution de votre application. Si nécessaire, vous pouvez étendre les plateformes de plusieurs façons pour configurer des options, installer des logiciels, ajouter des fichiers et des commandes de démarrage, fournir des instructions de génération et d'exécution et ajouter des scripts d'initialisation s'exécutant au cours des différentes étapes de provisionnement des instances Amazon Elastic Compute Cloud (Amazon EC2) de votre environnement.

### Buildfile et Procfile

Certaines plateformes vous permettent de personnaliser la façon dont vous créez ou préparez votre application, mais aussi de spécifier les processus qui exécutent votre application. Chaque rubrique de plate-forme mentionne spécifiquement Buildfile et/ou Procfile si la plate-forme les prend en charge. Recherchez votre plateforme spécifique sous [Plates-formes \(p. 25\)](#).

Pour toutes les plateformes de prise en charge, la syntaxe et la sémantique sont identiques et sont décrites dans cette page. Chaque rubrique de plateforme mentionne l'utilisation spécifique de ces fichiers pour la création et l'exécution d'applications dans leurs langages respectifs.

#### Buildfile

Pour spécifier une commande de génération et de configuration personnalisée pour votre application, placez un fichier nommé `Buildfile` dans le répertoire racine de votre source d'application. Le nom de fichier est sensible à la casse. Utilisez la syntaxe suivante pour votre `Buildfile`.

```
<process_name>: <command>
```

La commande dans votre fichier `Buildfile` doit correspondre à l'expression régulière suivante : `^[A-Za-z0-9_-]+:\s*[^\s].*\$`.

Elastic Beanstalk ne surveille pas l'application exécutée avec un fichier `Buildfile`. Utilisez un `Buildfile` pour les commandes qui s'exécutent pendant de courtes durées et s'arrêtent après avoir terminé leurs tâches. Pour les processus d'applications de longue durée qui ne doivent pas se fermer, utilisez un [Procfile \(p. 35\)](#).

Tous les chemins d'accès dans le `Buildfile` sont par rapport à la racine du groupe source. Dans l'exemple suivant de fichier `Buildfile`, `build.sh` est un script shell qui se trouve à la racine du bundle de fichiers source.

#### Example Buildfile

```
make: ./build.sh
```

Si vous souhaitez fournir des étapes de construction personnalisées, nous vous recommandons d'utiliser des hooks de plateforme `predeploy` pour tout sauf les commandes les plus simples, plutôt qu'un `Buildfile`. Les hooks de plateforme permettent des scripts plus riches et une meilleure gestion des erreurs. Les hooks de plateforme sont décrits dans la section suivante.

## Procfile

Pour spécifier des commandes personnalisées pour démarrer et exécuter votre application, placez un fichier nommé `Procfile` dans le répertoire racine de votre source d'application. Le nom de fichier est sensible à la casse. Utilisez la syntaxe suivante pour votre `Procfile`. Vous pouvez spécifier une ou plusieurs commandes.

```
<process_name1>: <command1>
<process_name2>: <command2>
...
...
```

Chaque ligne de votre fichier `Procfile` doit correspondre à l'expression régulière suivante : `^ [A-Za-z0-9_-]+ : \s* [^\s].* $`.

Utilisez un fichier `Procfile` pour les processus d'application de longue durée qui ne doivent pas se fermer. Elastic Beanstalk s'attend à ce que les processus s'exécutent à partir du fichier `Procfile` le fassent en continu. Elastic Beanstalk surveille ces processus et redémarre tout processus qui s'arrête. Pour les processus de courte durée, utilisez un [Buildfile \(p. 34\)](#).

Tous les chemins d'accès dans le `Procfile` sont par rapport à la racine du groupe source. L'exemple `Procfile` suivant définit trois processus. Le premier, appelé `web` dans l'exemple, est l'application Web principale.

### Example Procfile

```
web: bin/myserver
cache: bin/mycache
foo: bin/fooapp
```

Elastic Beanstalk configure le serveur proxy de sorte à transférer les demandes vers votre application web principale sur le port 5000. Vous pouvez configurer ce numéro de port. Une utilisation courante pour un `Procfile` est de transmettre ce numéro de port à votre application en tant qu'argument de commande. Pour plus d'informations sur la configuration du proxy, développez la section Configuration du proxy inverse sur cette page.

Elastic Beanstalk capture les flux d'erreurs et de sortie standard à partir des processus `Procfile` dans les fichiers journaux. Elastic Beanstalk donne le nom du processus aux fichiers journaux et les stocke dans `/var/log`. Par exemple, le processus `web` dans l'exemple précédent génère des journaux nommés `web-1.log` et `web-1.error.log` pour `stdout` et `stderr`, respectivement.

## Hooks de plateforme

Les hooks de plateforme sont spécifiquement conçus pour étendre la plateforme de votre environnement. Il s'agit de scripts personnalisés et autres fichiers exécutables que vous déployez dans le cadre du code source de votre application et qui sont exécutés par Elastic Beanstalk au cours de différentes étapes de provisionnement d'instance.

### Note

Les hooks de plateforme ne sont pas pris en charge sur les versions de plateforme de l'AMI Amazon Linux (précédemment Amazon Linux 2).

## Hooks de plateforme de déploiement d'applications

Un déploiement d'application se produit lorsque vous fournissez un nouveau bundle source pour le déploiement ou lorsque vous apportez une modification de configuration qui nécessite la résiliation et la récréation de toutes les instances d'environnement.

Pour fournir des hooks de plateforme qui s'exécutent pendant un déploiement d'application, placez les fichiers sous le répertoire `.platform/hooks` de votre bundle source, dans l'un des sous-réertoires suivants.

- **prebuild** – Les fichiers s'exécutent après que le moteur de plateforme Elastic Beanstalk a téléchargé et extrait le bundle de fichiers source de l'application, et avant qu'il installe et configure l'application et le serveur web.

Les fichiers **prebuild** s'exécutent après l'exécution des commandes trouvées dans la section [commands \(p. 746\)](#) de tout fichier de configuration et avant l'exécution des commandes **Buildfile**.

- **predeploy** – Les fichiers s'exécutent après que le moteur de plateforme Elastic Beanstalk a installé et configuré l'application et le serveur web, et avant qu'il les déploie dans leur emplacement d'exécution final.

Les fichiers **predeploy** s'exécutent après l'exécution des commandes trouvées dans la section [container\\_commands \(p. 748\)](#) de tout fichier de configuration et avant l'exécution des commandes **Procfile**.

- **postdeploy** – Les fichiers s'exécutent après que le moteur de plateforme Elastic Beanstalk a déployé l'application et le serveur proxy.

Il s'agit de la dernière étape du workflow de déploiement.

## Hooks de plateforme de déploiement de configuration

Un déploiement de configuration se produit lorsque vous apportez des modifications de configuration qui mettent uniquement à jour les instances d'environnement sans les recréer. Les mises à jour des options suivantes provoquent une mise à jour de la configuration.

- [Propriétés de l'environnement et paramètres spécifiques à la plateforme \(p. 632\)](#)
- [Fichiers statiques \(p. 790\)](#)
- [AWS X-Ray Démon \(p. 638\)](#)
- [Stockage des journaux et streaming \(p. 641\)](#)
- Port d'application (pour plus de détails, développez la section Configuration du proxy inverse de cette page)

Pour fournir des hooks qui s'exécutent lors d'un déploiement de configuration, placez-les sous le répertoire `.platform/confighooks` de votre bundle source. Les trois mêmes sous-répertoires que pour les hooks de déploiement d'applications s'appliquent.

### En savoir plus sur les hooks de plateforme

Les fichiers exécutables peuvent être des fichiers binaires ou des fichiers script commençant par une ligne `#!` et contenant leur chemin d'interpréteur, par exemple `#!/bin/bash`. Tous les fichiers doivent avoir l'autorisation d'exécution. Utilisez la commande `chmod +x` pour définir l'autorisation d'exécution sur vos fichiers exécutables.

Elastic Beanstalk exécute les fichiers dans chacun de ces répertoires et dans l'ordre lexicographique des noms de fichier. Tous les fichiers sont exécutés en tant qu'utilisateur `root`. Le répertoire de travail en cours (`cwd`) pour les hooks de plateforme est le répertoire racine de l'application. Pour les fichiers **prebuild** et **predeploy**, il s'agit du répertoire intermédiaire de l'application ; pour les fichiers **postdeploy**, il s'agit du répertoire en cours de l'application. Si un des fichiers échoue (fin d'exécution avec un code de sortie différent de zéro), le déploiement échoue.

Les fichiers hook ont accès à toutes les propriétés d'environnement que vous avez définies dans les options d'application, ainsi qu'aux variables d'environnement système `HOME`, `PATH` et `PORT`.

Pour obtenir des valeurs de variables d'environnement et d'autres options de configuration dans vos scripts de hook de plateforme, vous pouvez utiliser l'utilitaire `get-config` fourni par Elastic Beanstalk sur les instances d'environnement. Pour plus d'informations, consultez [the section called "Outils de script de plateforme" \(p. 43\)](#).

## Fichiers de configuration

Vous pouvez ajouter des [fichiers de configuration \(p. 737\)](#) au répertoire `.ebextensions` du code source de votre application afin de configurer différents aspects de votre environnement Elastic Beanstalk. Entre autres choses, les fichiers de configuration vous permettent de personnaliser le logiciel et d'autres fichiers sur les instances de votre environnement, mais aussi d'exécuter des commandes d'initialisation sur les instances. Pour de plus amples informations, veuillez consulter [the section called “Serveur Linux” \(p. 740\)](#).

Vous pouvez également définir des [options de configuration \(p. 658\)](#) à l'aide de fichiers de configuration. De nombreuses options permettent de contrôler le comportement de la plateforme, et certaines d'entre elles sont [spécifiques à la plateforme \(p. 727\)](#).

Sur les plateformes Amazon Linux 2, nous vous recommandons d'utiliser les Buildfile, Procfile et les hooks de plateforme pour configurer et exécuter du code personnalisé sur vos instances d'environnement pendant le provisionnement d'instance. Ces mécanismes sont décrits dans les sections précédentes de cette page. Vous pouvez toujours utiliser des commandes et des commandes de conteneur dans les fichiers de configuration `.ebextensions`, mais elles ne sont pas aussi simples à utiliser. Par exemple, écrire des scripts de commande dans un fichier YAML peut être difficile d'un point de vue syntaxique. Vous devez toujours utiliser des fichiers de configuration `.ebextensions` pour tout script nécessitant une référence à une ressource AWS CloudFormation.

## Configuration du proxy inverse

Toutes les versions de plateforme Amazon Linux 2 utilisent nginx comme serveur proxy inverse par défaut. Les plateformes Tomcat, Node.js, PHP et Python prennent également en charge Apache HTTPD comme alternative. Pour sélectionner Apache sur ces plateformes, définissez l'option `ProxyServer` dans l'espace de noms `aws:elasticbeanstalk:environment:proxy` sur `apache`. Toutes les plateformes permettent la configuration du serveur proxy de manière uniforme, comme décrit dans cette section.

### Note

Sur les versions de la plateforme d'AMI Amazon Linux (précédemment Amazon Linux 2), vous devrez peut-être configurer les serveurs proxy différemment. Vous trouverez ces détails hérités dans les [rubriques respectives de la plateforme \(p. 25\)](#) dans ce guide.

Elastic Beanstalk configure le serveur proxy sur les instances de votre environnement pour transférer le trafic web vers l'application web principale sur l'URL racine de l'environnement ; par exemple, `http://my-env.elasticbeanstalk.com`.

Par défaut, Elastic Beanstalk configure le proxy pour transférer les demandes entrantes sur le port 80 vers votre application web principale sur le port 5000. Vous pouvez configurer ce numéro de port en définissant la propriété d'environnement `PORT` à l'aide de l'espace de noms [aws:elasticbeanstalk:application:environment \(p. 697\)](#) dans un fichier de configuration, comme illustré dans l'exemple suivant.

```
option_settings:
  - namespace: aws:elasticbeanstalk:application:environment
    option_name: PORT
    value: <main_port_number>
```

Pour plus d'informations sur la définition des variables d'environnement pour votre application, consultez [the section called “Paramètres d'option” \(p. 738\)](#).

Votre application doit écouter sur le port configuré pour elle dans le proxy. Si vous modifiez le port par défaut à l'aide de la propriété d'environnement `PORT`, votre code peut y accéder en lisant la valeur de la variable d'environnement `PORT`. Par exemple, appelez `os.Getenv("PORT")` dans Go, ou `System.getenv("PORT")` dans Java. Si vous configurez votre proxy pour envoyer du trafic vers

plusieurs processus d'application, vous pouvez configurer plusieurs propriétés d'environnement et utiliser leurs valeurs à la fois dans la configuration proxy et dans votre code d'application. Une autre option consiste à transmettre la valeur de port au processus en tant qu'argument de commande dans le `Procfile`. Pour plus d'informations à ce sujet, développez la section `Buildfile` et `Procfile` sur cette page.

## Configuration de nginx

Elastic Beanstalk utilise nginx comme proxy inverse par défaut pour mapper votre application à votre équilibrer de charge Elastic Load Balancing. Elastic Beanstalk fournit une configuration nginx par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

### Note

Lorsque vous ajoutez ou modifiez un fichier de configuration `.conf` nginx, veillez à l'encoder en UTF-8.

Pour étendre la configuration nginx par défaut d'Elastic Beanstalk, ajoutez les fichiers de configuration `.conf` à un dossier nommé `.platform/nginx/conf.d/` dans le bundle de fichiers source de votre application. La configuration nginx d'Elastic Beanstalk inclut automatiquement les fichiers `.conf` dans ce dossier.

```
~/workspace/my-app/
|-- .platform
|   '-- nginx
|       '-- conf.d
|           '-- myconf.conf
`-- other source files
```

Pour remplacer complètement la configuration nginx par défaut d'Elastic Beanstalk, incluez une configuration dans votre bundle de fichiers source à l'emplacement `.platform/nginx/nginx.conf`:

```
~/workspace/my-app/
|-- .platform
|   '-- nginx
|       '-- nginx.conf
`-- other source files
```

Si vous remplacez la configuration nginx d'Elastic Beanstalk, ajoutez la ligne suivante à votre fichier `nginx.conf` afin d'extraire les configurations d'Elastic Beanstalk pour la [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#), les mappages d'application automatiques et les fichiers statiques.

```
include conf.d/elasticbeanstalk/*.conf;
```

## Configuration d'Apache HTTPD

Les plateformes Tomcat, Node.js, PHP et Python vous permettent de choisir le serveur proxy HTTPD Apache comme alternative à nginx. Ce n'est pas la valeur par défaut. L'exemple suivant montre comment configurer Elastic Beanstalk pour utiliser Apache HTTPD.

### Example `.ebextensions/httpd-proxy.config`

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
```

Vous pouvez étendre la configuration Apache Elastic Beanstalk par défaut avec vos fichiers de configuration supplémentaires. Sinon, vous pouvez remplacer complètement la configuration Apache Elastic Beanstalk par défaut.

Pour étendre la configuration Apache Elastic Beanstalk par défaut, ajoutez les fichiers de configuration .conf à un dossier nommé .platform/httpd/conf.d dans le bundle de fichiers source de votre application. La configuration Apache Elastic Beanstalk par défaut inclut automatiquement les fichiers .conf dans ce dossier.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd-proxy.config
|-- .platform
|   -- httpd
|       -- conf.d
|           -- port5000.conf
|           -- ssl.conf
-- index.jsp
```

Par exemple, la configuration Apache 2.4 suivante ajoute un écouteur sur le port 5000 :

Example .platform/httpd/conf.d/port5000.conf

```
listen 5000
<VirtualHost *:5000>
    <Proxy *>
        Require all granted
    </Proxy>
    ProxyPass / http://localhost:8080/ retry=0
    ProxyPassReverse / http://localhost:8080/
    ProxyPreserveHost on

    ErrorLog /var/log/httpd/elasticbeanstalk-error_log
</VirtualHost>
```

Pour remplacer complètement la configuration Apache Elastic Beanstalk par défaut, incluez une configuration dans votre bundle de fichiers source sur .platform/httpd/conf/httpd.conf.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd-proxy.config
|-- .platform
|   `-- httpd
|       `-- conf
|           `-- httpd.conf
-- index.jsp
```

Si vous remplacez la configuration Apache Elastic Beanstalk, ajoutez les lignes suivantes à votre fichier httpd.conf afin d'extraire les configurations Elastic Beanstalk pour la [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#), les mappages d'application automatiques et les fichiers statiques.

```
IncludeOptional conf.d/elasticbeanstalk/*.conf
```

Si vous migrez votre application Elastic Beanstalk vers une plateforme Amazon Linux 2, assurez-vous de consulter également les informations de la section [the section called “Mise à niveau vers Amazon Linux 2” \(p. 508\)](#).

## Rubriques

- [Exemple d'application avec extensions \(p. 40\)](#)
- [Flux de travail \(workflow\) de déploiement d'instance \(p. 40\)](#)
- [Outils de script de plateforme \(p. 43\)](#)

## Exemple d'application avec extensions

L'exemple suivant présente un bundle de fichiers source d'application avec plusieurs fonctionnalités d'extensibilité prises en charge par les plateformes Elastic Beanstalk Amazon Linux 2 : un fichier `Procfile`, des fichiers de configuration `.ebextensions`, des hooks personnalisés et des fichiers de configuration de proxy.

```
~/my-app/
|-- web.jar
|-- Procfile
|-- readme.md
|-- .ebextensions/
|   |-- options.config      # Option settings
|   `-- cloudwatch.config  # Other .ebextensions sections, for example files and
    container commands
`-- .platform/
    |-- nginx/               # Proxy configuration
    |   |-- nginx.conf
    |   `-- conf.d/
    |       `-- custom.conf
    |-- hooks/                # Application deployment hooks
    |   |-- prebuild/
    |   |   |-- 01_set_secrets.sh
    |   |   `-- 12_update_permissions.sh
    |   |-- predeploy/
    |   |   `-- 01_some_service_stop.sh
    |   |-- postdeploy/
    |   |   |-- 01_set_tmp_file_permissions.sh
    |   |   |-- 50_run_something_after_app_deployment.sh
    |   |   `-- 99_some_service_start.sh
    |-- confighooks/          # Configuration deployment hooks
    |   |-- prebuild/
    |   |   `-- 01_set_secrets.sh
    |   |-- predeploy/
    |   |   `-- 01_some_service_stop.sh
    |   |-- postdeploy/
    |       |-- 01_run_something_after_config_deployment.sh
    |       `-- 99_some_service_start.sh
```

### Note

Certaines de ces extensions ne sont pas prises en charge sur les versions de plateforme de l'AMI Amazon Linux (précédemment Amazon Linux 2).

## Flux de travail (workflow) de déploiement d'instance

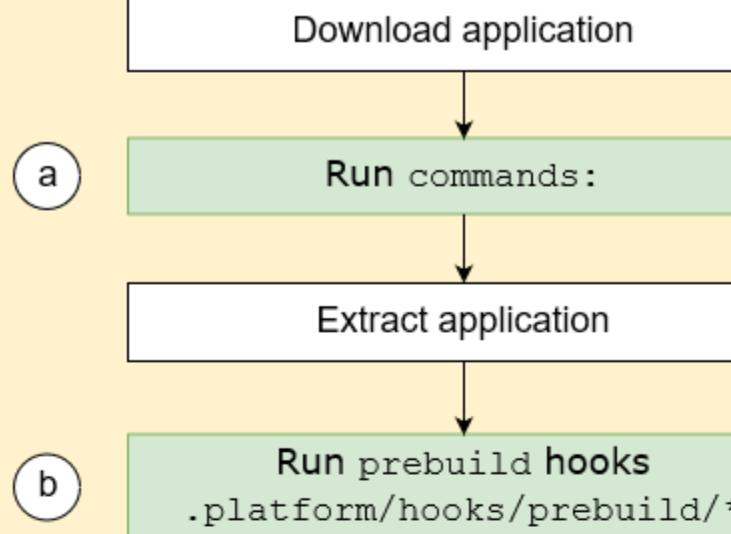
Avec de nombreuses façons d'étendre la plateforme de votre environnement, il est utile de savoir ce qui se passe chaque fois qu'Elastic Beanstalk alloue une instance ou exécute un déploiement sur une instance. Le diagramme suivant illustre l'ensemble du workflow de déploiement. Il décrit les différentes phases d'un déploiement et les étapes suivies par Elastic Beanstalk au cours de chaque phase.

### Notes

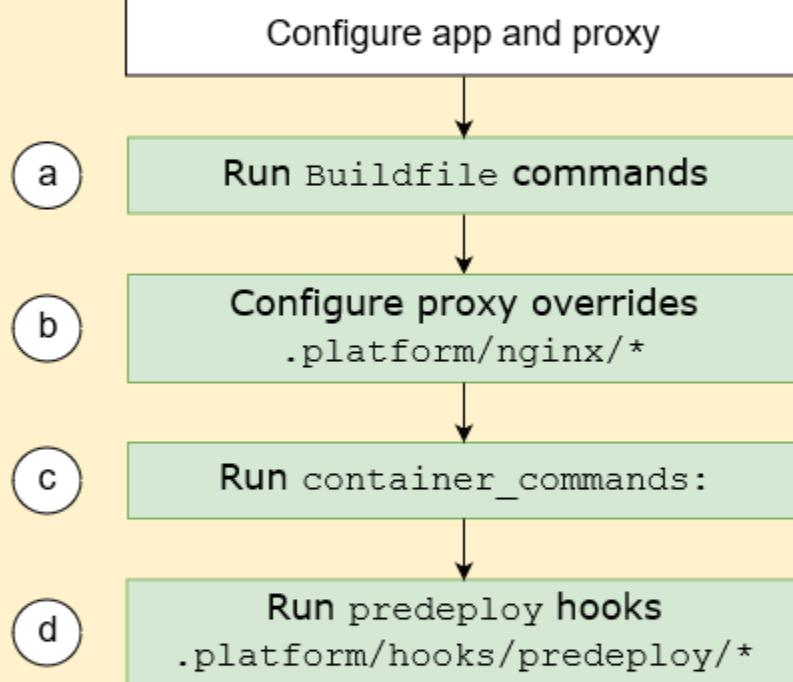
- Le diagramme ne représente pas l'ensemble complet des étapes suivies par Elastic Beanstalk sur les instances d'environnement au cours du déploiement. Nous fournissons ce diagramme à titre d'illustration, pour vous indiquer l'ordre et le contexte de l'exécution de vos personnalisations.
- Par souci de simplicité, le diagramme ne mentionne que les sous-répertoires `hook .platform/*` (pour les déploiements d'applications), et non les sous-répertoires `hook .platform/confighooks/*` (pour les déploiements de configuration). Les hooks dans ces derniers sous-

répertoires s'exécutent exactement au cours des mêmes étapes que les hooks dans les sous-répertoires correspondants indiqués dans le diagramme.

### 1. Initial steps



### 2. Configure



### 3. Deploy

Deploy/flip app and proxy

La liste suivante détaille les phases et les étapes de déploiement.

### 1. Étapes initiales

Elastic Beanstalk télécharge et extrait votre application. Après chacune de ces étapes, Elastic Beanstalk exécute l'une des étapes d'extensibilité.

- a. Exécute les commandes présentes dans la section [commands](#): (p. 746) de tout fichier de configuration.
- b. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/prebuild` de votre bundle source (`.platform/confighooks/prebuild` pour un déploiement de configuration).

### 2. Configuration

Elastic Beanstalk configure votre application et le serveur proxy.

- a. Exécute les commandes trouvées dans le bundle de fichiers source `Buildfile`.
- b. Copie vos fichiers de configuration proxy personnalisés, le cas échéant, du répertoire `.platform/nginx` de votre bundle de fichiers source vers leur emplacement d'exécution.
- c. Exécute les commandes de la section [container\\_commands](#): (p. 748) de tout fichier de configuration.
- d. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/predeploy` de votre bundle source (`.platform/confighooks/predeploy` pour un déploiement de configuration).

### 3. Déploiement

Elastic Beanstalk déploie et exécute votre application et le serveur proxy.

- a. Exécute la commande trouvée dans le fichier `Procfile` de votre bundle de fichiers source.
- b. Exécute ou réexécute le serveur proxy avec vos fichiers de configuration proxy personnalisés, le cas échéant.
- c. Exécute tous les fichiers exécutables trouvés dans le répertoire `.platform/hooks/postdeploy` de votre bundle source (`.platform/confighooks/postdeploy` pour un déploiement de configuration).

## Outils de script de plateforme

Cette rubrique décrit les outils que AWS Elastic Beanstalk fournit sur les instances d'environnement qui utilisent des versions de plateforme Amazon Linux. Vous pouvez utiliser ces outils pour améliorer les scripts de hook de plateforme qui s'exécutent sur instance dans votre environnement.

### get-config

Utilisez l'outil `get-config` pour récupérer les valeurs des variables d'environnement et d'autres informations de plateforme et d'instance. Cet outil est disponible dans `/opt/elasticbeanstalk/bin/get-config`.

#### Commandes get-config

Chaque commande d'outil `get-config` renvoie un type spécifique d'informations. Utilisez la syntaxe suivante pour exécuter l'une des commandes de l'outil.

```
$ /opt/elasticbeanstalk/bin/get-config command [ options ]
```

L'exemple suivant exécute uniquement la commande `environment`.

```
$ /opt/elasticbeanstalk/bin/get-config environment -k PORT
```

Selon la commande et les options que vous choisissez, l'outil renvoie un objet (JSON ou YAML) avec des paires clé-valeur, ou une seule valeur.

Vous pouvez tester get-config en utilisant SSH pour vous connecter à une instance dans votre environnement Elastic Beanstalk.

#### Note

Lorsque vous exécutez get-config pour des tests, certaines commandes peuvent nécessiter des priviléges utilisateur root pour accéder aux informations sous-jacentes. Si vous obtenez une erreur d'autorisation d'accès, exécutez à nouveau la commande sous sudo.

Vous n'avez pas besoin d'ajouter sudo lorsque vous utilisez l'outil dans les scripts que vous déployez dans votre environnement. Elastic Beanstalk exécute tous vos scripts en tant qu'utilisateur racine.

Les sections suivantes décrivent les commandes de l'outil.

### optionsettings – Options de configuration

La commande get-config optionsettings renvoie un objet répertoriant les options de configuration définies sur l'environnement et utilisées par la plateforme sur les instances d'environnement. Elles sont organisées par espace de noms.

```
$ /opt/elasticbeanstalk/bin/get-config optionsettings
{
  "aws:elasticbeanstalk:application:environment": {
    "JDBC_CONNECTION_STRING": "", "aws:elasticbeanstalk:container:tomcat:jvmoptions": {"JVM Options": "", "Xms": "256m", "Xmx": "256m"}, "aws:elasticbeanstalk:environment:proxy": {"ProxyServer": "nginx", "StaticFiles": [""]}, "aws:elasticbeanstalk:healthreporting:system": {"SystemType": "enhanced"}, "aws:elasticbeanstalk:hostmanager": {"LogPublicationControl": "false"}}
}
```

Pour renvoyer une option de configuration spécifique, utilisez l'option --namespace (-n) pour spécifier un espace de noms, et l'option --option-name (-o) pour spécifier un nom d'option.

```
$ /opt/elasticbeanstalk/bin/get-config optionsettings -n aws:elasticbeanstalk:container:php:phpini -o memory_limit
256M
```

### environment – Propriétés de l'environnement

La commande get-config environment renvoie un objet contenant une liste de propriétés d'environnement. Il s'agit à la fois des propriétés configurées par l'utilisateur et des propriétés fournies par Elastic Beanstalk.

```
$ /opt/elasticbeanstalk/bin/get-config environment
{
  "JDBC_CONNECTION_STRING": "", "RDS_PORT": "3306", "RDS_HOSTNAME": "anj9aw1b0tbj6b.cijbpanmxz5u.us-west-2.rds.amazonaws.com", "RDS_USERNAME": "testusername", "RDS_DB_NAME": "ebdb", "RDS_PASSWORD": "testpassword"
}
```

Par exemple, Elastic Beanstalk fournit des propriétés d'environnement pour la connexion à une instance de base de données Amazon RDS intégrée (RDS\_HOSTNAME, etc.) Ces propriétés de connexion RDS apparaissent dans la sortie de get-config environment, mais pas dans la sortie de get-config optionsettings, car elles n'ont pas été définies dans les options de configuration.

Pour renvoyer une propriété d'environnement spécifique, utilisez l'option --key (-k) pour spécifier une propriété clé.

```
$ /opt/elasticbeanstalk/bin/get-config environment -k TESTPROPERTY
```

```
testvalue
```

### container – Valeurs de configuration sur instance

La commande `get-config container` renvoie un objet qui répertorie les valeurs de configuration de la plateforme et de l'environnement telles qu'elles sont reflétées sur les instances d'environnement.

L'exemple suivant illustre la sortie de la commande dans un environnement Amazon Linux 2 Tomcat.

```
$ /opt/elasticbeanstalk/bin/get-config container
{"common_log_list": ["/var/log/eb-engine.log", "/var/log/eb-hooks.log"], "default_log_list": ["/var/log/nginx/access.log", "/var/log/nginx/error.log"], "environment_name": "myenv-1da84946", "instance_port": "80", "log_group_name_prefix": "/aws/elasticbeanstalk", "proxy_server": "nginx", "static_files": [""], "xray_enabled": "false"}
```

Pour renvoyer la valeur d'une clé spécifique, utilisez l'option `--key (-k)` pour spécifier la clé.

```
$ /opt/elasticbeanstalk/bin/get-config container -k environment_name
myenv-1da84946
```

### addons – Valeurs de configuration du module complémentaire

La commande `get-config addons` renvoie un objet contenant des informations de configuration des modules complémentaires d'environnement. Utilisez-la pour récupérer la configuration d'une base de données Amazon RDS associée à l'environnement.

```
$ /opt/elasticbeanstalk/bin/get-config addons
{"rds": {"Description": "RDS Environment variables", "env": {"RDS_DB_NAME": "ebdb", "RDS_HOSTNAME": "ea13k2wimuldh8i.c18mnpu5rwvg.us-east-2.rds.amazonaws.com", "RDS_PASSWORD": "password", "RDS_PORT": "3306", "RDS_USERNAME": "user"}}}
```

Vous pouvez restreindre le résultat de deux façons. Pour récupérer les valeurs d'un module complémentaire spécifique, utilisez l'option `--add-on (-a)` pour spécifier le nom du module complémentaire.

```
$ /opt/elasticbeanstalk/bin/get-config addons -a rds
{"Description": "RDS Environment variables", "env": {"RDS_DB_NAME": "ebdb", "RDS_HOSTNAME": "ea13k2wimuldh8i.c18mnpu5rwvg.us-east-2.rds.amazonaws.com", "RDS_PASSWORD": "password", "RDS_PORT": "3306", "RDS_USERNAME": "user"}}}
```

Pour renvoyer la valeur d'une clé spécifique dans un module complémentaire, ajoutez l'option `--key (-k)` pour spécifier la clé.

```
$ /opt/elasticbeanstalk/bin/get-config addons -a rds -k RDS_DB_NAME
ebdb
```

### platformconfig – Valeurs de configuration constantes

La commande `get-config platformconfig` renvoie un objet contenant des informations de configuration de la plateforme qui sont constantes pour la version de la plateforme. La sortie est la même dans tous les environnements exécutant la même version de plateforme. L'objet de sortie de la commande comporte deux objets incorporés :

- `GeneralConfig` – Contient des informations qui sont constantes sur les dernières versions de toutes les branches de plateforme Amazon Linux 2.
- `PlatformSpecificConfig` – Contient des informations qui sont constantes pour la version de la plateforme et qui lui sont spécifiques.

L'exemple suivant montre la sortie de la commande sur un environnement qui utilise Tomcat 8.5 exécutant la branche plateforme Corretto 11.

```
$ /opt/elasticbeanstalk/bin/get-config platformconfig
{"GeneralConfig":{"AppUser":"webapp","AppDeployDir":"/var/app/current/","AppStagingDir":"/var/app/staging/","ProxyServer":"nginx","DefaultInstancePort":"80"},"PlatformSpecificConfig":{"ApplicationPort":"8080","JavaVersion":"11","TomcatVersion":"8.5"}}
```

Pour renvoyer la valeur d'une clé spécifique, utilisez l'option `--key (-k)` pour spécifier la clé. Ces clés sont uniques sur les deux objets incorporés. Vous n'avez pas besoin de spécifier l'objet qui contient la clé.

```
$ /opt/elasticbeanstalk/bin/get-config platformconfig -k AppStagingDir
/var/app/staging/
```

### Options de sortie get-config

Utilisez l'option `--output` pour spécifier le format de l'objet en sortie. Les valeurs valides sont `JSON` (par défaut) et `YAML`. Il s'agit d'une option globale et vous devez la spécifier avant le nom de la commande.

L'exemple suivant renvoie des valeurs d'option de configuration au format `YAML`.

```
$ /opt/elasticbeanstalk/bin/get-config --output YAML optionsettings
aws:elasticbeanstalk:application:environment:
  JDBC_CONNECTION_STRING: ""
aws:elasticbeanstalk:container:tomcat:jvmoptions:
  JVM Options: ""
  Xms: 256m
  Xmx: 256m
aws:elasticbeanstalk:environment:proxy:
  ProxyServer: nginx
  StaticFiles:
    - ""
aws:elasticbeanstalk:healthreporting:system:
  SystemType: enhanced
aws:elasticbeanstalk:hostmanager:
  LogPublicationControl: "false"
```

### download-source-bundle (AMI Amazon Linux uniquement)

Sur les branches de plateforme de l'AMI Amazon Linux (précédemment Amazon Linux 2), Elastic Beanstalk fournit un outil supplémentaire, `download-source-bundle`. Utilisez-le pour télécharger le code source de votre application pendant le déploiement de votre plateforme. Cet outil est disponible dans `/opt/elasticbeanstalk/bin/download-source-bundle`.

L'exemple de script `00-unzip.sh` se trouve dans le dossier `appdeploy/pre` des instances d'environnement. Il montre l'utilisation de `download-source-bundle` pour télécharger le code source de l'application dans le dossier `/opt/elasticbeanstalk/deploy/appsource` pendant le déploiement.

## Déploiement d'applications Elastic Beanstalk à partir de conteneurs Docker

Elastic Beanstalk prend en charge le déploiement d'applications web à partir de conteneurs Docker. Les conteneurs Docker vous permettent de définir votre propre environnement d'exécution. Vous pouvez également choisir votre propre plateforme, votre langage de programmation et toutes les dépendances d'application (comme les gestionnaires de package ou les outils) qui ne sont généralement pas pris

en charge par d'autres plateformes. Les conteneurs Docker sont indépendants, et incluent toutes les informations de configuration et les logiciels dont votre application web a besoin pour fonctionner. Toutes les variables d'environnement définies dans la console Elastic Beanstalk sont transmises aux conteneurs.

L'utilisation de Docker avec Elastic Beanstalk vous permet de bénéficier d'une infrastructure qui gère les détails de l'allocation de la capacité, l'équilibrage de charge, la mise à l'échelle et la surveillance de l'état de l'application. Vous pouvez facilement gérer votre application web dans un environnement qui prend en charge les différents services intégrés dans Elastic Beanstalk. Ces environnements incluent, mais sans s'y limiter, [VPC](#), [RDS](#) et [IAM](#). Pour obtenir de plus amples informations sur Docker, y compris sur la façon de l'installer, les logiciels nécessaires et la façon d'utiliser les images Docker pour lancer les conteneurs Docker, veuillez accéder au site consacré à [Docker: the Linux container engine](#).

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

## Famille de plateformes Docker

La famille de plateformes Docker pour Elastic Beanstalk comprend plusieurs plateformes. La plateforme Docker qui s'exécute sur Amazon Linux 2 offre le plus d'avantages, par exemple le support à long terme. Les sections qui suivent détaillent les plateformes Docker proposées par Elastic Beanstalk et les chemins de migration recommandés vers Amazon Linux 2.

Pour de plus amples informations sur les versions de plateformes prises en charge pour chaque plateforme Docker, veuillez consulter la page [Plateformes prises en charge](#) dans le document Plateformes AWS Elastic Beanstalk.

### La plateforme Docker

Elastic Beanstalk peut déployer une image Docker et un code source sur des instances EC2 exécutant la plateforme Docker Elastic Beanstalk. La plateforme offre une prise en charge de plusieurs conteneurs (et d'un seul conteneur). Vous pouvez également tirer parti de l'outil Docker Compose sur la plateforme Docker pour simplifier la configuration, les tests et le déploiement de votre application.

Cette plateforme Docker Amazon Linux 2 offre les avantages suivants :

- Support à long terme. La plateforme Docker sur Amazon Linux 2 bénéficie d'un support à long terme, offrant des mises à jour pour la sécurité et les fonctionnalités.
- Fonctionnalités Docker Compose. Cette plateforme vous permettra de tirer parti des fonctionnalités fournies par l'outil Docker Compose pour définir et exécuter plusieurs conteneurs. Vous pouvez inclure le fichier `docker-compose.yml` à déployer sur Elastic Beanstalk.
- Utilisation d'images d'applications provenant de référentiels publics ou privés. Elastic Beanstalk appelle l'interface de ligne de commande Docker Compose. Pour cela, il traite le fichier `docker-compose.yml` pour extraire les images de l'application et les exécuter en tant qu'applications conteneurisées.
- Créez des images de conteneur pendant le déploiement. Vous n'avez pas besoin de pré-générer vos images d'application avant de les déployer pour qu'elles s'exécutent en tant que conteneurs. Pendant le déploiement, vous pouvez générer les images de conteneur à partir de zéro en spécifiant les dépendances dans le `Dockerfile`.

Pour de plus amples informations sur les exemples et pour obtenir de l'aide pour démarrer avec un environnement Docker, veuillez consulter [the section called "La plateforme Docker" \(p. 48\)](#). Pour de plus amples informations sur les formats de définition de conteneur et leur utilisation, veuillez consulter [the section called "Configuration Docker" \(p. 53\)](#).

Les sections suivantes sont pertinentes pour les environnements Docker Elastic Beanstalk qui utilisent la version antérieure de la plateforme AMI Amazon Linux (antérieure à Amazon Linux 2).

## Docker (AMI Amazon Linux)

La plateforme Docker basée sur des AMI Amazon Linux peut être utilisée pour déployer une image Docker (décrise dans une définition Dockerfile ou `Dockerrun.aws.json`) et le code source dans des instances EC2 s'exécutant dans un environnement Elastic Beanstalk. Cette plateforme Docker exécute un seul conteneur pour chaque instance.

Pour obtenir des exemples et de l'aide pour démarrer avec un environnement Docker, veuillez consulter [the section called "La plateforme Docker" \(p. 48\)](#). Pour de plus amples informations sur les formats de définition de conteneur et leur utilisation, veuillez consulter [the section called "Configuration Docker" \(p. 53\)](#).

## Docker multiconteneurs (AMI Amazon Linux)

### Note

Cette plateforme prend uniquement en charge le système d'exploitation AMI Amazon Linux (la version qui précède Amazon Linux 2). La plateforme [Docker \(p. 48\)](#) fournit la fonctionnalité Docker multiconteneurs avec Amazon Linux 2.

L'autre plateforme générique, Docker multiconteneurs, utilise Amazon Elastic Container Service (Amazon ECS) pour coordonner un déploiement de plusieurs conteneurs Docker dans un cluster Amazon ECS d'un environnement Elastic Beanstalk. Chacune des instances de l'environnement exécute le même ensemble de conteneurs, qui sont définis dans un fichier `Dockerrun.aws.json`. Si votre environnement Elastic Beanstalk utilise une version de plateforme AMI Amazon Linux AMI (antérieure à Amazon Linux 2), utilisez la plateforme multiconteneurs pour déployer plusieurs conteneurs Docker sur chaque instance.

Pour de plus amples informations sur la plateforme Docker multiconteneurs et son utilisation, veuillez consulter [Utilisation de la plateforme Docker multiconteneurs \(AMI Amazon Linux\) \(p. 64\)](#). La rubrique [Configuration Docker multi-conteneurs \(p. 69\)](#) contient des informations détaillées sur la version 2 du format `Dockerrun.aws.json`, qui est similaire à la version utilisée avec la plateforme Docker, mais qui n'est pas compatible avec cette version. Il existe également un [didacticiel \(p. 73\)](#) qui vous guide tout au long du déploiement d'un environnement multiconteneurs à partir de zéro. L'environnement décrit exécute un site web PHP avec un proxy NGINX qui s'exécute devant lui dans un conteneur séparé.

## Conteneurs Docker préconfigurés

Outre les deux plateformes Docker génériques, il existe plusieurs branches de plateforme Docker préconfigurées, que vous pouvez utiliser pour exécuter votre application dans l'une des piles de logiciels couramment utilisées, telles que Java avec Glassfish ou Python avec uWSGI. Utilisez un conteneur préconfiguré s'il correspond au logiciel utilisé par votre application.

### Note

Toutes les branches de la plateforme Docker préconfigurée utilisent le système d'exploitation AMI Amazon Linux (version antérieure à Amazon Linux 2). Pour migrer votre application GlassFish vers Amazon Linux 2, utilisez la plateforme Docker générique et déployez GlassFish et votre code d'application vers une image Docker Amazon Linux 2. Pour de plus amples informations à ce sujet, veuillez consulter [the section called "Tutorial - GlassFish sur Docker : chemin vers Amazon Linux 2" \(p. 61\)](#).

Pour plus d'informations, consultez [Conteneurs Docker préconfigurés \(p. 82\)](#).

## Utilisation de la plateforme Docker

### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de

plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [the section called “Mise à niveau vers Amazon Linux 2” \(p. 508\)](#).

AWS Elastic Beanstalk peut lancer des environnements Docker en créant une image décrite dans un `Dockerfile` ou en extrayant une image Docker à distance. Si vous déployez une image Docker à distance, vous n'avez pas besoin d'inclure un `Dockerfile`. Au lieu de cela, si vous utilisez également Docker Compose, utilisez un fichier `docker-compose.yml`, qui spécifie une image à utiliser et des options de configuration supplémentaires. Si vous n'utilisez pas Docker Compose avec vos environnements Docker, utilisez plutôt un fichier `Dockerrun.aws.json`.

#### Rubriques

- [Prerequisites \(p. 49\)](#)
- [Mise en conteneur d'une application Elastic Beanstalk \(p. 49\)](#)
- [Test d'un conteneur en local \(p. 50\)](#)
- [Déploiement d'un conteneur avec un Dockerfile \(p. 51\)](#)
- [Test d'une image Docker distante \(p. 51\)](#)
- [Déploiement d'une image Docker distante sur Elastic Beanstalk \(p. 53\)](#)
- [Nettoyage \(p. 53\)](#)
- [Configuration Docker \(p. 53\)](#)
- [Déploiement d'une application GlassFish sur la plateforme Docker : un chemin de migration vers Amazon Linux 2 \(p. 61\)](#)

## Prerequisites

Ce tutoriel suppose que vous ayez quelques connaissances des opérations Elastic Beanstalk de base, de l'interface de ligne de commande Elastic Beanstalk (EB CLI) et de Docker. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk. Ce tutoriel utilise [l'interface de ligne de commande \(CLI\) EB \(p. 1027\)](#), mais vous pouvez également créer des environnements et télécharger des applications à l'aide de la console Elastic Beanstalk.

Pour suivre ce tutoriel, vous aurez également besoin des composants Docker suivants :

- Une installation locale en état de fonctionnement de Docker. Pour de plus amples informations, veuillez consulter [Get Docker](#) (Obtenir Docker) sur le site web de documentation de Docker.
- Accès à Docker Hub. Vous devez créer un ID Docker pour accéder au Docker Hub. Pour de plus amples informations, veuillez consulter [Share the application](#) (Partager l'application) sur le site Web de documentation de Docker.

Pour en savoir plus sur la configuration des environnements Docker sur des plateformes Elastic Beanstalk, consultez [Configuration Docker \(p. 53\)](#) dans ce même chapitre.

## Mise en conteneur d'une application Elastic Beanstalk

Pour cet exemple, nous allons créer une image Docker de l'exemple d'application Flask généré à la section [Déploiement d'une application Flask sur Elastic Beanstalk \(p. 365\)](#). L'application se compose d'un fichier principal, `application.py`. Nous avons également besoin d'un `Dockerfile`. Placez les deux fichiers à la racine d'un répertoire.

```
~/eb-docker-flask/  
|-- Dockerfile
```

```
|-- application.py
```

#### Example ~/eb-docker-flask/application.py

```
from flask import Flask

# Print a nice greeting
def say_hello(username = "World"):
    return '<p>Hello %s!</p>\n' % username

# Some bits of text for the page
header_text = '''
<html>\n<head> <title>EB Flask Test</title> </head>\n<body>'''

instructions = '''
<p><em>Hint</em>: This is a RESTful web service! Append a username
to the URL (for example: <code>/Thelonious</code>) to say hello to
someone specific.</p>\n'''

home_link = '<p><a href="/">Back</a></p>\n'
footer_text = '</body>\n</html>'

# Elastic Beanstalk looks for an 'application' that is callable by default
application = Flask(__name__)

# Add a rule for the index page
application.add_url_rule('/', 'index', (lambda: header_text +
    say_hello() + instructions + footer_text))

# Add a rule when the page is accessed with a name appended to the site
# URL
application.add_url_rule('/<username>', 'hello', (lambda username:
    header_text + say_hello(username) + home_link + footer_text))

# Run the application
if __name__ == "__main__":
    # Setting debug to True enables debug output. This line should be
    # removed before deploying a production application.
    application.debug = True
    application.run(host="0.0.0.0")
```

#### Example ~/eb-docker-flask/Dockerfile

```
FROM python:3.6
COPY . /app
WORKDIR /app
RUN pip install Flask==1.0.2
EXPOSE 5000
CMD ["python", "application.py"]
```

## Test d'un conteneur en local

Utilisez l'interface de ligne de commande Elastic Beanstalk (EB) pour configurer votre référentiel local pour le déploiement dans Elastic Beanstalk. Définissez le Dockerfile de votre application à la racine du répertoire.

```
~/eb-docker-flask$ eb init -p docker application-name
```

(Facultatif) Utilisez la commande eb local run pour créer et exécuter votre conteneur en local.

```
~/eb-docker-flask$ eb local run --port 5000
```

### Note

Pour de plus amples informations sur la commande eb local, veuillez consulter [the section called “eb local” \(p. 1097\)](#). La commande n'est pas prise en charge sur Windows. Vous pouvez également créer et exécuter votre conteneur avec les commandes docker build et docker run. Pour plus d'informations, consultez la [documentation Docker](#).

(Facultatif) Lorsque votre conteneur est en cours d'exécution, utilisez la commande eb local open pour afficher votre application dans un navigateur web. Vous pouvez également ouvrir <http://localhost:5000/> dans un navigateur web.

```
~/eb-docker-flask$ eb local open
```

## Déploiement d'un conteneur avec un Dockerfile

Après avoir testé votre application localement, déployez-la dans un environnement Elastic Beanstalk. Elastic Beanstalk utilise les instructions de votre fichier **Dockerfile** pour construire et exécuter l'image.

Utilisez la commande eb create pour créer un environnement et déployer votre application.

```
~/eb-docker-flask$ eb create environment-name
```

Après le lancement de votre environnement, utilisez la commande eb open pour l'afficher dans un navigateur web.

```
~/eb-docker-flask$ eb open
```

## Test d'une image Docker distante

Nous allons ensuite créer une image Docker de l'application Flask générée à partir de la section précédente et la transmettre à Docker Hub.

### Notes

- Les étapes suivantes créent une image Docker disponible publiquement.
- Vous utiliserez les commandes Docker de votre installation Docker locale, ainsi que vos informations d'identification Docker Hub. Pour en savoir plus, consultez la section [Prerequisites \(p. 49\)](#) précédente.

Une fois que nous avons créé et transmis notre image, nous pouvons la déployer dans Elastic Beanstalk avec un fichier `docker-compose.yml`, si vous utilisez Docker Compose avec votre environnement Docker. Si vous n'utilisez pas Docker Compose avec votre environnement Docker, utilisez un fichier `Dockerrun.aws.json`. Pour créer une image Docker de l'application Flask et la transmettre à Docker Hub, exécutez les commandes suivantes. Nous utiliserons le même répertoire que celui de l'exemple précédent, mais vous pouvez utiliser n'importe quel répertoire avec le code de votre application. Saisissez votre ID Docker pour `docker-id` pour vous connecter à Docker Hub.

```
~/eb-docker-flask$ docker build -t docker-id/beanstalk-flask:latest .
~/eb-docker-flask$ docker push docker-id/beanstalk-flask:latest
```

### Note

Avant de transmettre votre image, il se peut que vous deviez exécuter docker login. Vous serez invité à saisir vos informations d'identification Docker Hub si vous exéutez la commande sans paramètres.

Si vous utilisez l'outil Docker Compose pour gérer votre environnement Docker, vous pouvez maintenant déployer votre application en utilisant uniquement un fichier `docker-compose.yml`. Pour en savoir plus sur les fichiers `docker-compose.yml`, veuillez consulter [Configuration Docker \(p. 53\)](#).

Si vous n'utilisez pas Docker Compose, utilisez un fichier `Dockerrun.aws.json`. Pour plus d'informations, consultez [Déploiement à l'aide de `Dockerrun.aws.json` v1 \(sans Docker Compose\) \(p. 52\)](#).

Créez un répertoire et un fichier `docker-compose.yml`.

Example `~/remote-docker/docker-compose.yml`

```
version: '3.8'
services:
  beanstalk-flask:
    image: "username/beanstalk-flask"
    ports:
      - "80:5000"
```

[Déploiement à l'aide de `Dockerrun.aws.json` v1 \(sans Docker Compose\)](#)

Si vous n'utilisez pas l'outil Docker Compose pour gérer votre environnement Docker, vous pouvez maintenant déployer votre application en utilisant uniquement un fichier `Dockerrun.aws.json`. Pour en savoir plus sur les fichiers `Dockerrun.aws.json`, veuillez consulter [Configuration des plateformes Docker \(sans Docker Compose\) \(p. 57\)](#).

Créez un répertoire et un fichier `Dockerrun.aws.json`.

Example `~/remote-docker/Dockerrun.aws.json`

```
{
  "AWSEBDockerrunVersion": "1",
  "Image": {
    "Name": "username/beanstalk-flask",
    "Update": "true"
  },
  "Ports": [
    {
      "ContainerPort": "5000"
    }
  ]
}
```

Utilisez l'interface de ligne de commande EB pour configurer votre référentiel local pour le déploiement dans Elastic Beanstalk.

```
~/remote-docker$ eb init -p docker application-name
```

(Facultatif) Utilisez `eb local run` pour créer et exécuter votre conteneur en local. Pour de plus amples informations sur la commande `eb local`, veuillez consulter [eb local \(p. 1097\)](#).

```
~/remote-docker$ eb local run --port 5000
```

(Facultatif) Lorsque votre conteneur est en cours d'exécution, utilisez la commande `eb local open` pour afficher votre application dans un navigateur web. Vous pouvez également ouvrir <http://localhost:5000> dans un navigateur web.

```
~/remote-docker$ eb local open
```

## Déploiement d'une image Docker distante sur Elastic Beanstalk

Après avoir testé votre conteneur localement, déployez-le dans un environnement Elastic Beanstalk. Elastic Beanstalk utilise le fichier `docker-compose.yml` pour extraire et exécuter votre image si vous utilisez Docker Compose. Sinon, Elastic Beanstalk utilise le fichier `Dockerrun.aws.json`.

Utilisez l'interface de ligne de commande EB pour créer un environnement et déployer votre image.

```
~/remote-docker$ eb create environment-name
```

Une fois que votre environnement est lancé, utilisez `eb open` pour l'afficher dans un navigateur web.

```
~/remote-docker$ eb open
```

## Nettoyage

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 538\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Ou, avec l'interface de ligne de commande EB :

```
~/remote-docker$ eb terminate environment-name
```

## Configuration Docker

Cette section décrit comment préparer votre image et votre conteneur Docker pour les déployer dans Elastic Beanstalk.

### Environnement Docker avec Docker Compose

Cette section décrit comment préparer votre image et votre conteneur Docker pour les déployer dans Elastic Beanstalk. Toute application web que vous déployez sur Elastic Beanstalk dans un environnement Docker doit inclure un fichier `docker-compose.yml` si vous utilisez également l'outil Docker Compose.

Vous pouvez déployer votre application web en tant que service conteneurisé sur Elastic Beanstalk en effectuant l'une des actions suivantes :

- Créez un fichier `docker-compose.yml` afin de déployer une image Docker à partir d'un référentiel hébergé dans Elastic Beanstalk. Aucun autre fichier n'est requis si tous vos déploiements proviennent d'images dans des référentiels publics. (Si votre déploiement doit obtenir une image à partir d'un référentiel privé, vous devez inclure des fichiers de configuration supplémentaires pour l'authentification. Pour plus d'informations, consultez [Utilisation d'images provenant d'un référentiel privé \(p. 54\)](#).) Pour de plus amples informations sur le fichier `docker-compose.yml`, veuillez consulter [Compose file reference](#) sur le site web Docker.
- Créez un fichier `Dockerfile` pour permettre à Elastic Beanstalk de créer et d'exécuter une image personnalisée. Ce fichier est facultatif. Cela dépend des besoins de votre déploiement. Pour de plus amples informations sur le fichier `Dockerfile`, veuillez consulter [Dockerfile reference](#) sur le site web Docker.
- Créez un fichier `.zip` contenant vos fichiers d'application, toutes les dépendances du fichier d'application, le `Dockerfile` et le fichier `docker-compose.yml`. Si vous utilisez l'interface de ligne de commande EB pour déployer votre application, elle crée un fichier `.zip` automatiquement. Les deux fichiers doivent être à la racine, ou au niveau supérieur, de l'archive `.zip`.

Si vous utilisez uniquement un fichier `docker-compose.yml` pour déployer votre application, vous n'avez pas besoin de créer un fichier `.zip`.

Cette rubrique est une référence de syntaxe. Pour obtenir des procédures détaillées sur le lancement d'environnements Docker à l'aide d'Elastic Beanstalk, veuillez consulter [Utilisation de la plateforme Docker \(p. 48\)](#).

Pour en savoir plus sur Docker Compose et savoir comment l'installer, veuillez consulter les sites Docker [Overview of Docker Compose](#) et [Install Docker Compose](#).

#### Note

Si vous n'utilisez pas Docker Compose pour configurer vos environnements Docker, vous ne devez pas non plus utiliser le fichier `docker-compose.yml`. Au lieu de cela, utilisez le fichier `Dockerrun.aws.json`, le fichier `Dockerfile` ou les deux.

Pour plus d'informations, consultez [the section called "Configuration des plateformes Docker \(sans Docker Compose\)" \(p. 57\)](#).

## Utilisation d'images à partir d'un référentiel privé

Elastic Beanstalk doit s'authentifier auprès du registre en ligne qui héberge le référentiel privé avant de pouvoir extraire et déployer vos images à partir d'un référentiel privé. Vous disposez de deux options pour stocker et récupérer les informations d'identification pour que votre environnement Elastic Beanstalk s'authentifie auprès d'un référentiel.

- Le stockage de paramètres AWS Systems Manager (SSM)
- le fichier `Dockerrun.aws.json v3` ;

### Utilisation du stockage de paramètres AWS Systems Manager (SSM)

Vous pouvez configurer Elastic Beanstalk pour qu'il se connecte à votre référentiel privé avant de commencer le processus de déploiement. Cela permet à Elastic Beanstalk d'accéder aux images à partir du référentiel et de les déployer dans votre environnement Elastic Beanstalk.

Cette configuration initie des événements dans la phase de préénération du processus de déploiement Elastic Beanstalk. La configuration a lieu dans le répertoire de configuration [.ebextensions \(p. 737\)](#). La configuration utilise des scripts de [hook de plateforme \(p. 35\)](#) qui appellent docker login à s'authentifier

auprès du registre en ligne qui héberge le référentiel privé. Voici une répartition détaillée de ces étapes de configuration.

Pour configurer Elastic Beanstalk afin qu'il s'authentifie auprès de votre référentiel privé avec AWS SSM

#### Note

Vous devez configurer AWS Systems Manager pour effectuer ces étapes. Pour de plus amples informations, veuillez consulter le [Guide de l'utilisateur AWS Systems Manager](#)

1. Créez votre structure de répertoire `.ebextensions` comme suit.

```
### .ebextensions
#   ### env.config
### .platform
#   ### confighooks
#   #     ### prebuild
#   #       ### 01login.sh
#   ### hooks
#     ### prebuild
#       ### 01login.sh
### docker-compose.yml
```

2. Utilisez le stockage de paramètres [AWS Systems Manager](#) pour enregistrer des informations d'identification de votre référentiel privé afin qu'Elastic Beanstalk puisse récupérer vos informations d'identification si nécessaire. Pour cela, exécutez la commande [put-parameter](#).

```
aws ssm put-parameter --name USER --type String --value "username"
aws ssm put-parameter --name PASSWD --type String --value "passwd"
```

3. Créez le fichier `env.config` suivant et placez-le dans le répertoire `.ebextensions` comme indiqué dans la structure de répertoire précédente.

#### Note

Dans le script, `USER` et `PASSWD` doivent correspondre aux mêmes chaînes que celles utilisées dans les commandes `ssm put-parameter` précédentes.

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    USER: '{{resolve:ssm:USER:1}}'
    PASSWD: '{{resolve:ssm:PASSWD:1}}'
```

4. Créez le fichier de script `01login.sh` suivant et placez-le dans les répertoires suivants (comme illustré également dans la structure de répertoires précédente) :

- `.platform/confighooks/prebuild`
- `.platform/hooks/prebuild`

```
### example 01login.sh
#!/bin/bash
USER=/opt/elasticbeanstalk/bin/get-config environment -k USER
PASSWD=/opt/elasticbeanstalk/bin/get-config environment -k PASSWD
docker login -u $USER -p $PASSWD
```

Le script `01login.sh` appelle d'abord l'outil `get-config` pour récupérer les informations d'identification du référentiel, puis appelle `docker login` pour s'authentifier auprès du référentiel.

## Notes

- Tous les fichiers de script doivent disposer d'une autorisation d'exécution. Utilisez la commande chmod +x pour définir l'autorisation d'exécution sur vos fichiers exécutables.
- Les fichiers exécutables peuvent être des fichiers binaires ou des fichiers script commençant par une ligne #! et contenant leur chemin d'interpréteur, par exemple #!/bin/bash.
- Pour de plus amples informations, veuillez consulter [the section called “Hooks de plateforme” \(p. 35\)](#) dans Extension des plateformes Linux Elastic Beanstalk.

Une fois qu'Elastic Beanstalk peut s'authentifier auprès du registre en ligne qui héberge le référentiel privé, vos images peuvent être déployées et extraites.

### Utilisation du fichier `Dockerrun.aws.json v3`

Cette section décrit une autre approche pour authentifier Elastic Beanstalk auprès d'un référentiel privé. Avec cette approche, vous générez un fichier d'authentification avec la commande Docker, puis téléchargez le fichier d'authentification dans un compartiment Amazon S3. Vous devez également inclure les informations du compartiment dans votre fichier `Dockerrun.aws.json v3`.

Pour générer un fichier d'authentification et le fournir à Elastic Beanstalk

1. Générez un fichier d'authentification avec la commande docker login. Pour les référentiels sur Docker Hub, exécutez docker login:

```
$ docker login
```

Pour d'autres registres, incluez l'URL du serveur de registre :

```
$ docker login registry-server-url
```

#### Note

Si votre environnement Elastic Beanstalk utilise la version de plateforme Docker AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations dans [the section called “Configuration Docker sur l'AMI Amazon Linux \(antérieure à Amazon Linux 2\)” \(p. 92\)](#).

Pour plus d'informations sur le fichier d'authentification, consultez [Store images on Docker Hub et docker login](#) sur le site web de Docker.

2. Chargez une copie du fichier d'authentification nommé `.dockercfg` dans un compartiment Amazon S3 sécurisé.
  - Le compartiment Amazon S3 doit être hébergé dans la même Région AWS que l'environnement qui l'utilise. Elastic Beanstalk ne peut pas télécharger de fichiers à partir d'un compartiment Amazon S3 hébergé dans d'autres régions.
  - Accordez des autorisations pour l'opération `s3:GetObject` au rôle IAM dans le profil d'instance. Pour plus d'informations, consultez [Gestion des profils d'instance Elastic Beanstalk \(p. 92\)](#).
3. Incluez les informations sur le compartiment Amazon S3 dans le paramètre `Authentication` de votre fichier `Dockerrun.aws.json v3`.

Voici un exemple de fichier `Dockerrun.aws.json v3`.

```
{  
  "AWSEBDockerrunVersion": "3",  
  "Authentication": {
```

```
        "bucket": "DOC-EXAMPLE-BUCKET",
        "key": "mydockercfg"
    }
}
```

#### Note

Le paramètre `AWSEBDockerrunVersion` indique la version du fichier `Dockerrun.aws.json`.

- La plateforme Docker Amazon Linux 2 utilise le fichier `Dockerrun.aws.json v3` pour les environnements qui utilisent Docker Compose. Elle utilise le fichier `Dockerrun.aws.json v1` pour les environnements qui n'utilisent pas Docker Compose.
- La plateforme AMI Docker Amazon Linux 2 Docker multiconteneurs utilise le fichier `Dockerrun.aws.json v2`.

Une fois qu'Elastic Beanstalk peut s'authentifier auprès du registre en ligne qui héberge le référentiel privé, vos images peuvent être déployées et extraites.

## Création d'images personnalisées avec un Dockerfile

Vous devez créer un `Dockerfile` si vous ne disposez d'aucune image existante hébergée dans un référentiel.

L'extrait suivant représente un exemple de `Dockerfile`. Si vous suivez les instructions de la page [Utilisation de la plateforme Docker \(p. 48\)](#), vous pouvez charger ce fichier `Dockerfile` comme indiqué. Elastic Beanstalk exécute le jeu 2048 lorsque vous utilisez ce fichier `Dockerfile`.

```
FROM ubuntu:12.04

RUN apt-get update
RUN apt-get install -y nginx zip curl

RUN echo "daemon off;" >> /etc/nginx/nginx.conf
RUN curl -o /usr/share/nginx/www/master.zip -L https://codeload.github.com/gabrielecirulli/2048/zip/master
RUN cd /usr/share/nginx/www/ && unzip master.zip && mv 2048-master/* . && rm -rf 2048-master.zip

EXPOSE 80

CMD ["/usr/sbin/nginx", "-c", "/etc/nginx/nginx.conf"]
```

Pour plus d'informations sur les instructions que vous pouvez inclure dans le `Dockerfile`, consultez la référence sur [Dockerfile](#) sur le site web de Docker.

## Configuration des plateformes Docker (sans Docker Compose)

Si votre environnement Docker Elastic Beanstalk n'utilise pas Docker Compose, lisez les informations supplémentaires des sections suivantes.

### Configuration de la plateforme Docker sans Docker Compose

Toute application web que vous déployez dans Elastic Beanstalk dans un environnement Docker doit inclure un fichier `Dockerfile` ou `Dockerrun.aws.json`. Pour déployer votre application web à partir d'un conteneur Docker dans Elastic Beanstalk, effectuez l'une des actions suivantes :

- Créez un fichier `Dockerfile` pour permettre à Elastic Beanstalk de créer et d'exécuter une image personnalisée.

- Créez un fichier `Dockerrun.aws.json` afin de déployer une image Docker à partir d'un référentiel hébergé dans Elastic Beanstalk.
- Créez un fichier `.zip` contenant vos fichiers d'application, toutes les dépendances du fichier d'application, le `Dockerfile` et le fichier `Dockerrun.aws.json`. Si vous utilisez l'interface de ligne de commande EB pour déployer votre application, elle crée un fichier `.zip` automatiquement.

Si vous utilisez uniquement un `Dockerfile` ou un fichier `Dockerrun.aws.json` pour déployer votre application, vous n'avez pas besoin de créer un fichier `.zip`.

Cette rubrique est une référence de syntaxe. Pour obtenir des procédures détaillées sur le lancement d'environnements Docker, veuillez consulter [Utilisation de la plateforme Docker \(p. 48\)](#).

#### [Dockerrun.aws.json v1](#)

Un fichier `Dockerrun.aws.json` décrit comment déployer une image Docker distante en tant qu'application Elastic Beanstalk. Ce fichier JSON est spécifique à Elastic Beanstalk. Si votre application s'exécute sur une image qui est disponible dans un référentiel hébergé, vous pouvez spécifier l'image dans un fichier `Dockerrun.aws.json v1` et omettre le `Dockerfile`.

Les clés et valeurs valides pour le fichier `Dockerrun.aws.json v1` incluent les opérations suivantes.

##### AWSEBDockerrunVersion

(Obligatoire) Indique la valeur `1` comme numéro de version pour les environnements Docker à conteneur unique.

##### Authentification

(Obligatoire uniquement pour les référentiels privés) Indique l'objet Amazon S3 qui stocke le fichier `.dockercfg`.

Voir [Utilisation d'images à partir d'un référentiel privé \(p. 60\)](#).

##### Image

Spécifie l'image de base Docker sur un référentiel Docker existant, à partir de laquelle vous créez un conteneur Docker. Spécifiez la valeur de la clé `Name` au format `<organisation>/<nom de l'image>` pour les images sur Docker Hub ou `<site>/nom de l'organisation>/<nom de l'image>` pour les autres sites.

Lorsque vous spécifiez une image dans le fichier `Dockerrun.aws.json`, chaque instance de votre environnement Elastic Beanstalk exécute `docker pull` pour exécuter l'image. Vous pouvez également inclure la clé `Update`. La valeur par défaut est `true` et demande à Elastic Beanstalk de vérifier le référentiel, d'extraire les mises à jour de l'image et de remplacer toutes les images mises en cache.

Lorsque vous utilisez un fichier `Dockerfile`, ne spécifiez pas la clé `Image` dans le fichier `Dockerrun.aws.json`. Elastic Beanstalk crée et utilise toujours l'image décrite dans le fichier `Dockerfile`, lorsqu'il existe.

##### Ports

(Obligatoire si vous spécifiez la clé `Image`) Répertorie les ports à exposer sur le conteneur Docker. Elastic Beanstalk utilise la valeur `ContainerPort` pour connecter le conteneur Docker au proxy inverse exécuté sur l'hôte.

Vous pouvez spécifier plusieurs ports de conteneur, mais Elastic Beanstalk utilise uniquement le premier port. Il utilise ce port pour connecter votre conteneur au proxy inverse de l'hôte et acheminer les demandes depuis le réseau Internet public. Si vous utilisez un `Dockerfile`, la première valeur `ContainerPort` doit correspondre à la première entrée de la liste `EXPOSE` du `Dockerfile`.

Vous pouvez également spécifier une liste de ports dans HostPort. Les entrées HostPort spécifient les ports d'hôte avec lesquelles les valeurs ContainerPort sont mappées. Si vous ne spécifiez pas de valeur HostPort, la valeur ContainerPort est utilisée par défaut.

```
{  
    "Image": {  
        "Name": "image-name"  
    },  
    "Ports": [  
        {  
            "ContainerPort": 8080,  
            "HostPort": 8000  
        }  
    ]  
}
```

## Volumes

Mappez les volumes d'une instance EC2 à votre conteneur Docker. Spécifiez un ou plusieurs groupes de volumes à mapper.

```
{  
    "Volumes": [  
        {  
            "HostDirectory": "/path/inside/host",  
            "ContainerDirectory": "/path/inside/container"  
        }  
    ]  
    ...  
}
```

## Journalisation

Spécifiez le répertoire du conteneur dans lequel votre application écrit les journaux. Elastic Beanstalk télécharge tous les journaux de ce répertoire sur Amazon S3 lorsque vous demandez des journaux de processus ou de groupe. Si vous effectuez une rotation des journaux dans un dossier nommé rotated au sein de ce répertoire, vous pouvez également configurer Elastic Beanstalk afin de charger les journaux concernés dans Amazon S3 pour un stockage permanent. Pour plus d'informations, consultez [Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk \(p. 884\)](#).

## Commande

Spécifiez une commande à exécuter dans le conteneur. Si vous spécifiez un point d'entrée, la valeur Command (Commande) est ajoutée comme argument dans Entrypoint (Point d'entrée). Pour plus d'informations, consultez [CMD](#) dans la documentation Docker.

## Entrypoint

Spécifiez une commande par défaut à exécuter lorsque le conteneur démarre. Pour plus d'informations, consultez [ENTRYPOINT](#) dans la documentation Docker.

L'extrait suivant est un exemple illustrant la syntaxe du fichier `Dockerrun.aws.json` pour un conteneur unique.

```
{  
    "AWSEBDockerrunVersion": "1",  
    "Image": {  
        "Name": "janedoe/image",  
    }  
}
```

```
        "Update": "true"
    },
    "Ports": [
        {
            "ContainerPort": "1234"
        }
    ],
    "Volumes": [
        {
            "HostDirectory": "/var/app/mydb",
            "ContainerDirectory": "/etc/mysql"
        }
    ],
    "Logging": "/var/log/nginx",
    "Entrypoint": "/app/bin/myapp",
    "Command": "--argument"
}
```

Vous pouvez fournir à Elastic Beanstalk le fichier `Dockerrun.aws.json` uniquement, ou une archive `.zip` contenant les fichiers `Dockerrun.aws.json` et `Dockerfile`. Si vous fournissez les deux fichiers, le `Dockerfile` décrit l'image Docker et le fichier `Dockerrun.aws.json` fournit des informations supplémentaires pour le déploiement, comme indiqué ultérieurement dans cette section.

#### Note

Les deux fichiers doivent être à la racine, ou au niveau supérieur, de l'archive `.zip`. Ne créez pas l'archive à partir d'un répertoire contenant les fichiers. Au lieu de cela, parcourez ce répertoire et créez l'archive dans le répertoire.

Si vous fournissez les deux fichiers, ne spécifiez aucune image dans le fichier `Dockerrun.aws.json`. Elastic Beanstalk crée et utilise l'image décrite dans le fichier `Dockerfile` et ignore celle spécifiée dans le fichier `Dockerrun.aws.json`.

#### Utilisation d'images à partir d'un référentiel privé

Ajoutez les informations relatives au compartiment Amazon S3 contenant le fichier d'authentification dans le paramètre `Authentication` du fichier `Dockerrun.aws.json v1`. Assurez-vous que le paramètre `Authentication` contient une clé et un compartiment Amazon S3 valides. Le compartiment Amazon S3 doit être hébergé dans la même Région AWS que l'environnement qui l'utilise. Elastic Beanstalk ne télécharge pas de fichiers à partir de compartiments Amazon S3 hébergés dans d'autres régions.

Pour plus d'informations sur la génération et le chargement du fichier d'authentification, consultez [Utilisation d'images à partir d'un référentiel privé \(p. 90\)](#).

L'exemple suivant décrit comment utiliser un fichier d'authentification nommé `mydockercfg` dans un compartiment nommé `DOC-EXAMPLE-BUCKET` afin d'utiliser une image privée dans un registre tiers.

```
{
    "AWSEBDockerrunVersion": "1",
    "Authentication": {
        "Bucket": "DOC-EXAMPLE-BUCKET",
        "Key": "mydockercfg"
    },
    "Image": {
        "Name": "quay.io/johndoe/private-image",
        "Update": "true"
    },
    "Ports": [
        {
            "ContainerPort": "1234"
        }
    ],
    "Volumes": [
```

```
{  
    "HostDirectory": "/var/app/mydb",  
    "ContainerDirectory": "/etc/mysql"  
},  
    "Logging": "/var/log/nginx"  
}
```

## Déploiement d'une application GlassFish sur la plateforme Docker : un chemin de migration vers Amazon Linux 2

Le but de ce tutoriel est de fournir aux clients qui utilisent la plateforme Docker GlassFish préconfigurée (basée sur l'AMI Amazon Linux) un chemin de migration vers Amazon Linux 2. Vous pouvez migrer votre application GlassFish vers Amazon Linux 2 en déployant GlassFish et votre code d'application sur une image Docker Amazon Linux 2.

Le tutoriel vous guide à travers l'utilisation de la plateforme AWS Elastic Beanstalk Docker pour déployer une application basée sur le [serveur d'applications Java EE GlassFish](#) dans un environnement Elastic Beanstalk.

Nous illustrons deux approches de création d'une image Docker :

- Simple – Fournissez le code source de votre application GlassFish et laissez Elastic Beanstalk créer et exécuter une image Docker dans le cadre de la mise en service de votre environnement. Ceci est facile à configurer, au prix d'une augmentation du temps de mise en service des instances.
- Avancé – Créez une image Docker personnalisée contenant votre code d'application et vos dépendances, et fournissez-la à Elastic Beanstalk afin de l'utiliser dans votre environnement. Cette approche est légèrement plus impliquée et réduit le temps de mise en service des instances dans votre environnement.

### Prerequisites

Ce tutoriel suppose que vous ayez quelques connaissances des opérations Elastic Beanstalk de base, de l'interface de ligne de commande Elastic Beanstalk (EB CLI) et de Docker. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk. Ce tutoriel utilise [l'interface de ligne de commande \(CLI\) EB \(p. 1027\)](#), mais vous pouvez également créer des environnements et télécharger des applications à l'aide de la console Elastic Beanstalk.

Pour suivre ce tutoriel, vous aurez également besoin des composants Docker suivants :

- Une installation locale en état de fonctionnement de Docker. Pour de plus amples informations, veuillez consulter [Get Docker](#) (Obtenir Docker) sur le site web de documentation de Docker.
- Accès à Docker Hub. Vous devez créer un ID Docker pour accéder au Docker Hub. Pour de plus amples informations, veuillez consulter [Share the application](#) (Partager l'application) sur le site Web de documentation de Docker.

Pour en savoir plus sur la configuration des environnements Docker sur des plateformes Elastic Beanstalk, consultez [Configuration Docker \(p. 53\)](#) dans ce même chapitre.

### Exemple simple : fournissez votre code d'application

C'est un moyen facile de déployer votre application GlassFish. Vous fournissez au code source de votre application le fichier `Dockerfile` inclus dans ce tutoriel. Elastic Beanstalk crée une image Docker qui inclut votre application et la pile logicielle GlassFish. Ensuite, Elastic Beanstalk exécute l'image sur les instances de votre environnement.

Cette approche présente un problème, à savoir qu'Elastic Beanstalk crée l'image Docker localement chaque fois qu'il crée une instance pour votre environnement. La génération de l'image augmente le temps de mise en service de l'instance. Cet impact n'est pas limité à la création initiale de l'environnement. Il se produit également lors des actions de montée en charge.

Pour lancer un environnement avec un exemple d'application GlassFish

1. Téléchargez l'exemple docker-glassfish-al2-v1.zip, puis développez le fichier .zip dans un répertoire de votre environnement de développement.

```
~$ curl https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/samples/docker-glassfish-al2-v1.zip --output docker-glassfish-al2-v1.zip
~$ mkdir glassfish-example
~$ cd glassfish-example
~/glassfish-example$ unzip ../docker-glassfish-al2-v1.zip
```

La structure de votre répertoire doit être la suivante.

```
~/glassfish-example
|-- Dockerfile
|-- Dockerrun.aws.json
|-- glassfish-start.sh
|-- index.jsp
|-- META-INF
|   |-- LICENSE.txt
|   |-- MANIFEST.MF
|   `-- NOTICE.txt
|-- robots.txt
`-- WEB-INF
    `-- web.xml
```

Les fichiers suivants sont essentiels à la création et à l'exécution d'un conteneur Docker dans votre environnement :

- Dockerfile – Fournit des instructions que Docker utilise pour créer une image avec votre application et les dépendances requises.
  - glassfish-start.sh – Script shell exécuté par l'image Docker pour démarrer votre application.
  - Dockerrun.aws.json – Fournit une clé de journalisation pour inclure le journal du serveur d'applications GlassFish dans les [demandes de fichiers journaux \(p. 884\)](#). Si vous n'êtes pas intéressé par les journaux GlassFish, vous pouvez omettre ce fichier.
2. Configurez votre répertoire local pour le déploiement sur Elastic Beanstalk.

```
~/glassfish-example$ eb init -p docker glassfish-example
```

3. (Facultatif) Utilisez la commande eb local run pour créer et exécuter votre conteneur en local.

```
~/glassfish-example$ eb local run --port 8080
```

#### Note

Pour de plus amples informations sur la commande eb local, veuillez consulter [the section called "eb local" \(p. 1097\)](#). La commande n'est pas prise en charge sur Windows. Vous pouvez également créer et exécuter votre conteneur avec les commandes docker build et docker run. Pour plus d'informations, consultez la [documentation Docker](#).

4. (Facultatif) Lorsque votre conteneur est en cours d'exécution, utilisez la commande eb local open pour afficher votre application dans un navigateur web. Vous pouvez également ouvrir <http://localhost:8080/> dans un navigateur web.

```
~/glassfish-example$ eb local open
```

5. Utilisez la commande eb create pour créer un environnement et déployer votre application.

```
~/glassfish-example$ eb create glassfish-example-env
```

6. Après le lancement de votre environnement, utilisez la commande eb open pour l'afficher dans un navigateur web.

```
~/glassfish-example$ eb open
```

Lorsque vous avez terminé l'exemple, arrêtez l'environnement et supprimez les ressources associées.

```
~/glassfish-example$ eb terminate --all
```

## Exemple avancé : fournir une image Docker préconstruite

Il s'agit d'un moyen plus avancé de déployer votre application GlassFish. En vous appuyant sur le premier exemple, vous créez une image Docker contenant votre code d'application et la pile logicielle GlassFish, puis vous la poussez vers Docker Hub. Après avoir effectué cette étape unique, vous pouvez lancer des environnements Elastic Beanstalk basés sur votre image personnalisée.

Lorsque vous lancez un environnement et que vous fournissez votre image Docker, les instances de votre environnement téléchargent et utilisent cette image directement et n'ont pas besoin de créer une image Docker. Par conséquent, le temps de mise en service de l'instance est réduit.

### Notes

- Les étapes suivantes créent une image Docker disponible publiquement.
- Vous utiliserez les commandes Docker de votre installation Docker locale, ainsi que vos informations d'identification Docker Hub. Pour en savoir plus, consultez la section [Prerequisites \(p. 49\)](#) précédente.

Pour lancer un environnement avec une image Docker d'application GlassFish préconstruite

1. Téléchargez et développez l'exemple `docker-glassfish-al2-v1.zip` comme dans l'[exemple simple \(p. 61\)](#) précédent. Si vous avez terminé cet exemple, vous pouvez utiliser le répertoire que vous possédez déjà.
2. Créez une image Docker et poussez-la vers Docker Hub. Saisissez votre ID Docker pour `docker-id` pour vous connecter à Docker Hub.

```
~/glassfish-example$ docker build -t docker-id/beanstalk-glassfish-example:latest .
~/glassfish-example$ docker push docker-id/beanstalk-glassfish-example:latest
```

### Note

Avant de transmettre votre image, il se peut que vous deviez exécuter `docker login`. Vous serez invité à saisir vos informations d'identification Docker Hub si vous exécutez la commande sans paramètres.

3. Créez un répertoire supplémentaire.

```
~$ mkdir glassfish-prebuilt
```

```
~$ cd glassfish-prebuilt
```

4. Copiez l'exemple suivant dans un fichier nommé `Dockerrun.aws.json`.

Example `~/glassfish-prebuilt/Dockerrun.aws.json`

```
{  
    "AWSEBDockerrunVersion": "1",  
    "Image": {  
        "Name": "docker-username/beanstalk-glassfish-example"  
    },  
    "Ports": [  
        {  
            "ContainerPort": 8080,  
            "HostPort": 8080  
        }  
    ],  
    "Logging": "/usr/local/glassfish5/glassfish/domains/domain1/logs"  
}
```

5. Configurez votre répertoire local pour le déploiement sur Elastic Beanstalk.

```
~/glassfish-prebuilt$ eb init -p docker glassfish-prebuilt
```

6. (Facultatif) Utilisez la commande `eb local run` pour exécuter votre conteneur localement.

```
~/glassfish-prebuilt$ eb local run --port 8080
```

7. (Facultatif) Lorsque votre conteneur est en cours d'exécution, utilisez la commande `eb local open` pour afficher votre application dans un navigateur web. Vous pouvez également ouvrir <http://localhost:8080/> dans un navigateur web.

```
~/glassfish-prebuilt$ eb local open
```

8. Utilisez la commande `eb create` pour créer un environnement et déployer votre image Docker.

```
~/glassfish-prebuilt$ eb create glassfish-prebuilt-env
```

9. Après le lancement de votre environnement, utilisez la commande `eb open` pour l'afficher dans un navigateur web.

```
~/glassfish-prebuilt$ eb open
```

Lorsque vous avez terminé l'exemple, arrêtez l'environnement et supprimez les ressources associées.

```
~/glassfish-prebuilt$ eb terminate --all
```

## Utilisation de la plateforme Docker multiconteneurs (AMI Amazon Linux)

### Note

Cette plateforme prend uniquement en charge le système d'exploitation AMI Amazon Linux (antérieur à Amazon Linux 2). La fonctionnalité Docker multiconteneurs sur Amazon Linux 2 est fournie par la plateforme [Docker \(p. 48\)](#) et bénéficie d'un support à long terme.

Si votre environnement Docker Elastic Beanstalk utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), vous pouvez créer des environnements Docker qui prennent en charge plusieurs conteneurs par instance Amazon EC2 via la plateforme Docker multiconteneurs pour Elastic Beanstalk.

Elastic Beanstalk utilise Amazon Elastic Container Service (Amazon ECS) pour coordonner les déploiements de conteneurs vers des environnements Docker multiconteneurs. Amazon ECS fournit des outils permettant de gérer un cluster d'instances exécutant des conteneurs Docker. Elastic Beanstalk prend en charge les tâches Amazon ECS, y compris la création de clusters, la définition et l'exécution des tâches. Chacune des instances de l'environnement exécute le même ensemble de conteneurs, qui sont définis dans un fichier `Dockerrun.aws.json`.

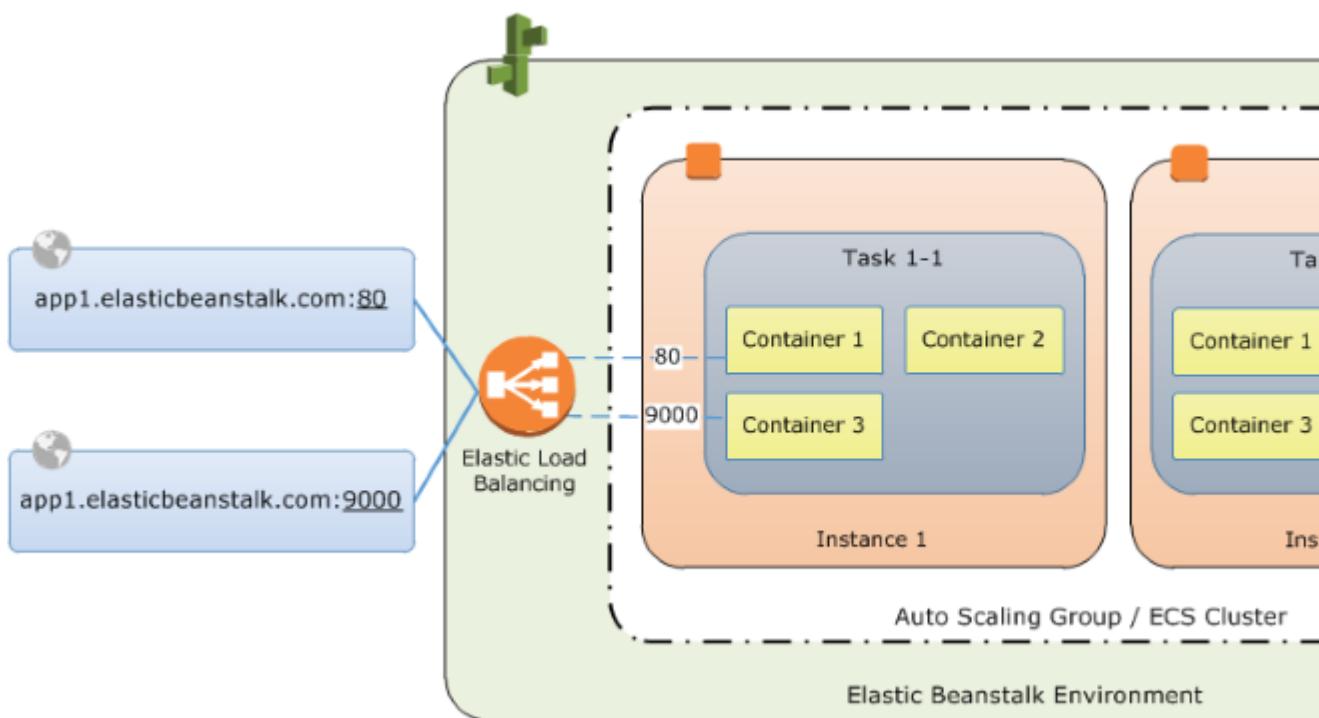
#### Rubriques

- [Plateforme Docker multi-conteneurs \(p. 65\)](#)
- [Dockerrun.aws.json file \(p. 66\)](#)
- [Images Docker \(p. 66\)](#)
- [Rôle de l'instance de conteneur \(p. 67\)](#)
- [Ressources Amazon ECS créées par Elastic Beanstalk \(p. 67\)](#)
- [Utilisation de plusieurs écouteurs Elastic Load Balancing \(p. 68\)](#)
- [Échec de déploiements de conteneurs \(p. 69\)](#)
- [Configuration Docker multi-conteneurs \(p. 69\)](#)
- [Environnements Docker multiconteneurs avec la console Elastic Beanstalk \(p. 73\)](#)
- [Migration vers la plateforme Docker Amazon Linux 2 \(p. 79\)](#)

## Plateforme Docker multi-conteneurs

Sur Elastic Beanstalk, les plateformes Docker préconfigurées et génériques standard ne prennent en charge qu'un seul conteneur Docker par environnement Elastic Beanstalk. Afin de tirer pleinement parti de Docker, Elastic Beanstalk vous permet de créer un environnement dans lequel vos instances Amazon EC2 exécutent plusieurs conteneurs Docker côté à côté.

Le schéma suivant présente un exemple d'environnement Elastic Beanstalk configuré avec trois conteneurs Docker exécutés sur chaque instance Amazon EC2 d'un groupe Auto Scaling :



## Dockerrun.aws.json file

Instances de conteneur sont des instances Amazon EC2 exécutant un Docker multiconteneurs dans un environnement Elastic Beanstalk. Elles requièrent un fichier de configuration nommé `Dockerrun.aws.json`. Ce fichier est propre à Elastic Beanstalk. Il peut être utilisé seul ou combiné au code source et au contenu d'un [bundle de fichiers source \(p. 415\)](#) afin de créer un environnement sur une plateforme Docker.

### Note

La version 1 du format `Dockerrun.aws.json` permet de lancer un seul conteneur Docker dans un environnement Elastic Beanstalk. La version 2 ajoute la prise en charge de plusieurs conteneurs par instance Amazon EC2 et ne peut être utilisée qu'avec la plateforme Docker multiconteneurs. Le format varie considérablement par rapport à la version précédente, qui est détaillée sur la page [Configuration Docker \(p. 53\)](#).

Pour obtenir des détails sur le format mis à jour ainsi qu'un exemple de fichier, veuillez consulter [Dockerrun.aws.json v2 \(p. 69\)](#).

## Images Docker

La plateforme Docker multiconteneurs pour Elastic Beanstalk nécessite que des images soient préconstruites et stockées dans un référentiel d'images en ligne public ou privé.

### Note

La création d'images personnalisées pendant le déploiement via un fichier `Dockerfile` n'est pas prise en charge par la plateforme Docker multiconteneurs sur Elastic Beanstalk. Créez vos images et déployez-les dans un référentiel en ligne avant de créer un environnement Elastic Beanstalk.

Spécifiez des images par nom dans `Dockerrun.aws.json`. Notez ces conventions :

- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).

- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu`).

Pour configurer Elastic Beanstalk afin qu'il s'authentifie auprès d'un référentiel privé, incluez le paramètre `authentication` dans votre fichier `Dockerrun.aws.json`.

## Rôle de l'instance de conteneur

Elastic Beanstalk utilise une AMI optimisée pour Elastic Beanstalk avec un agent de conteneur Amazon ECS qui s'exécute dans un conteneur Docker. L'agent communique avec Amazon ECS pour coordonner les déploiements de conteneur. Afin de communiquer avec Amazon ECS, chaque instance Amazon EC2 doit disposer des autorisations correspondantes dans IAM. Ces autorisations sont associées au [profil d'instance \(p. 21\)](#) par défaut lorsque vous créez un environnement dans la console de gestion Elastic Beanstalk :

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "ECSAccess",  
            "Effect": "Allow",  
            "Action": [  
                "ecs:Poll",  
                "ecs:StartTask",  
                "ecs:StopTask",  
                "ecs:DiscoverPollEndpoint",  
                "ecs:StartTelemetrySession",  
                "ecs:RegisterContainerInstance",  
                "ecs:DeregisterContainerInstance",  
                "ecs:DescribeContainerInstances",  
                "ecs:Submit*"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

Si vous créez votre propre profil d'instance, vous pouvez associer la stratégie gérée `AWSElasticBeanstalkMulticontainerDocker` pour faire en sorte que les autorisations restent à jour. Pour obtenir des instructions sur la création de stratégies et de rôles dans IAM, veuillez consulter [Création de rôles IAM](#) dans le Guide de l'utilisateur IAM.

## Ressources Amazon ECS créées par Elastic Beanstalk

Lorsque vous créez un environnement via la plateforme Docker multiconteneurs, Elastic Beanstalk crée et configure automatiquement plusieurs ressources Amazon Elastic Container Service tout en créant l'environnement afin de créer les conteneurs nécessaires sur chaque instance Amazon EC2.

- Cluster Amazon ECS – Dans Amazon ECS, les instances de conteneur sont organisées en clusters. En cas d'utilisation avec Elastic Beanstalk, un cluster est toujours créé pour chaque environnement Docker multiconteneurs.
- Définition de tâche Amazon ECS – Elastic Beanstalk utilise le fichier `Dockerrun.aws.json` dans votre projet pour générer la définition de tâche Amazon ECS qui est utilisée pour configurer les instances de conteneur dans l'environnement.
- Tâche Amazon ECS – Elastic Beanstalk communique avec Amazon ECS pour exécuter une tâche sur chaque instance de l'environnement afin de coordonner le déploiement de conteneur. Dans un

environnement évolutif, Elastic Beanstalk lance une nouvelle tâche chaque fois qu'une instance est ajoutée au cluster. Dans de rares cas, vous devrez peut-être accroître la quantité d'espace réservée aux conteneurs et aux images. Pour en savoir plus, reportez-vous à la section [Configuration des environnements Docker \(p. 85\)](#).

- Agent de conteneur Amazon ECS – L'agent s'exécute dans un conteneur Docker sur les instances de votre environnement. L'agent interroge le service Amazon ECS et attend l'exécution d'une tâche.
- Volumes de données Amazon ECS – Pour faciliter la collecte des journaux, Elastic Beanstalk insère les définitions des volumes (en plus des volumes que vous définissez dans `Dockerrun.aws.json`) dans la définition de tâche.

Elastic Beanstalk crée des volumes de fichiers journaux sur l'instance de conteneur (un pour chaque conteneur), à l'emplacement `/var/log/containers/contianername`. Ces volumes sont nommés `awseb-logs-contianername` et sont fournis pour être montés par les conteneurs. Pour de plus amples informations sur la façon de les monter, veuillez consulter [Format des définitions de conteneur \(p. 72\)](#).

## Utilisation de plusieurs écouteurs Elastic Load Balancing

Vous pouvez configurer plusieurs écouteurs Elastic Load Balancing dans un environnement Docker multiconteneurs afin de prendre en charge le trafic entrant pour les proxys ou autres services qui ne fonctionnent pas sur le port HTTP par défaut.

Créez un dossier `.ebextensions` dans votre bundle de fichiers source et ajoutez un fichier avec une extension `.config`. L'exemple suivant présente un fichier de configuration qui crée un écouteur Elastic Load Balancing sur le port 8080.

### `.ebextensions/elb-listener.config`

```
option_settings:  
  aws:elb:listener:8080:  
    ListenerProtocol: HTTP  
    InstanceProtocol: HTTP  
    InstancePort: 8080
```

Si votre environnement s'exécute dans un [Amazon Virtual Private Cloud](#) (Amazon VPC) personnalisé que vous avez créé, Elastic Beanstalk s'occupe du reste. Dans un VPC par défaut, vous devez configurer le groupe de sécurité de votre instance pour autoriser le trafic entrant provenant de l'équilibreur de charge. Ajoutez un deuxième fichier de configuration, qui ajoute une règle de trafic entrant au groupe de sécurité :

### `.ebextensions/elb-ingress.config`

```
Resources:  
  port8080SecurityGroupIngress:  
    Type: AWS::EC2::SecurityGroupIngress  
    Properties:  
      GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}  
      IpProtocol: tcp  
      ToPort: 8080  
      FromPort: 8080  
      SourceSecurityGroupName: {"Fn::GetAtt": ["AWSEBLoadBalancer",  
      "SourceSecurityGroup.GroupName"] }
```

Pour de plus amples informations sur le format du fichier de configuration, veuillez consulter [Ajout et personnalisation des ressources de l'environnement Elastic Beanstalk \(p. 759\)](#) et [Paramètres d'option \(p. 738\)](#).

En plus d'ajouter un écouteur à la configuration Elastic Load Balancing et d'ouvrir un port dans le groupe de sécurité, vous devez mapper le port sur l'instance de l'hôte à un port sur le conteneur Docker, dans la

section `containerDefinitions` du fichier `Dockerrun.aws.json`. L'extrait suivant en présente un exemple:

```
"portMappings": [  
    {  
        "hostPort": 8080,  
        "containerPort": 8080  
    }  
]
```

Pour de plus amples informations sur le format du fichier `Dockerrun.aws.json`, veuillez consulter [Dockerrun.aws.json v2 \(p. 69\)](#).

## Échec de déploiements de conteneurs

En cas d'échec d'une tâche Amazon ECS, un ou plusieurs conteneurs de votre environnement Elastic Beanstalk ne démarreront pas. Elastic Beanstalk ne restaure pas les environnements multiconteneurs suite à l'échec d'une tâche Amazon ECS. Si le démarrage d'un conteneur échoue dans votre environnement, redéployez la version actuelle ou une version de travail précédente à partir de la console Elastic Beanstalk.

Pour déployer une version existante

1. Ouvrez la console Elastic Beanstalk dans la région de votre environnement.
2. Cliquez sur Actions à droite du nom de votre application, puis cliquez sur View application versions (Afficher les versions de l'application).
3. Sélectionnez une version de votre application, puis cliquez sur Déploiement.

## Configuration Docker multi-conteneurs

Un fichier `Dockerrun.aws.json` est un fichier JSON propre à Elastic Beanstalk qui décrit comment déployer un ensemble de conteneurs Docker sous la forme d'une application Elastic Beanstalk. Vous pouvez utiliser un fichier `Dockerrun.aws.json` pour un environnement Docker multi-conteneurs.

Le fichier `Dockerrun.aws.json` décrit les conteneurs à déployer dans chaque instance de conteneur (instance Amazon EC2 qui héberge les conteneurs Docker) de l'environnement, ainsi que les volumes de données à créer sur l'instance de l'hôte, qui seront montés par les conteneurs.

Un fichier `Dockerrun.aws.json` peut être utilisé de façon autonome ou compressé avec un code source supplémentaire dans une archive unique. Le code source archivé dans un fichier `Dockerrun.aws.json` est déployé dans les instances de conteneur Amazon EC2 et accessible dans le répertoire `/var/app/current/`. Utilisez la section `volumes` du fichier de configuration pour fournir les volumes de fichier pour les conteneurs Docker s'exécutant sur l'instance hôte. Utilisez la section `mountPoints` des définitions de conteneur intégré pour mapper ces volumes aux points de montage que les applications des conteneurs Docker peuvent utiliser.

### Rubriques

- [Dockerrun.aws.json v2 \(p. 69\)](#)
- [Utilisation d'images à partir d'un référentiel privé \(p. 71\)](#)
- [Format des définitions de conteneur \(p. 72\)](#)

## Dockerrun.aws.json v2

Le fichier `Dockerrun.aws.json` comprend trois sections :

#### AWSEBDockerrunVersion

Indique la valeur 2 comme numéro de version pour les environnements Docker multiconteneurs.  
containerDefinitions

Tableau des définitions de conteneur, détaillées ci-dessous.  
volumes

Permet de créer des volumes à partir de dossiers de l'instance de conteneur Amazon EC2 ou à partir du bundle de fichiers source (déployé sur /var/app/current). Montez ces volumes sur les chemins de vos conteneurs Docker à l'aide des mountPoints de la [définition de conteneur \(p. 72\)](#).

##### Note

Elastic Beanstalk configure des volumes supplémentaires pour les journaux (un pour chaque conteneur). Ils doivent être montés par vos conteneurs Docker afin d'écrire les journaux dans l'instance de l'hôte. Consultez [Format des définitions de conteneur \(p. 72\)](#) pour plus de détails.

Les volumes sont spécifiés dans le format suivant :

```
"volumes": [
  {
    "name": "volumename",
    "host": {
      "sourcePath": "/path/on/host/instance"
    }
  }
],
```

#### authentification

(facultatif) Dans Amazon S3, emplacement d'un fichier .dockercfg contenant les données d'authentification pour un référentiel privé. Utilise le format suivant :

```
"authentication": {
  "bucket": "DOC-EXAMPLE-BUCKET",
  "key": "mydockercfg"
},
```

Consultez [Utilisation d'images à partir d'un référentiel privé \(p. 71\)](#) pour plus de détails.

L'extrait suivant est un exemple illustrant la syntaxe du fichier Dockerrun.aws.json pour une instance avec deux conteneurs.

```
{
  "AWSEBDockerrunVersion": 2,
  "volumes": [
    {
      "name": "php-app",
      "host": {
        "sourcePath": "/var/app/current/php-app"
      }
    },
    {
      "name": "nginx-proxy-conf",
      "host": {
        "sourcePath": "/var/app/current/proxy/conf.d"
      }
    }
  ],
}
```

```
"containerDefinitions": [
  {
    "name": "php-app",
    "image": "php:fpm",
    "environment": [
      {
        "name": "Container",
        "value": "PHP"
      }
    ],
    "essential": true,
    "memory": 128,
    "mountPoints": [
      {
        "sourceVolume": "php-app",
        "containerPath": "/var/www/html",
        "readOnly": true
      }
    ]
  },
  {
    "name": "nginx-proxy",
    "image": "nginx",
    "essential": true,
    "memory": 128,
    "portMappings": [
      {
        "hostPort": 80,
        "containerPort": 80
      }
    ],
    "links": [
      "php-app"
    ],
    "mountPoints": [
      {
        "sourceVolume": "php-app",
        "containerPath": "/var/www/html",
        "readOnly": true
      },
      {
        "sourceVolume": "nginx-proxy-conf",
        "containerPath": "/etc/nginx/conf.d",
        "readOnly": true
      },
      {
        "sourceVolume": "awseb-logs-nginx-proxy",
        "containerPath": "/var/log/nginx"
      }
    ]
  }
]
```

## Utilisation d'images à partir d'un référentiel privé

Ajoutez les informations relatives au compartiment Amazon S3 contenant le fichier d'authentification dans le paramètre `authentication` du fichier `Dockerrun.aws.json`. Assurez-vous que le paramètre `authentication` contient une clé et un compartiment Amazon S3 valides. Le compartiment Amazon S3 doit être hébergé dans la même région que l'environnement qui l'utilise. Elastic Beanstalk ne télécharge pas de fichiers à partir de compartiments Amazon S3 hébergés dans d'autres régions.

Pour plus d'informations sur la génération et le chargement du fichier d'authentification, consultez [Utilisation d'images à partir d'un référentiel privé \(p. 90\)](#).

## Format des définitions de conteneur

Les sections des volumes et des définitions de conteneur du fichier `Dockerrun.aws.json` utilisent le même format que les sections correspondantes d'un fichier de définition de tâche Amazon ECS.

Les exemples suivants illustrent un sous-ensemble de paramètres fréquemment utilisés. D'autres paramètres facultatifs sont disponibles. Pour en savoir plus sur le format de définition de tâche et obtenir la liste complète des paramètres de définition de tâche, veuillez consulter [Définitions des tâches Amazon ECS](#) dans le Manuel du développeur Amazon Elastic Container Service.

Un fichier `Dockerrun.aws.json` contient un tableau d'un ou plusieurs objets de définition de conteneur comportant les champs suivants :

nom

Nom du conteneur. Consultez [Paramètres de définition de conteneur standards](#) pour obtenir plus d'informations sur la longueur maximale et les caractères autorisés.

image

Nom d'une image Docker dans un référentiel Docker en ligne à partir de laquelle vous créez un conteneur Docker. Notez ces conventions :

- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu`).

environment

Tableau des variables d'environnement à transmettre au conteneur.

Par exemple, l'entrée suivante définit une variable d'environnement avec le nom `Container` et la valeur `PHP` :

```
"environment": [  
    {  
        "name": "Container",  
        "value": "PHP"  
    }  
,
```

essential

True si la tâche doit s'arrêter en cas d'échec du conteneur. Les conteneurs non essentiels peuvent s'arrêter ou se bloquer sans conséquences sur les autres conteneurs de l'instance.

memory

Quantité de mémoire sur l'instance de conteneur à réserver pour le conteneur. Spécifiez un entier non nul pour l'un des paramètres `memory` ou `memoryReservation` (ou les deux) dans les définitions de conteneur.

memoryReservation

La limite flexible (en MiB) de mémoire à réservé pour le conteneur. Spécifiez un entier non nul pour l'un des paramètres `memory` ou `memoryReservation` (ou les deux) dans les définitions de conteneur.

mountPoints

Volumes à monter à partir de l'instance de conteneur Amazon EC2 et emplacement dans lequel ils doivent être montés dans le système de fichiers du conteneur Docker. Lorsque vous montez

des volumes qui contiennent des contenus applicatifs, votre conteneur peut lire les données que vous chargez dans votre bundle source. Lorsque vous montez des volumes de journal pour écrire des données de journaux, Elastic Beanstalk peut collecter les données de journaux à partir de ces volumes.

Elastic Beanstalk crée les volumes de fichiers journaux sur l'instance de conteneur (un pour chaque conteneur Docker), à l'emplacement `/var/log/containers/containernname`. Ces volumes sont nommés `awseb-logs-containernname` et doivent être montés à l'emplacement où les journaux sont écrits dans le système de fichiers du conteneur.

Par exemple, le point de montage suivant mappe l'emplacement du journal nginx dans le conteneur sur le volume généré par Elastic Beanstalk pour le conteneur `nginx-proxy`.

```
{  
    "sourceVolume": "awseb-logs-nginx-proxy",  
    "containerPath": "/var/log/nginx"  
}
```

#### portMappings

Mappe les ports réseau du conteneur sur les ports de l'hôte.

#### links

Liste des conteneurs à lier les uns aux autres. Les conteneurs liés peuvent s'identifier entre eux et communiquer en toute sécurité.

#### volumesFrom

Montez tous les volumes à partir d'un autre conteneur. Par exemple, pour monter les volumes à partir d'un conteneur nommé `web`:

```
"volumesFrom": [  
    {  
        "sourceContainer": "web"  
    }  
,
```

## Environnements Docker multiconteneurs avec la console Elastic Beanstalk

Vous pouvez lancer un cluster d'instances multiconteneurs dans un environnement Elastic Beanstalk à une seule instance ou évolutif, à l'aide de la console Elastic Beanstalk. Ce didacticiel détaille la configuration des conteneurs et la préparation du code source pour un environnement qui utilise deux conteneurs.

Les conteneurs, une application PHP et un proxy nginx, exécutés côté à côté sur chacune des instances Amazon Elastic Compute Cloud (Amazon EC2) dans un environnement Elastic Beanstalk. Après avoir créé l'environnement et vérifié que les applications sont en cours d'exécution, vous allez vous connecter à une instance de conteneur pour voir comment tout fonctionne ensemble.

#### Sections

- [Définition de conteneurs Docker \(p. 74\)](#)
- [Ajout de contenu \(p. 75\)](#)
- [Déploiement sur Elastic Beanstalk \(p. 76\)](#)
- [Connexion à une instance de conteneur \(p. 77\)](#)
- [Mise à jour de l'agent du conteneur Amazon ECS \(p. 78\)](#)

## Définition de conteneurs Docker

La première étape de la création d'un environnement Docker consiste à créer un répertoire pour vos données d'application. Ce dossier peut être situé n'importe où sur votre ordinateur local et porter le nom de votre choix. En plus d'un fichier de configuration de conteneur, ce dossier inclut le contenu que vous téléchargez sur Elastic Beanstalk et que vous déployez dans votre environnement.

### Note

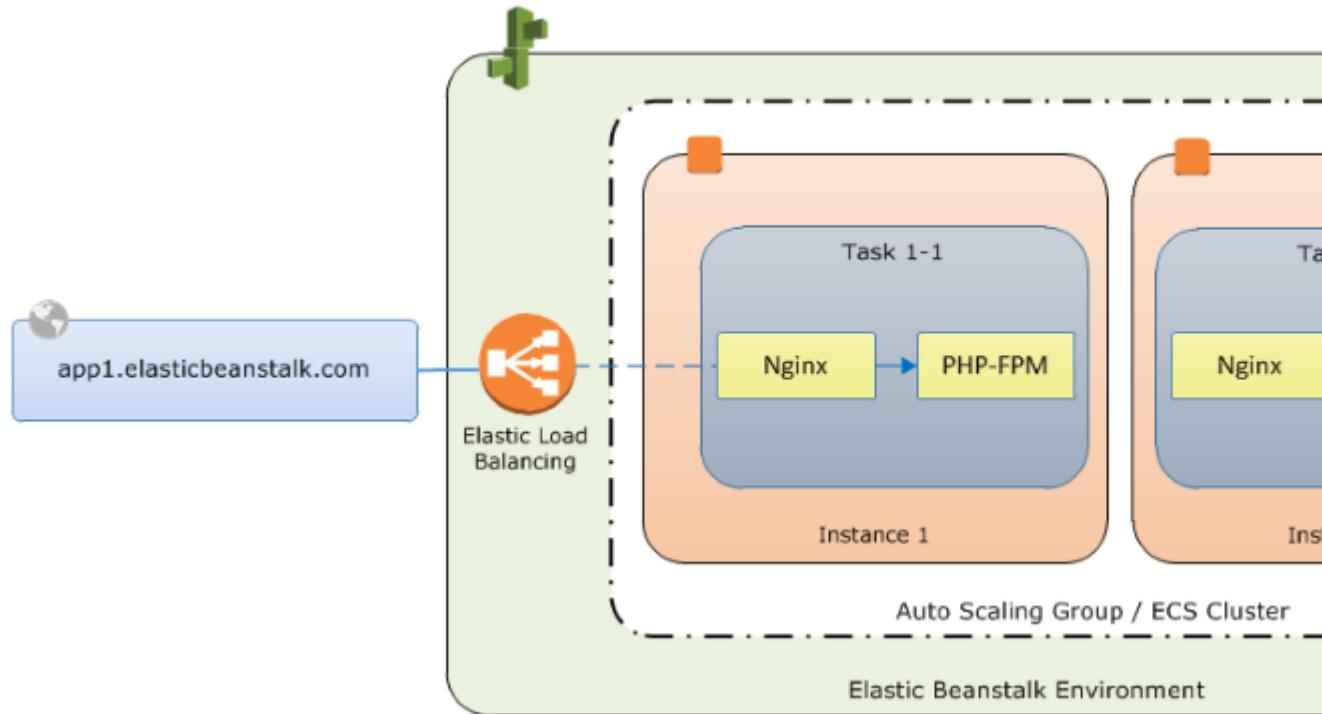
Tout le code pour ce didacticiel est disponible dans le référentiel awslabs sur GitHub à l'emplacement suivant : <https://github.com/awslabs/eb-docker-nginx-proxy>.

Le fichier utilisé par Elastic Beanstalk pour configurer les conteneurs sur une instance Amazon EC2 est un fichier texte au format JSON nommé `Dockerrun.aws.json`. Créez un fichier texte portant ce nom à la racine de votre application et ajoutez le texte suivant :

```
{
  "AWSEBDockerrunVersion": 2,
  "volumes": [
    {
      "name": "php-app",
      "host": {
        "sourcePath": "/var/app/current/php-app"
      }
    },
    {
      "name": "nginx-proxy-conf",
      "host": {
        "sourcePath": "/var/app/current/proxy/conf.d"
      }
    }
  ],
  "containerDefinitions": [
    {
      "name": "php-app",
      "image": "php:fpm",
      "essential": true,
      "memory": 128,
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html",
          "readOnly": true
        }
      ]
    },
    {
      "name": "nginx-proxy",
      "image": "nginx",
      "essential": true,
      "memory": 128,
      "portMappings": [
        {
          "hostPort": 80,
          "containerPort": 80
        }
      ],
      "links": [
        "php-app"
      ],
      "mountPoints": [
        {
          "sourceVolume": "php-app",
          "containerPath": "/var/www/html"
        }
      ]
    }
  ]
}
```

```
        "containerPath": "/var/www/html",
        "readOnly": true
    },
    {
        "sourceVolume": "nginx-proxy-conf",
        "containerPath": "/etc/nginx/conf.d",
        "readOnly": true
    },
    {
        "sourceVolume": "awseb-logs-nginx-proxy",
        "containerPath": "/var/log/nginx"
    }
]
}
}
```

Cet exemple de configuration définit deux conteneurs, un site web PHP avec un proxy nginx devant. Ces deux conteneurs s'exécutent côté à côté dans des conteneurs Docker sur chaque instance dans votre environnement Elastic Beanstalk, accédant à du contenu partagé (le contenu du site web) à partir de volumes sur l'instance hôte, qui sont aussi définis dans ce fichier. Les conteneurs eux-mêmes sont créés à partir d'images hébergées dans des référentiels officiels sur Docker Hub. Vous obtenez alors un environnement similaire au suivant :



Les volumes définis dans la configuration correspondent au contenu que vous allez créer ensuite et télécharger dans le cadre du groupe source de votre application. Les conteneurs accèdent au contenu sur l'hôte en montant des volumes dans la section `mountPoints` des définitions de conteneur.

Pour de plus amples informations sur le format du fichier `Dockerrun.aws.json` et ses paramètres, veuillez consulter [Format des définitions de conteneur \(p. 72\)](#).

## Ajout de contenu

Ensuite, vous allez ajouter du contenu à votre site PHP pour l'afficher aux visiteurs, et un fichier de configuration pour le proxy nginx.

php-app/index.php

```
<h1>Hello World!!!</h1>
<h3>PHP Version <pre><?= phpversion()?></pre></h3>
```

php-app/static.html

```
<h1>Hello World!</h1>
<h3>This is a static HTML page.</h3>
```

proxy/conf.d/default.conf

```
server {
    listen 80;
    server_name localhost;
    root /var/www/html;

    index index.php;

    location ~ [^/]\.php(/|$) {
        fastcgi_split_path_info ^(.+?\.\php)(/.*)$;
        if (!-f $document_root$fastcgi_script_name) {
            return 404;
        }

        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        fastcgi_param PATH_INFO $fastcgi_path_info;
        fastcgi_param PATH_TRANSLATED $document_root$fastcgi_path_info;

        fastcgi_pass php-app:9000;
        fastcgi_index index.php;
    }
}
```

## Déploiement sur Elastic Beanstalk

Votre dossier d'application contient maintenant les fichiers suivants :

```
### Dockerrun.aws.json
### php-app
#   ### index.php
#   ### static.html
### proxy
### conf.d
### default.conf
```

C'est tout ce dont vous avez besoin pour créer l'environnement Elastic Beanstalk. Créez une archive .zip des fichiers et dossiers ci-dessus (sans inclure le dossier de projet de niveau supérieur). Pour créer l'archive dans l'Explorateur Windows, sélectionnez le contenu du dossier de projet, effectuez un clic droit, sélectionnez Envoyer vers, puis cliquez sur Dossier compressé.

### Note

Pour de plus amples informations sur la structure de fichiers requise et pour obtenir des instructions pour créer des archives dans d'autres environnements, veuillez consulter [Création d'une solution groupée d'application \(p. 415\)](#)

Ensuite, téléchargez le bundle de fichiers source sur Elastic Beanstalk et créez votre environnement. Pour Platform (Plateforme), sélectionnez Docker. Pour Platform branch (Branche de plateforme), sélectionnez

Multi-container Docker running on 64bit Amazon Linux (Docker multi-conteneurs s'exécutant sur Amazon Linux 64 bits).

#### Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk avec ce lien préconfiguré : [console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Plateforme, sélectionnez la plateforme et la branche de plateforme qui correspondent à la langue utilisée par votre application, ou la plateforme Docker pour les applications basées sur des conteneurs.
3. Pour Application code (Code d'application), choisissez Upload your code (Charger votre code).
4. Choisissez Local file (Fichier local), Choose file (Choisir un fichier), puis ouvrez le bundle source.
5. Choisissez Vérifier et lancer.
6. Vérifiez les paramètres disponibles et choisissez Créer une application.

La console Elastic Beanstalk vous redirige vers le tableau de bord de gestion pour votre nouvel environnement. Cet écran présente l'état de l'environnement et la sortie d'événements par le service Elastic Beanstalk. Quand le statut est vert, cliquez sur l'URL à côté du nom de l'environnement pour voir votre nouveau site web.

#### Connexion à une instance de conteneur

Ensuite, vous vous connectez à une instance EC2 dans votre environnement Amazon EC2 pour voir le fonctionnement de certains des éléments mobiles.

La manière la plus simple de se connecter à une instance dans votre environnement consiste à utiliser l'interface de ligne de commande EB. Pour l'utiliser, [installez l'interface de ligne de commande EB \(p. 1028\)](#), si vous ne l'avez pas déjà fait. Vous devez également configurer votre environnement avec une paire de clés SSH Amazon EC2. Utilisez soit la [page de configuration de la sécurité \(p. 626\)](#) de la console, soit la commande [eb init \(p. 1093\)](#) de l'interface de ligne de commande EB. Pour vous connecter à une instance de l'environnement, utilisez la commande [eb ssh \(p. 1113\)](#) de l'interface de ligne de commande EB.

Maintenant que vous êtes connecté à l'instance Amazon EC2 hébergeant vos conteneurs docker, vous avez accès à la configuration. Exécutez ls sur /var/app/current :

```
[ec2-user@ip-10-0-0-117 ~]$ ls /var/app/current
Dockerrun.aws.json  php-app  proxy
```

Ce répertoire contient les fichiers du groupe source que vous avez téléchargés sur Elastic Beanstalk pendant la création de l'environnement.

```
[ec2-user@ip-10-0-0-117 ~]$ ls /var/log/containers
nginx                           nginx-proxy-fffffd873ada5-stdouterr.log  rotated
nginx-66a4fd37eb63-stdouterr.log  php-app
nginx-proxy                      php-app-b894601a1364-stdouterr.log
```

C'est à cet emplacement que les journaux sont créés sur l'instance de conteneur et collectés par Elastic Beanstalk. Elastic Beanstalk crée un volume dans ce répertoire pour chaque conteneur, que vous montez sur l'emplacement du conteneur où les journaux sont écrits.

Vous pouvez également regarder Docker pour voir les conteneurs en cours d'exécution avec docker ps.

```
[ec2-user@ip-10-0-0-117 ~]$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            NAMES
STATUS              PORTS
```

```
fffffd873ada5      nginx:1.7          "nginx -g 'daemon off;' About an
hour ago  Up About an hour    443/tcp, 0.0.0.0:80->80/tcp  ecs-eb-dv-example-env-
ycmk5geqrm-2-nginx-proxy-90fce996cc8cbe8b2800
b894601a1364      php:5-fpm        "php-fpm"           About an
hour ago  Up About an hour    9000/tcp                  ecs-eb-dv-example-env-
ycmk5geqrm-2-php-app-cec0918ed1a3a49a8001
09fb19828e38      amazon/amazon-ecs-agent:latest  "/agent"           About an hour
ago      Up About an hour    127.0.0.1:51678->51678/tcp  ecs-agent
```

Vous pouvez y voir les deux conteneurs en cours d'exécution que vous avez déployés, ainsi que l'agent de conteneur Amazon ECS qui a coordonné le déploiement.

## Mise à jour de l'agent du conteneur Amazon ECS

Les instances Amazon EC2 d'un environnement Docker multiconteneurs sur Elastic Beanstalk exécutent un processus d'agent dans un conteneur Docker. Cet agent se connecte au service Amazon ECS afin de coordonner des déploiements de conteneurs. Ces déploiements sont exécutés comme des tâches dans Amazon ECS, qui sont configurées dans les fichiers de définition de tâche. Elastic Beanstalk crée ces fichiers de définition de tâche en se basant sur le fichier `Dockerrun.aws.json` que vous téléchargez dans un groupe de fichiers source.

Vérifiez le statut de l'agent de conteneur avec une demande get HTTP pour `http://localhost:51678/v1/metadata`:

```
[ec2-user@ip-10-0-0-117 ~]$ curl http://localhost:51678/v1/metadata
{
  "Cluster": "eb-dv-example-env-qpoxiguye24",
  "ContainerInstanceArn": "arn:aws:ecs:us-east-2:123456789012:container-
instance/6a72af64-2838-400d-be09-3ab2d836ebcd"
}
```

Cette structure affiche le nom du cluster Amazon ECS et l'ARN ([Amazon Resource Name](#)) de l'instance de cluster (l'instance Amazon EC2 à laquelle vous êtes connecté).

Pour plus d'informations, effectuez une demande get HTTP disponible sur `http://localhost:51678/v1/tasks`:

```
[ec2-user@ip-10-0-0-117 ~]$ curl http://localhost:51678/v1/tasks
{
  "Tasks": [
    {
      "Arn": "arn:aws:ecs:us-east-2:123456789012:task/3ff2bf0f-790d-4f6d-affb-5b127b3b6e4a",
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Family": "eb-dv-example-env-qpoxiguye24",
      "Version": "2",
      "Containers": [
        {
          "DockerId": "b894601a1364a438156a239813c77cdef17040785bc4d5e49349470dc1556b15",
          "DockerName": "ecs-eb-dv-example-env-qpoxiguye24-2-php-app-cec0918ed1a3a49a8001",
          "Name": "php-app"
        },
        {
          "DockerId": "fffffd873ada5f537c88862cce4e1de7ec3edf962645982fb236961c833a5d0fe",
          "DockerName": "ecs-eb-dv-example-env-qpoxiguye24-2-nginx-
proxy-90fce996cc8cbe8b2800",
          "Name": "nginx-proxy"
        }
      ]
    }
  ]
}
```

Cette structure décrit la tâche qui est exécutée pour déployer les deux conteneurs Docker à partir du projet exemple de ce didacticiel. Les informations suivantes sont affichées :

- KnownStatus – Le statut `RUNNING` indique que les conteneurs sont toujours actifs.
- Famille – Nom de la définition de tâche créée par Elastic Beanstalk à partir du fichier `Dockerrun.aws.json`.
- Version – Version de la définition de tâche. Elle est augmentée chaque fois que le fichier de définition de tâche est mis à jour.
- Containers – Informations sur les conteneurs exécutés sur l'instance.

Encore plus d'informations sont disponibles à partir du service Amazon ECS lui-même, que vous pouvez appeler à l'aide de l'AWS Command Line Interface. Pour obtenir des instructions sur l'utilisation de l'interface de ligne de commande (AWS CLI) avec Amazon ECS et des informations sur Amazon ECS en général, consultez le [Guide de l'utilisateur d'Amazon ECS](#).

## Migration vers la plateforme Docker Amazon Linux 2

### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [called “Mise à niveau vers Amazon Linux 2” \(p. 508\)](#).

Vous pouvez migrer vos applications qui s'exécutent sur la [plateforme Docker multiconteneurs sur l'AMI Amazon Linux \(p. 64\)](#) vers la plateforme Docker Amazon Linux 2. La plateforme Docker multiconteneurs sur l'AMI Amazon Linux nécessite de spécifier des images d'application prédéfinies à exécuter en tant que conteneurs. Après la migration, cette limitation ne s'applique plus, car la plateforme Docker Amazon Linux 2 permet également à Elastic Beanstalk de créer vos images de conteneur pendant le déploiement.

Vos applications continueront à s'exécuter dans des environnements multiconteneurs tout en profitant des avantages supplémentaires que procure l'outil Docker Compose. Pour en savoir plus sur Docker Compose et savoir comment l'installer, veuillez consulter les sites Docker [Overview of Docker Compose](#) et [Install Docker Compose](#).

### le fichier `docker-compose.yml` ;

L'outil Docker Compose utilise le fichier `docker-compose.yml` pour la configuration de vos services d'application. Ce fichier remplace votre fichier `Dockerrun.aws.json v2` dans votre répertoire de projet d'application et dans le bundle de fichiers source de l'application. Vous créez le fichier `docker-compose.yml` manuellement et il s'avérera utile de référencer votre fichier `Dockerrun.aws.json v2` pour la plupart des valeurs de paramètre.

Voici un exemple de fichier `docker-compose.yml` et le fichier `Dockerrun.aws.json v2` correspondant pour la même application. Pour de plus amples informations sur le fichier `docker-compose.yml`, veuillez consulter [Compose file reference](#). Pour de plus amples informations sur le fichier `Dockerrun.aws.json v2`, veuillez consulter [Dockerrun.aws.json V2 \(p. 69\)](#).

<code>docker-compose.yml</code>	<code>Dockerrun.aws.json v2</code>
<pre>version: '2.4' services:   php-app:     image: "php:fpm"     volumes:</pre>	<pre>{   "AWSEBDockerrunVersion": 2,   "volumes": [     {       "name": "php-app",</pre>

<b>docker-compose.yml</b>	<b>Dockerrun.aws.json v2</b>
<pre>         - "./php-app:/var/www/html:ro"         - "\${EB_LOG_BASE_DIR}/php-app:/var/ log/sample-app"       mem_limit: 128m       environment:         Container: PHP       nginx-proxy:         image: "nginx"         ports:           - "80:80"         volumes:           - "./php-app:/var/www/html:ro"           - "./proxy/conf.d:/etc/nginx/ conf.d:ro"           - "\${EB_LOG_BASE_DIR}/nginx-proxy:/var/ log/nginx"       mem_limit: 128m       links:         - php-app     </pre>	<pre>       "host": {         "sourcePath": "/var/app/current/ php-app"       }     },     {       "name": "nginx-proxy-conf",       "host": {         "sourcePath": "/var/app/current/ proxy/conf.d"       }     }   ],   "containerDefinitions": [     {       "name": "php-app",       "image": "php:fpm",       "environment": [         {           "name": "Container",           "value": "PHP"         }       ],       "essential": true,       "memory": 128,       "mountPoints": [         {           "sourceVolume": "php-app",           "containerPath": "/var/www/ html",           "readOnly": true         }       ]     },     {       "name": "nginx-proxy",       "image": "nginx",       "essential": true,       "memory": 128,       "portMappings": [         {           "hostPort": 80,           "containerPort": 80         }       ],       "links": [         "php-app"       ],       "mountPoints": [         {           "sourceVolume": "php-app",           "containerPath": "/var/www/ html",           "readOnly": true         },         {           "sourceVolume": "nginx-proxy- conf",           "containerPath": "/etc/nginx/ conf.d",           "readOnly": true         },         {           "sourceVolume": "nginx-proxy- conf",           "containerPath": "/etc/nginx/ conf.d"         }       ]     }   ] } </pre>

<code>docker-compose.yml</code>	<code>Dockerrun.aws.json v2</code>
	<pre>       "sourceVolume": "awseb-logs- nginx-proxy",       "containerPath": "/var/log/ nginx"     }   ] } ] } </pre>

## Considérations supplémentaires sur la migration

La plateforme Docker Amazon Linux 2 et la plateforme AMI multiconteneurs Amazon Linux implémentent les propriétés d'environnement différemment. Ces deux plateformes ont également des répertoires de journaux différents créés par Elastic Beanstalk pour chacun de leurs conteneurs. Après avoir effectué la migration à partir de la plateforme Docker multi-conteneur AMI Amazon Linux, vous devez être conscient de ces différentes implémentations pour votre nouvel environnement de plateforme Docker Amazon Linux 2.

Area	Plateforme Docker sur Amazon Linux 2 avec Docker Compose	Plateforme Docker multiconteneurs sur l'AMI Amazon Linux
Propriétés de l'environnement	Pour que vos conteneurs puissent accéder aux propriétés de l'environnement, vous devez ajouter une référence au fichier <code>.env</code> dans le fichier <code>docker-compose.yml</code> . Elastic Beanstalk génère le fichier <code>.env</code> , répertoriant chaque propriété en tant que variable d'environnement. Pour plus d'informations, consultez <a href="#">Référencement de variables d'environnement dans les conteneurs (p. 86)</a> .	Elastic Beanstalk peut transmettre directement les propriétés de l'environnement au conteneur. Votre code qui s'exécute dans le conteneur peut accéder à ces propriétés en tant que variables d'environnement sans configuration supplémentaire.
Répertoires de journaux	Pour chaque conteneur, Elastic Beanstalk crée un répertoire de journal appelé <code>/var/log/eb-docker/containers/&lt;service name&gt;</code> (ou <code>#{EB_LOG_BASE_DIR}/&lt;service name&gt;</code> ). Pour plus d'informations, consultez <a href="#">Journalisation personnalisée du conteneur Docker (Docker Compose) (p. 88)</a> .	Pour chaque conteneur, Elastic Beanstalk crée un répertoire de journal appelé <code>/var/log/containers/&lt;containernname&gt;</code> . Pour de plus amples informations, veuillez consulter le champ <code>mountPoints</code> dans <a href="#">Format des définitions de conteneur (p. 72)</a> .

## Etapes de la migration

### Pour migrer vers la plateforme Docker Amazon Linux 2

- Créez le fichier `docker-compose.yml` de votre application, en fonction de son fichier `Dockerrun.aws.json v2` existant. Pour de plus amples informations, veuillez consulter la section ci-dessus [le fichier docker-compose.yml ; \(p. 79\)](#).
- Dans le répertoire racine de votre dossier de projet d'application, remplacez le fichier `Dockerrun.aws.json v2` par le fichier `docker-compose.yml` que vous venez de créer.

La structure de votre répertoire doit être la suivante.

```
~/myApplication
|-- docker-compose.yml
|-- .ebextensions
|-- php-app
|-- proxy
```

3. Utilisez la commande eb init pour configurer votre répertoire local pour le déploiement vers Elastic Beanstalk.

```
~/myApplication$ eb init -p docker application-name
```

4. Utilisez la commande eb create pour créer un environnement et déployer votre image Docker.

```
~/myApplication$ eb create environment-name
```

5. Si votre application est une application web, après le lancement de votre environnement, utilisez la commande eb open pour l'afficher dans un navigateur web.

```
~/myApplication$ eb open environment-name
```

6. Vous pouvez afficher l'état de votre environnement nouvellement créé à l'aide de la commande eb status.

```
~/myApplication$ eb status environment-name
```

## Conteneurs Docker préconfigurés

Elastic Beanstalk comporte une branche de plateforme qui exécute un conteneur Docker qui est préconfiguré avec la pile logicielle du serveur d'application Java EE GlassFish. Vous pouvez utiliser le conteneur Docker préconfiguré pour développer et tester votre application localement, puis la déployer dans un environnement Elastic Beanstalk identique à votre environnement local.

### Notes

- Elastic Beanstalk prend également en charge les branches de plateforme avec des conteneurs Docker préconfigurés pour Go et Python. La mise hors service de ces branches de plateforme est programmée.
- Toutes les branches de la plateforme Docker préconfigurée utilisent le système d'exploitation AMI Amazon Linux (antérieur à Amazon Linux 2). Pour migrer votre application GlassFish vers Amazon Linux 2, utilisez la plateforme Docker générique et déployez GlassFish et votre code d'application vers une image Docker Amazon Linux 2. Pour plus d'informations, consultez [the section called “Tutorial - GlassFish sur Docker : chemin vers Amazon Linux 2” \(p. 61\)](#).

La section suivante fournit une procédure détaillée pour déployer une application dans Elastic Beanstalk à l'aide d'un conteneur Docker préconfiguré.

Pour plus d'informations sur les versions de plateforme Docker préconfigurées actuellement prises en charge, consultez la page [Docker préconfiguré](#) dans le document Plateformes AWS Elastic Beanstalk .

## Mise en route avec les conteneurs Docker préconfigurés

Cette section décrit comment développer localement un exemple d'application, puis déployer l'application dans Elastic Beanstalk à l'aide d'un conteneur Docker préconfiguré.

## Configuration de votre environnement de développement local

Pour cette procédure, nous utilisons un exemple d'application GlassFish.

### Configuration de votre environnement

- Créez un dossier pour l'exemple d'application.

```
~$ mkdir eb-preconf-example  
~$ cd eb-preconf-example
```

- Téléchargez le code de l'exemple d'application dans le nouveau dossier.

```
~$ wget https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/samples/docker-glassfish-v1.zip  
~$ unzip docker-glassfish-v1.zip  
~$ rm docker-glassfish-v1.zip
```

## Développement et test en local

Pour développer un exemple d'application GlassFish

- Ajoutez un Dockerfile au dossier racine de votre application. Dans le fichier, spécifiez l'image de base Docker AWS Elastic Beanstalk à utiliser pour exécuter votre conteneur Docker préconfiguré local. Vous déployerez ultérieurement votre application sur une version de plateforme Docker GlassFish préconfigurée Elastic Beanstalk. Choisissez l'image de base Docker que cette version de plateforme utilise. Pour déterminer l'image Docker actuelle de la version de plateforme, consultez la section [Docker préconfiguré](#) de la page Plateformes prises en charge AWS Elastic Beanstalk dans le guide Plateformes AWS Elastic Beanstalk.

Example ~/Eb-preconf-example/Dockerfile

```
# For Glassfish 5.0 Java 8  
FROM amazon/aws-eb-glassfish:5.0-al-onbuild-2.11.1
```

Pour plus d'informations sur l'utilisation d'un Dockerfile, consultez [Configuration Docker \(p. 53\)](#).

- Développez l'image Docker.

```
~/eb-preconf-example$ docker build -t my-app-image .
```

- Exécutez le conteneur Docker à partir de l'image.

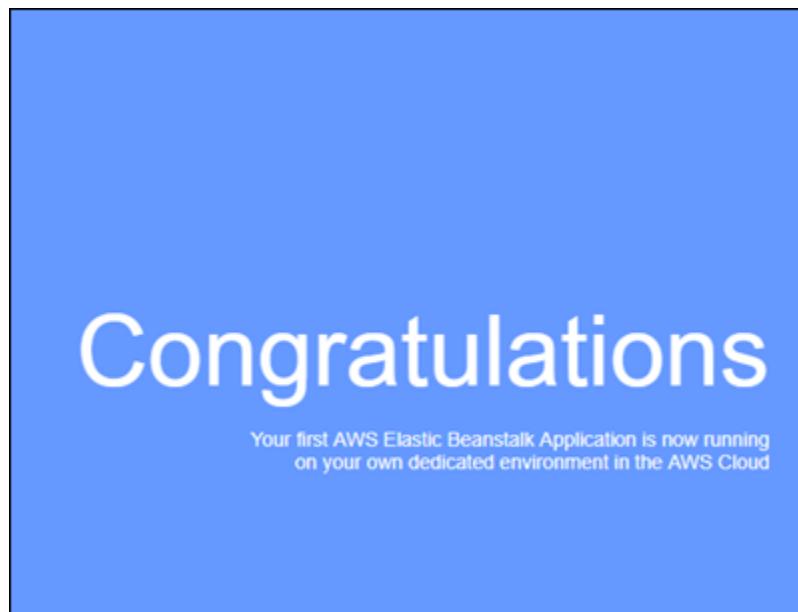
### Note

Vous devez inclure l'indicateur `-p` pour mapper le port 8080 sur le conteneur au port localhost 3000. Les conteneurs Docker Elastic Beanstalk exposent toujours l'application sur le port 8080 sur le conteneur. Les indicateurs `-it` exécutent l'image comme un processus interactif. L'indicateur `--rm` supprime le système de fichiers conteneur lorsque le conteneur s'arrête. Vous pouvez inclure le cas échéant l'indicateur `-d` pour exécuter l'image comme un démon.

```
$ docker run -it --rm -p 3000:8080 my-app-image
```

- Pour voir l'exemple d'application, tapez l'URL suivante dans votre navigateur web.

<http://localhost:3000>



## What's Next?

- [Learn how to build, deploy, and manage your own Elastic Beanstalk](#)
- [AWS Elastic Beanstalk concepts](#)
- [Learn how to create new application versions](#)
- [Learn how to manage your application environment](#)

## GlassFish Samples

- [Explore community-provided sample applications](#)

## Déploiement sur Elastic Beanstalk

Après avoir testé votre application, vous êtes prêt à la déployer dans Elastic Beanstalk.

Pour déployer votre application sur Elastic Beanstalk

1. Dans le dossier racine de votre application, renommez le Dockerfile Dockerfile.local. Cette étape est obligatoire pour qu'Elastic Beanstalk utilise le fichier Dockerfile qui contient les instructions correctes pour permettre à Elastic Beanstalk de développer une image Docker personnalisée sur chaque instance Amazon EC2 dans votre environnement Elastic Beanstalk.

### Note

Vous n'avez pas besoin d'effectuer cette étape si votre Dockerfile inclut des instructions qui modifient l'image Docker de base de la version de plateforme. Vous n'avez pas besoin d'utiliser un Dockerfile si votre Dockerfile contient seulement une ligne FROM pour spécifier l'image de base à partir de laquelle développer le conteneur. Dans ce cas, le Dockerfile est redondant.

2. Créez un groupe source d'application.

```
~/eb-preconf-example$ zip myapp.zip -r *
```

3. Ouvrez la console Elastic Beanstalk avec ce lien préconfiguré : [console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
4. Pour Plateforme, sous Preconfigured – Docker (Préconfiguré – Docker), choisissez Glassfish.
5. Pour Code de l'application, choisissez Charger votre code puis Charger.
6. Choisissez Fichier local, Browse, puis ouvrez le groupe source d'application que vous venez de créer.
7. Choisissez Upload.
8. Choisissez Vérifier et lancer.

9. Vérifiez les paramètres disponibles et choisissez Crée une application.
10. Lorsque l'environnement est créé, vous pouvez afficher l'application déployée. Choisissez l'URL de l'environnement qui s'affiche en haut du tableau de bord de la console.

## Configuration des environnements Docker

Il existe plusieurs façons de configurer le comportement de votre environnement Docker Elastic Beanstalk.

### Note

Si votre environnement Elastic Beanstalk utilise une version de plateforme Docker AMI Amazon Linux (antérieure à Amazon Linux 2), lisez impérativement les informations supplémentaires dans [the section called “Configuration Docker sur l'AMI Amazon Linux \(antérieure à Amazon Linux 2\)” \(p. 92\)](#).

### Sections

- [Configuration des logiciels dans les environnements Docker \(p. 85\)](#)
- [Référencement de variables d'environnement dans les conteneurs \(p. 86\)](#)
- [Génération de journaux pour la création de rapports d'intégrité améliorée \(Docker Compose\) \(p. 87\)](#)
- [Journalisation personnalisée du conteneur Docker \(Docker Compose\) \(p. 88\)](#)
- [Images Docker \(p. 89\)](#)
- [Récupération de l'espace de stockage Docker \(p. 91\)](#)
- [Configuration des mises à jour gérées pour les environnements Docker \(p. 91\)](#)
- [Espaces de noms de la configuration Python \(p. 92\)](#)
- [Configuration Docker sur l'AMI Amazon Linux \(antérieure à Amazon Linux 2\) \(p. 92\)](#)

## Configuration des logiciels dans les environnements Docker

Vous pouvez utiliser la console Elastic Beanstalk pour configurer le logiciel s'exécutant sur les instances de votre environnement.

Pour configurer votre environnement Docker dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Effectuez les modifications de configuration nécessaires.
6. Choisissez Apply.

Pour de plus amples informations sur la configuration des paramètres logiciels dans n'importe quel environnement, veuillez consulter [the section called “Paramètres du logiciel” \(p. 632\)](#). Les sections suivantes couvrent des informations spécifiques à Docker.

## Options du conteneur

La section Options du conteneur contient des options spécifiques à la plateforme. Pour les environnements Docker, il vous permet de choisir si votre environnement inclut ou non le serveur proxy Nginx.

### Environnements avec Docker Compose

Si vous gérez votre environnement Docker avec Docker Compose, Elastic Beanstalk suppose que vous exécutez un serveur proxy en tant que conteneur. Par conséquent, sa valeur par défaut est None (Aucun) pour le paramètre du Proxy server (Serveur proxy) et Elastic Beanstalk ne fournit pas de configuration NGINX.

#### Note

Même si vous sélectionnez NGINX comme serveur proxy, ce paramètre est ignoré dans un environnement avec Docker Compose. Par défaut, le paramètre Proxy server (Serveur proxy) est toujours None (Aucun).

Étant donné que le proxy de serveur web NGINX est désactivé pour la plateforme Docker sur Amazon Linux 2 avec Docker Compose, vous devez suivre les instructions pour générer des journaux afin de profiter de rapports d'état améliorés. Pour plus d'informations, consultez [Génération de journaux pour la création de rapports d'intégrité améliorée \(Docker Compose\) \(p. 87\)](#).

## Propriétés de l'environnement et variables d'environnement

La section Environment Properties (Propriétés de l'environnement) vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon Elastic Compute Cloud (Amazon EC2) exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application. Dans un environnement Docker, Elastic Beanstalk transmet les propriétés de l'environnement aux conteneurs en tant que variables d'environnement.

Votre code d'application s'exécutant dans un conteneur peut faire référence à une variable d'environnement par son nom et lire sa valeur. Le code source qui lit ces variables d'environnement varie selon le langage de programmation. Pour obtenir des instructions pour lire les valeurs des variables d'environnement dans les langages de programmation pris en charge par les plateformes gérées par Elastic Beanstalk, veuillez consulter la rubrique correspondante de chaque plateforme. Pour obtenir la liste des liens vers ces rubriques, veuillez consulter [the section called "Paramètres du logiciel" \(p. 632\)](#).

### Environnements avec Docker Compose

Si vous gérez votre environnement Docker avec Docker Compose, vous devez effectuer une configuration supplémentaire pour récupérer les variables d'environnement dans les conteneurs. Pour que les fichiers exécutables qui s'exécutent dans votre conteneur puissent accéder à ces variables d'environnement, vous devez les référencer dans le fichier `docker-compose.yml`. Pour plus d'informations, consultez [Référencement de variables d'environnement dans les conteneurs \(p. 86\)](#).

## Référencement de variables d'environnement dans les conteneurs

Si vous utilisez l'outil Docker Compose sur la plateforme Docker Amazon Linux 2, Elastic Beanstalk génère un fichier d'environnement Docker Compose appelé `.env` dans le répertoire racine de votre projet d'application. Ce fichier stocke les variables d'environnement que vous avez configurées pour Elastic Beanstalk.

#### Note

Si vous incluez un fichier `.env` dans le bundle de fichiers de votre application, Elastic Beanstalk ne générera pas de fichier `.env`.

Pour qu'un conteneur référence les variables d'environnement que vous définissez dans Elastic Beanstalk, vous devez suivre l'une de ces approches de configuration, ou les deux.

- Ajoutez le fichier `.env` généré par Elastic Beanstalk à l'option de configuration `env_file` dans le fichier `docker-compose.yml`.
- Définissez directement les variables d'environnement dans le fichier `docker-compose.yml`.

Les fichiers suivants fournissent un exemple. L'exemple de fichier `docker-compose.yml` illustre les deux approches.

- Si vous définissez les propriétés d'environnement `DEBUG_LEVEL=1` et `LOG_LEVEL=error`, Elastic Beanstalk génère le fichier `.env` suivant pour vous :

```
DEBUG_LEVEL=1
LOG_LEVEL=error
```

- Dans ce fichier `docker-compose.yml`, l'option de configuration `env_file` pointe vers le fichier `.env`, et elle définit également la variable d'environnement `DEBUG=1` directement dans le fichier `docker-compose.yml`.

```
services:
  web:
    build: .
    environment:
      - DEBUG=1
    env_file:
      - .env
```

## Notes

- Si vous définissez la même variable d'environnement dans les deux fichiers, la variable définie dans le fichier `docker-compose.yml` a une priorité plus élevée que la variable définie dans le fichier `.env`.
- Veillez à ne pas laisser d'espace entre le signe égal (=) et la valeur attribuée à votre variable, afin d'empêcher l'ajout d'espaces à la chaîne.

Pour en savoir plus sur les variables d'environnement dans Docker Compose, veuillez consulter [Environment variables in Compose](#)

## Génération de journaux pour la création de rapports d'intégrité améliorée (Docker Compose)

L'[agent d'état](#) (p. 840) fournit des métriques sur l'état du système d'exploitation et des applications pour les environnements Elastic Beanstalk. Il s'appuie sur des formats de journaux de serveur web qui transmettent les informations dans un format spécifique.

Elastic Beanstalk suppose que vous exécutez un proxy de serveur web en tant que conteneur. Par conséquent, le proxy du serveur web NGINX est désactivé pour les environnements Docker exécutant Docker Compose. Vous devez configurer votre serveur pour qu'il écrive les journaux à l'emplacement et au format utilisés par l'agent d'état Elastic Beanstalk. Cela vous permet d'utiliser pleinement les rapports d'intégrité améliorés, même si le proxy du serveur web est désactivé.

Pour obtenir des instructions sur la façon de procéder, veuillez consulter [Configuration de journal de serveur web](#) (p. 870)

## Journalisation personnalisée du conteneur Docker (Docker Compose)

Afin de résoudre efficacement les problèmes et de surveiller vos services conteneurisés, vous pouvez [demander des journaux d'instance \(p. 884\)](#) à Elastic Beanstalk à partir de la console de gestion de l'environnement ou de l'interface de ligne de commande EB. Les journaux d'instance sont composés de journaux de groupe et de journaux de fin, combinés et empaquetés pour vous permettre d'afficher les journaux et les événements récents de manière efficace et directe.

Elastic Beanstalk crée des répertoires de journaux sur l'instance de conteneur, un pour chaque service défini dans le fichier `docker-compose.yml`, à l'emplacement `/var/log/eb-docker/containers/<service name>`. Si vous utilisez la fonctionnalité Docker Compose sur la plateforme Docker Amazon Linux 2, vous pouvez monter ces répertoires à l'emplacement souhaité dans la structure du fichier de conteneur où les journaux sont écrits. Lorsque vous montez des répertoires de journaux pour écrire des données de journaux, Elastic Beanstalk peut collecter les données de journaux à partir de ces répertoires.

Si vos applications se trouvent sur une plateforme Docker qui n'utilise pas Docker Compose, vous pouvez suivre la procédure standard décrite dans [Journalisation personnalisée du conteneur Docker \(Docker Compose\) \(p. 88\)](#).

Pour configurer les fichiers journaux de votre service afin qu'ils soient des fichiers de fin et des journaux de groupe récupérables

1. Modifiez le fichier `docker-compose.yml`.
2. Sous la clé `volumes` de votre service, ajoutez un montage lié de la manière suivante :

```
"${EB_LOG_BASE_DIR}/<service name>:<log directory inside container>
```

Dans l'exemple de fichier `docker-compose.yml` ci-dessous :

- `nginx-proxy` est `<nom du service>`
- `/var/log/nginx` est `<conteneur intérieur du répertoire de journaux>`

```
services:  
  nginx-proxy:  
    image: "nginx"  
    volumes:  
      - "${EB_LOG_BASE_DIR}/nginx-proxy:/var/log/nginx"
```

- Le répertoire `var/log/nginx` contient les journaux du service `nginx-proxy` dans le conteneur, et il sera mappé au répertoire `/var/log/eb-docker/containers/nginx-proxy` sur l'hôte.
- Tous les journaux de ce répertoire peuvent désormais être récupérés sous forme de journaux de processus et de groupe via la fonctionnalité de [demande de journaux d'instance \(p. 884\)](#) d'Elastic Beanstalk.

### Notes

- `${EB_LOG_BASE_DIR}` est une variable d'environnement définie par Elastic Beanstalk avec la valeur `/var/log/eb-docker/containers`.
- Elastic Beanstalk crée automatiquement le répertoire `/var/log/eb-docker/containers/<service name>` pour chaque service dans le fichier `docker-compose.yml`.

## Images Docker

Les plateformes Docker et Docker multiconteneurs pour Elastic Beanstalk prennent en charge l'utilisation d'images Docker stockées dans un référentiel d'images en ligne public ou privé.

Spécifiez des images par nom dans `Dockerrun.aws.json`. Notez ces conventions :

- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu` ou `account-id.dkr.ecr.us-east-2.amazonaws.com/ubuntu:trust`).

Pour les environnements utilisant la plateforme Docker uniquement, vous pouvez également créer votre propre image lors de la création d'environnement avec un fichier Dockerfile. Consultez [Création d'images personnalisées avec un Dockerfile \(p. 57\)](#) pour plus de détails. La plateforme Docker multi-conteneurs ne prend pas en charge cette fonctionnalité.

### Utilisation d'images à partir d'un référentiel Amazon ECR

Vous pouvez stocker vos images Docker personnalisées dans AWS avec [Amazon Elastic Container Registry](#) (Amazon ECR). Lorsque vous stockez vos images Docker dans Amazon ECR, Elastic Beanstalk s'authentifie automatiquement sur le registre Amazon ECR avec le [profil d'instance \(p. 22\)](#) de votre environnement. Vous n'avez donc pas besoin de [générer un fichier d'authentification \(p. 90\)](#) ni de le charger dans Amazon Simple Storage Service (Amazon S3).

Vous devez, toutefois, fournir vos instances avec l'autorisation d'accéder aux images dans votre référentiel Amazon ECR en ajoutant des autorisations au profil d'instance de votre environnement. Vous pouvez attacher la stratégie gérée [AmazonEC2ContainerRegistryReadOnly](#) au profil d'instance pour fournir un accès en lecture seule à tous les référentiels Amazon ECR de votre compte, ou accorder un accès au référentiel unique en utilisant le modèle suivant pour créer une stratégie personnalisée :

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowEbAuth",  
            "Effect": "Allow",  
            "Action": [  
                "ecr:GetAuthorizationToken"  
            ],  
            "Resource": [  
                "*"  
            ]  
        },  
        {  
            "Sid": "AllowPull",  
            "Effect": "Allow",  
            "Resource": [  
                "arn:aws:ecr:us-east-2:account-id:repository/repository-name"  
            ],  
            "Action": [  
                "ecr:GetAuthorizationToken",  
                "ecr:BatchCheckLayerAvailability",  
                "ecr:GetDownloadUrlForLayer",  
                "ecr:getRepositoryPolicy",  
                "ecr:DescribeRepositories",  
                "ecr:ListTagsForResource",  
                "ecr:PutImage",  
                "ecr:BatchGetImage",  
                "ecr:CompleteLayerUpload",  
                "ecr:StartLayerUpload",  
                "ecr:DeleteRepository",  
                "ecr:DeleteImage",  
                "ecr:DescribeImage",  
                "ecr:BatchDeleteImage",  
                "ecr:TagImage",  
                "ecr:UntagImage",  
                "ecr:ListImages",  
                "ecr:ListTagsForResource"  
            ]  
        }  
    ]  
}
```

```
        "ecr>ListImages",
        "ecrBatchGetImage"
    ]
}
]
```

Remplacez le nom Amazon Resource Name (ARN) dans la stratégie ci-dessus par l'ARN de votre référentiel.

Dans votre fichier `Dockerrun.aws.json`, reportez-vous à l'image par URL. Pour [Plateforme Docker \(p. 53\)](#), l'URL va dans la définition `Image` :

```
"Image": {
    "Name": "account-id.dkr.ecr.us-east-2.amazonaws.com/repository-name:latest",
    "Update": "true"
},
```

Pour [Plateforme multi-conteneurs \(p. 69\)](#), utilisez la clé `image` dans un objet de définition de conteneur :

```
"containerDefinitions": [
    {
        "name": "my-image",
        "image": "account-id.dkr.ecr.us-east-2.amazonaws.com/repository-name:latest"
    }
],
```

### Utilisation d'images à partir d'un référentiel privé

Pour utiliser une image Docker dans un référentiel privé hébergé par un registre en ligne, vous devez fournir un fichier d'authentification qui contient les informations requises pour s'authentifier avec le registre.

Générez un fichier d'authentification avec la commande `docker login`. Pour les référentiels sur Docker Hub, exécutez `docker login`:

```
$ docker login
```

Pour d'autres registres, incluez l'URL du serveur de registre :

```
$ docker login registry-server-url
```

#### Note

Si votre environnement Elastic Beanstalk utilise une version de plateforme Docker AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans [the section called “Configuration Docker sur l'AMI Amazon Linux \(antérieure à Amazon Linux 2\)” \(p. 92\)](#).

Chargez une copie nommée `.dockercfg` du fichier d'authentification sur un compartiment Amazon S3 sécurisé. Le compartiment Amazon S3 doit être hébergé dans la même région AWS que l'environnement qui l'utilise. Elastic Beanstalk ne peut pas télécharger de fichiers à partir d'un compartiment Amazon S3 hébergé dans d'autres régions. Accordez des autorisations pour l'opération `s3:GetObject` au rôle IAM dans le profil d'instance. Pour plus d'informations, consultez [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#).

Incluez les informations sur le compartiment Amazon S3 dans le paramètre `Authentication (v1)` ou `authentication (v2)` de votre fichier `Dockerrun.aws.json`.

Pour de plus amples informations sur le format `Dockerrun.aws.json` des environnements Docker, veuillez consulter [Configuration Docker \(p. 53\)](#). Pour les environnements multiconteneurs, consultez [Configuration Docker multi-conteneurs \(p. 69\)](#).

Pour plus d'informations sur le fichier d'authentification, consultez [Store images on Docker Hub et docker login](#) sur le site web de Docker.

## Récupération de l'espace de stockage Docker

Docker ne nettoie (supprime) pas l'espace utilisé lorsqu'un fichier est créé, puis supprimé à partir d'un conteneur en cours d'exécution ; l'espace est uniquement renvoyé dans le pool une fois que le conteneur est supprimé. Cela pose un problème si un processus de conteneur crée et supprime de nombreux fichiers, par exemple, en vidant régulièrement des sauvegardes de base de données, remplissant ainsi l'espace de stockage de l'application.

Une solution consiste à augmenter la taille de l'espace de stockage d'application, comme décrit dans la section précédente. L'autre option est moins performante : exécuter périodiquement `fstrim` sur le système d'exploitation hôte à l'aide de `cron`, sur l'espace libre du conteneur pour récupérer les blocs de données de conteneur inutilisés.

```
docker ps -q | xargs docker inspect --format='{{ .State.Pid }}' | xargs -IZ sudo fstrim /proc/z/root/
```

## Configuration des mises à jour gérées pour les environnements Docker

Avec les [mises à jour gérées de la plateforme \(p. 501\)](#), vous pouvez configurer votre environnement afin qu'il se mette à jour automatiquement avec la dernière version d'une plateforme selon un calendrier défini.

Dans le cas des environnements Docker, vous pouvez déterminer si une mise à jour de la plateforme automatique doit être appliquée en cas de changement de version de Docker (lorsque la nouvelle version de plateforme inclut une nouvelle version de Docker). Elastic Beanstalk prend en charge les mises à jour de plateformes gérées dans toutes les versions Docker lors de la mise à jour à partir d'un environnement exécutant une version de plateforme Docker antérieure à la version 2.9.0. Lorsqu'une nouvelle version de plateforme inclut une nouvelle version de Docker, Elastic Beanstalk incrémentera le numéro de version de la mise à jour mineure. Par conséquent, pour autoriser les mises à jour de plateforme gérées sur différentes versions de Docker, activez les mises à jour gérées de la plateforme pour les mises à jour de version mineure et les correctifs. Pour empêcher les mises à jour de plateforme gérées sur différentes versions de Docker, activez les mises à jour gérées de la plateforme afin d'appliquer uniquement les mises à jour contenant des correctifs.

Par exemple, le [fichier de configuration \(p. 737\)](#) suivant active les mises à jour de plateforme gérées à 9 h UTC chaque mardi pour les mises à jour de version mineure et correctifs, permettant ainsi les mises à jour gérées sur plusieurs versions de Docker :

Example .ebextensions/managed-platform-update.config

```
option_settings:
  aws:elasticbeanstalk:managedactions:
    ManagedActionsEnabled: true
    PreferredStartTime: "Tue:09:00"
  aws:elasticbeanstalk:managedactions:platformupdate:
    UpdateLevel: minor
```

Pour les environnements exécutant des versions de plateforme Docker 2.9.0 ou antérieures, Elastic Beanstalk n'effectue jamais de mises à jour des plateformes gérées si la nouvelle version de plateforme inclut une nouvelle version de Docker.

## Espaces de noms de la configuration Python

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

### Note

Ces informations s'appliquent uniquement à l'environnement Docker qui n'exécute pas Docker Compose. Cette option a un comportement différent avec les environnements Docker qui exécutent Docker Compose. Pour de plus amples informations sur les services proxy avec Docker Compose, veuillez consulter [Options du conteneur \(p. 86\)](#).

La plateforme Docker prend en charge les options des espaces de noms suivants en plus des [options prises en charge pour tous les environnements Elastic Beanstalk \(p. 679\)](#) :

- `aws:elasticbeanstalk:environment:proxy` – Choisissez le serveur proxy pour votre environnement. Docker prend en charge l'exécution de Nginx ou aucun serveur proxy.

L'exemple de fichier de configuration suivant configure un environnement Docker de façon à ce qu'il n'exécute aucun serveur proxy.

Example `.ebextensions/docker-settings.config`

```
option_settings:  
  aws:elasticbeanstalk:environment:proxy:  
    ProxyServer: none
```

## Configuration Docker sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Docker utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires de cette section.

### Utilisation d'un fichier d'authentification pour un référentiel privé

Ces informations sont pertinentes pour vous si vous [utilisez des images provenant d'un référentiel privé \(p. 90\)](#). En commençant par Docker version 1.7, la commande docker login a modifié le nom du fichier d'authentification et le format du fichier. Les versions de la plateforme Docker AMI Amazon Linux (antérieures à Amazon Linux 2) requièrent l'ancien fichier de configuration au format `~/.dockercfg`.

Avec Docker version 1.7 et les versions ultérieures, la commande docker login crée le fichier d'authentification dans `~/.docker/config.json` au format suivant.

```
{  
  "auths": {  
    "server": {  
      "auth": "key"  
    }  
  }  
}
```

Avec Docker version 1.6.2 et les versions antérieures, la commande docker login crée le fichier d'authentification dans `~/.dockercfg` au format suivant.

```
{  
  "server" :  
}
```

```
{  
    "auth" : "auth_token",  
    "email" : "email"  
}
```

Pour convertir un fichier config.json, supprimez la clé auths extérieure, ajoutez une clé email et aplatissez le document JSON pour qu'il corresponde à l'ancien format.

Sur les versions de la plateforme Docker Amazon Linux 2, Elastic Beanstalk utilise le nom et le format de fichier d'authentification les plus récents. Si vous utilisez une version de la plateforme Docker Amazon Linux 2, vous pouvez utiliser le fichier d'authentification créé par la commande docker login sans aucune conversion.

### Configuration de volumes de stockage supplémentaires

Pour améliorer les performances sur l'AMI Amazon Linux, Elastic Beanstalk configure deux volumes de stockage Amazon EBS pour les instances Amazon EC2 de votre environnement Docker. Outre le volume racine fourni pour tous les environnements Elastic Beanstalk, un deuxième volume de 12 Go nommé xvdcz est mis en service pour le stockage d'images sur les environnements Docker.

Si vous avez besoin de plus d'espace de stockage ou d'IOPS pour les images Docker, vous pouvez personnaliser le volume de stockage d'image à l'aide de l'option de configuration BlockDeviceMapping dans l'espace de noms [aws:autoscaling:launchconfiguration \(p. 681\)](#).

Par exemple, le [fichier de configuration \(p. 737\)](#) suivant augmente la taille du volume stockage à 100 Go avec 500 IOPS provisionnées :

Example .ebextensions/blockdevice-xvdcz.config

```
option_settings:  
  aws:autoscaling:launchconfiguration:  
    BlockDeviceMappings: /dev/xvdcz=:100::io1:500
```

Si vous utilisez l'option BlockDeviceMappings pour configurer des volumes supplémentaires pour votre application, vous devez inclure un mappage pour xvdcz pour vous assurer de sa création. L'exemple suivant configure deux volumes, le volume de stockage d'image xvdcz avec les paramètres par défaut et un volume d'application de 24 Go supplémentaires nommé sdh :

Example .ebextensions/blockdevice-sdh.config

```
option_settings:  
  aws:autoscaling:launchconfiguration:  
    BlockDeviceMappings: /dev/xvdcz=:12:true:gp2,/dev/sdh=:24
```

#### Note

Lorsque vous modifiez les paramètres dans cet espace de noms, Elastic Beanstalk remplace toutes les instances de votre environnement par des instances exécutant la nouvelle configuration. Consultez [Configuration changes \(p. 488\)](#) pour plus de détails.

## Exécution d'un environnement Docker localement avec l'interface de ligne de commande EB

Vous pouvez utiliser l'interface de ligne de commande Elastic Beanstalk (CLI) (EB) pour exécuter des conteneurs Docker configurés dans votre application AWS Elastic Beanstalk localement. L'interface de ligne de commande EB utilise le fichier de configuration Docker (`Dockerfile` ou `Dockerrun.aws.json`) et le code source dans votre répertoire de projet pour exécuter votre application localement dans Docker.

L'interface de ligne de commande EB prend en charge les applications en cours d'exécution localement définies à l'aide des plateformes Docker, Docker multi-conteneurs et Docker préconfigurées.

#### Rubriques

- [Conditions requises pour exécuter des applications Docker localement \(p. 94\)](#)
- [Préparation d'une application Docker en vue de son utilisation avec l'interface de ligne de commande EB \(p. 95\)](#)
- [Exécution d'une application Docker localement \(p. 95\)](#)
- [Nettoyage après l'exécution d'une application Docker localement \(p. 96\)](#)

## Conditions requises pour exécuter des applications Docker localement

- Linux OS ou Mac OS X
- [Interface de ligne de commande EB version 3.3 ou plus \(p. 1028\)](#)

Exécutez eb init dans votre répertoire de projet pour initialiser un référentiel de l'interface de ligne de commande EB. Si vous n'avez jamais utilisé l'interface de ligne de commande EB, veuillez consulter [Gestion des environnements Elastic Beanstalk avec l'interface de ligne de commande EB \(p. 1040\)](#).

- [Docker version 1.6 ou ultérieure](#)

Ajoutez vous-même au groupe docker, déconnectez-vous et puis reconnectez-vous afin de garantir que vous pouvez exécuter des commandes Docker sans sudo :

```
$ sudo usermod -a -G docker $USER
```

Exécutez docker ps pour vérifier que le démon Docker est opérationnel :

\$ docker ps	CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
	PORTE	NAMES			

- Une application Docker

Si vous n'avez pas d'application Docker dans un dossier de projet sur votre ordinateur local, consultez [Déploiement d'applications Elastic Beanstalk à partir de conteneurs Docker \(p. 46\)](#) pour une introduction à l'utilisation de Docker avec AWS Elastic Beanstalk.

- Profil de docker (facultatif)

Si votre application utilise des images Docker dans un référentiel privé, exécutez docker login et suivez les invites pour créer un profil d'authentification.

- w3m (facultatif)

W3m est un navigateur web que vous pouvez utiliser pour afficher votre application web en cours d'exécution au sein d'un terminal de ligne de commande avec eb local run. Si vous utilisez la ligne de commande dans un environnement de bureau, vous n'avez pas besoin de w3m.

Les conteneurs Docker s'exécutent localement sans émuler les ressources AWS qui sont allouées lorsque vous déployez une application Elastic Beanstalk, y compris des groupes de sécurité et des données ou des niveaux de travail.

Vous pouvez configurer vos conteneurs locaux pour vous connecter à une base de données en passant la chaîne de connexion nécessaires ou d'autres variables avec l'option envvars, mais vous devez vous

assurer que toutes les ressources dans AWS sont accessibles à partir de votre ordinateur local en [ouvrant les ports appropriés](#) dans leurs groupes de sécurité affectés ou en attachant une [passerelle par défaut](#) ou une [adresse IP Elastic](#).

## Préparation d'une application Docker en vue de son utilisation avec l'interface de ligne de commande EB

Préparez vos données source et votre fichier de configuration Docker comme si vous étiez en train de les déployer sur Elastic Beanstalk. Cette rubrique utilise l'exemple de proxy PHP et nginx du [didacticiel Docker multi-conteneurs \(p. 73\)](#) présenté précédemment dans ce manuel à titre d'exemple, mais vous pouvez utiliser les mêmes commandes avec n'importe quelle application Docker, Docker multi-conteneurs ou Docker préconfigurée.

## Exécution d'une application Docker localement

Exécutez votre application Docker localement avec la commande eb local run à partir du répertoire de projet :

```
~/project$ eb local run
Creating elasticbeanstalk_phpapp_1...
Creating elasticbeanstalk_nginxproxy_1...
Attaching to elasticbeanstalk_phpapp_1, elasticbeanstalk_nginxproxy_1
phpapp_1    | [23-Apr-2015 23:24:25] NOTICE: fpm is running, pid 1
phpapp_1    | [23-Apr-2015 23:24:25] NOTICE: ready to handle connections
```

L'interface de ligne de commande EB lit la configuration Docker et exécute les commandes Docker nécessaires pour exécuter votre application. La première fois que vous exécutez un projet localement, Docker télécharge des images à partir d'un référentiel distance et les stocke sur votre ordinateur local. Ce processus peut prendre plusieurs minutes.

### Note

La commande eb local run prend deux paramètres facultatifs, `port` et `envvars`.

Pour remplacer le port par défaut pour une application Docker multi-conteneurs, utilisez l'option `port` :

```
$ eb local run --port 8080
```

Cette commande indique l'interface de ligne de commande EB pour utiliser le port 8080 sur l'hôte et le mapper avec le port exposé sur le conteneur. Si vous ne spécifiez pas un port, l'interface de ligne de commande EB utilise le port du conteneur pour l'hôte. Cette option ne fonctionne qu'avec les applications utilisant la plateforme Docker.

Pour passer des variables d'environnement aux conteneurs d'application, utilisez l'option `envvars` :

```
$ eb local run --envvars RDS_HOST=$RDS_HOST,RDS_DB=$RDS_DB,RDS_USER=
$RDS_USER,RDS_PASS=$RDS_PASS
```

Utilisez des variables de l'environnement pour configurer une connexion de base de données, définissez des options de débogage ou transmettez des secrets en toute sécurité à votre application. Pour de plus amples informations sur les options prises en charge par les sous-commandes eb local, veuillez consulter [eb local \(p. 1097\)](#).

Une fois que les conteneurs sont opérationnels dans Docker, ils sont prêts à prendre des demandes des clients. Le processus eb local reste ouvert aussi longtemps que les conteneurs sont en cours d'exécution. Si vous avez besoin d'arrêter le processus et les conteneurs, appuyez sur Ctrl+C.

Ouvrez un second terminal pour exécuter des commandes supplémentaires pendant que le processus eb local est en cours d'exécution. Utilisez eb local status pour afficher l'état de votre application :

```
~/project$ eb local status
Platform: 64bit Amazon Linux 2014.09 v1.2.1 running Multi-container Docker 1.3.3 (Generic)
Container name: elasticbeanstalk_nginxproxy_1
Container ip: 127.0.0.1
Container running: True
Exposed host port(s): 80
Full local URL(s): 127.0.0.1:80

Container name: elasticbeanstalk_phpapp_1
Container ip: 127.0.0.1
Container running: True
Exposed host port(s): None
Full local URL(s): None
```

Vous pouvez utiliser docker ps afin de voir l'état des conteneurs du point de vue de Docker :

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
PORTS		NAMES		
6a8e71274fed	nginx:latest	"nginx -g 'daemon off;'	9 minutes ago	Up 9 minutes
	0.0.0.0:80->80/tcp, 443/tcp	elasticbeanstalk_nginxproxy_1		
82cbf620bdc1	php:fpm	"php-fpm"	9 minutes ago	Up 9 minutes
	9000/tcp	elasticbeanstalk_phpapp_1		

Ensuite, affichez votre application en action avec eb local open:

```
~/project$ eb local open
```

Cette commande ouvre votre application dans le navigateur web par défaut. Si vous exécutez un terminal dans un environnement de bureau, il peut s'agir de Firefox, de Google Chrome ou de Safari. Si vous exécutez un terminal dans un environnement sans tête ou via une connexion SSH, un navigateur de ligne de commande, comme w3m, sera utilisé s'il est disponible.

Revenez au terminal exécutant le processus de demande pendant un moment et notez le résultat supplémentaire :

```
phpapp_1 | 172.17.0.36 - 21/Apr/2015:23:46:17 +0000 "GET /index.php" 200
```

Cela montre que l'application web dans le conteneur Docker a reçu une demande HTTP GET pour index.php qui a été renvoyée avec succès avec un état 200 (sans erreur).

Exécutez eb local logs pour voir où l'interface de ligne de commande EB écrit les journaux.

```
~/project$ eb local logs
Elastic Beanstalk will write logs locally to /home/user/project/.elasticbeanstalk/logs/local.
Logs were most recently created 3 minutes ago and written to /home/user/project/.elasticbeanstalk/logs/local/150420_234011665784.
```

## Nettoyage après l'exécution d'une application Docker localement

Lorsque vous avez terminé le test de votre application localement, vous pouvez arrêter les applications et supprimer les images téléchargées par Docker lorsque vous utilisez eb local run. La suppression des images est optionnelle. Vous pouvez souhaiter les conserver pour les utiliser ultérieurement.

Revenez au terminal exécutant le processus eb local et appuyez sur Ctrl+C pour arrêter l'application :

```
^CGracefully stopping... (press Ctrl+C again to force)
Stopping elasticbeanstalk_nginxproxy_1...
Stopping elasticbeanstalk_phpapp_1...

Aborting.
[1]+  Exit 5                  eb local run
```

L'interface de ligne de commande EB tente d'arrêter chaque conteneur en cours d'exécution normalement à l'aide de commandes Docker. Si vous avez besoin d'arrêter un processus immédiatement, appuyez à nouveau sur Ctrl+C.

Une fois que vous avez arrêté les applications, les conteneurs Docker doivent aussi arrêter de s'exécuter. Vérifiez cela avec docker ps:

```
$ docker ps --all
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS
              PORTS
73d515d99d2a      nginx:latest       "nginx -g 'daemon of  21 minutes ago   Exited
(0) 11 minutes ago
7061c76220de      php:fpm           "php-fpm"          21 minutes ago   Exited
(0) 11 minutes ago
elasticbeanstalk_nginxproxy_1
elasticbeanstalk_phpapp_1
```

L'option all montre des conteneurs arrêtés (si vous omettez cette option, le résultat sera vide). Dans l'exemple ci-dessus, Docker montre que les deux conteneurs ont terminé avec un état 0 (sans erreur).

Si vous avez terminé d'utiliser les commandes locales de l'interface de ligne de commande EB et Docker, vous pouvez supprimer les images Docker de votre ordinateur local pour économiser de l'espace.

Pour supprimer les images Docker de votre ordinateur local

1. Affichez les images que vous avez téléchargées à l'aide de docker images:

```
$ docker images
REPOSITORY          TAG      IMAGE ID      CREATED        VIRTUAL
php                 fpm      68bc5150cffc  1 hour ago    414.1
nginx               latest   637d3b2f5fb5  1 hour ago    93.44
MB
```

2. Supprimez les deux conteneurs Docker avec docker rm:

```
$ docker rm 73d515d99d2a 7061c76220de
73d515d99d2a
7061c76220de
```

3. Supprimez les images avec docker rmi:

```
$ docker rmi 68bc5150cffc 637d3b2f5fb5
Untagged: php:fpm
Deleted: 68bc5150cffc0526c66b92265c3ed8f2ea50f3c71d266aa655b7a4d20c3587b0
Untagged: nginx:latest
Deleted: 637d3b2f5fb5c4f70895b77a9e76751a6e7670f4ef27a159dad49235f4fe61e0
```

# Création et déploiement d'applications Go sur Elastic Beanstalk

## Rubriques

- [Mise en route avec Go sur Elastic Beanstalk \(p. 98\)](#)
- [Configuration de votre environnement de développement Go \(p. 100\)](#)
- [Utilisation de la plateforme Go Elastic Beanstalk \(p. 101\)](#)
- [Déploiement d'une application Go sur Elastic Beanstalk \(p. 106\)](#)

AWS Elastic Beanstalk pour Go permet de déployer, de gérer et de mettre à l'échelle facilement vos applications web Go à l'aide d'Amazon Web Services. Elastic Beanstalk pour Go est accessible à toute personne développant ou hébergeant une application web à l'aide de Go. Ce chapitre fournit, étape par étape, des instructions permettant le déploiement de votre application web dans Elastic Beanstalk.

Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser la CLI EB pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, AWS CLI ou les API.

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

## Mise en route avec Go sur Elastic Beanstalk

Pour commencer à utiliser des applications Go sur AWS Elastic Beanstalk, il vous suffit de charger la [solution groupée \(p. 415\)](#) source d'une application comme première version de l'application et de la déployer dans un environnement. Lorsque vous créez un environnement, Elastic Beanstalk alloue toutes les ressources AWS nécessaires pour exécuter une application web hautement évolutive.

## Lancement d'un environnement avec un exemple d'application Go

Elastic Beanstalk fournit des exemples d'application d'une seule page pour chaque plateforme. Elastic Beanstalk fournit également des exemples plus complexes qui illustrent l'utilisation de ressources AWS supplémentaires, comme Amazon RDS, des fonctionnalités propres aux différents langages ou plateformes, et des API.

### Samples

Configurations prises en charge	Type d'environnement	Solution groupée source	Description
Go	Serveur web	<a href="#">go.zip</a>	Application avec page unique.

Téléchargez l'exemple d'application et déployez-le dans Elastic Beanstalk en procédant comme suit :

Pour lancer un environnement avec un exemple d'application (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le volet de navigation, choisissez Applications, puis le nom d'une application existante dans la liste ou créez-en un (p. 406).
3. Sur la page de présentation de l'application, choisissez Create a new environment (Créer un nouvel environnement).

Environment name	Health	Date created	Last modified	URL	Platform
GettingStartedApp-env	<span>OK</span>	2020-01-28 12:05:50 UTC-0800	2020-01-30 15:02:35 UTC-0800	GettingStartedApp-env.bn7dx222kw.us-east-2.elasticbeanstalk.com	Tomcat running Linux
GettingStartedApp-Worker	<span>OK</span>	2020-01-28 16:34:29 UTC-0800	2020-01-28 16:38:20 UTC-0800	GettingStartedApp-Worker.bn7dx222kw.us-east-2.elasticbeanstalk.com	IIS 10.0 Windows

4. Ensuite, pour le niveau d'environnement, choisissez l'environnement de serveur web ou le niveau d'environnement de travail (p. 14). Vous ne pouvez pas modifier le niveau d'un environnement après sa création.

#### Note

La plateforme .NET sur Windows Server (p. 192) ne prend pas en charge le niveau d'environnement worker.

**Select environment tier**

AWS Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web server environments are for applications that listen for and then process HTTP requests, typically over port 80. Workers are specialized applications that perform a background processing task that listens for messages on an Amazon SQS queue. Worker applications post the results of their processing back to the application by using HTTP.

**Web server environment**  
Run a website, web application, or web API that serves HTTP requests.  
[Learn more](#)

**Worker environment**  
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule.  
[Learn more](#)

5. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

### Note

Elastic Beanstalk prend en charge plusieurs [versions \(p. 31\)](#) pour la plupart des plateformes répertoriées. Par défaut, la console sélectionne la version recommandée pour la plateforme et la branche de plateforme que vous choisissez. Si votre application nécessite une version différente, vous pouvez la sélectionner ici, ou choisir Configurer plus d'options, selon les instructions de l'étape 7. Pour plus d'informations sur les versions de plateforme prises en charge, consultez [the section called “Plateformes prises en charge” \(p. 31\)](#).

6. Pour l'option Code de l'application, choisissez Exemple d'application.
7. Pour personnaliser davantage votre environnement, choisissez Configurer plus d'options. Les options suivantes peuvent être définies uniquement lors de la création de l'environnement :
  - Nom de l'environnement
  - Nom de domaine
  - Version de plateforme
  - VPC
  - Palier

Vous pouvez modifier les paramètres suivants après la création de l'environnement, mais ils requièrent la mise en œuvre de nouvelles instances ou d'autres ressources et leur application peut prendre du temps :

- Type d'instance, volume racine, paire de clés et rôle AWS Identity and Access Management (IAM)
- Base de données interne Amazon RDS
- Equilibreur de charge

Pour de plus amples informations sur tous les paramètres disponibles, veuillez consulter [Assistant de création d'un environnement \(p. 442\)](#).

8. Choisissez Create environment.

## Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2.

Une fois que vous avez déployé un exemple d'application ou deux, et que vous êtes prêt à développer et exécuter les applications Go en local, consultez [Configuration de votre environnement de développement Go \(p. 100\)](#).

## Configuration de votre environnement de développement Go

Configurez un environnement de développement Go pour tester votre application en local avant de la déployer dans AWS Elastic Beanstalk. Cette rubrique décrit les étapes de configuration de votre environnement de développement et fournit des liens vers les pages d'installation relatives aux outils utilisables.

Pour accéder aux outils et aux étapes de configuration courants qui s'appliquent à toutes les langues, veuillez consulter [Configuration de votre machine de développement pour une utilisation avec Elastic Beanstalk \(p. 1024\)](#).

## Installation de Go

Pour exécuter des applications Go localement, installez Go. Si vous n'avez pas besoin d'une version spécifique, procurez-vous la dernière version prise en charge par Elastic Beanstalk. Pour obtenir la liste des versions prises en charge, accédez à [Go](#) dans le document Plateformes AWS Elastic Beanstalk.

Téléchargez Go à partir du lien <https://golang.org/doc/install>.

## Installation du kit SDK AWS pour Go

Si vous avez besoin de gérer des ressources AWS dans votre application, installez le kit SDK AWS pour Go à l'aide de la commande suivante.

```
$ go get github.com/aws/aws-sdk-go
```

Pour plus d'informations, veuillez consulter [Kit SDK AWS pour Go](#).

## Utilisation de la plateforme Go Elastic Beanstalk

### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [the section called "Mise à niveau vers Amazon Linux 2" \(p. 508\)](#).

Vous pouvez utiliser AWS Elastic Beanstalk pour exécuter, développer et configurer des applications basées sur Go. Pour des applications Go simples, il existe deux façons de déployer votre application :

- Fournissez un groupe source avec un fichier source à la racine appelé `application.go` qui contient le package principal pour votre application. Elastic Beanstalk crée le binaire à l'aide de la commande suivante :

```
go build -o bin/application application.go
```

Une fois que l'application est créée, Elastic Beanstalk la démarre sur le port 5000.

- Fournissez un groupe source avec un fichier binaire appelé `application`. Le fichier binaire peut être situé soit à la racine du bundle de fichiers source, soit dans le répertoire `bin/` du bundle de fichiers source. Si vous placez le fichier binaire `application` dans les deux emplacements, Elastic Beanstalk utilise le fichier du répertoire `bin/`.

Elastic Beanstalk lance cette application sur le port 5000.

Dans les deux cas, avec Go 1.11 ou version ultérieure, vous pouvez également fournir des exigences de module dans un fichier appelé `go.mod`. Pour plus d'informations, consultez [Migrer vers des modules Go](#) dans le blog Go.

Pour des applications Go plus complexes, il existe deux façons de déployer votre application :

- Fournissez un bundle source qui inclut vos fichiers sources d'application, avec un [Buildfile \(p. 105\)](#) et un [Procfile \(p. 104\)](#). Le Buildfile inclut une commande pour générer l'application et le Procfile inclut des instructions pour exécuter l'application.
- Fournissez un groupe source qui inclut vos fichiers binaires d'application, avec un Procfile. Le Procfile inclut des instructions pour exécuter l'application.

La plateforme Go inclut un serveur proxy pour servir les ressources statiques et transférer le trafic vers votre application. Vous pouvez [étendre ou remplacer la configuration du serveur proxy par défaut \(p. 105\)](#) pour les scénarios avancés.

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter [the section called “Extension des plateformes Linux” \(p. 34\).](#)

## Configuration de votre environnement Go

Les paramètres de la plateforme Go vous permettent d'affiner le comportement de vos instances Amazon EC2. Vous pouvez modifier la configuration des instances Amazon EC2 de l'environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour permettre la rotation des journaux sur Amazon S3 et configurer des variables que votre application peut lire à partir de l'environnement.

Pour configurer votre environnement Go dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

## Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

## Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour de plus amples informations sur la configuration des fichiers statiques à l'aide de la console Elastic Beanstalk, veuillez consulter [the section called “Fichiers statiques” \(p. 790\).](#)

## Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

Dans l'environnement Go en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de la fonction `os.Getenv`. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
endpoint := os.Getenv("API_ENDPOINT")
```

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Espaces de noms de la configuration Go

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

La plate-forme Go ne définit aucun espace de noms spécifique à la plate-forme. Vous pouvez configurer le proxy pour qu'il traite les fichiers statiques à l'aide de l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles`. Pour plus de détails et un exemple, reportez-vous à la section [the section called "Fichiers statiques" \(p. 790\)](#).

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Plateforme Go d'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Go utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

### Espaces de noms de configuration Go

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

La plateforme Go de l'AMI Amazon Linux prend en charge l'espace de noms de configuration spécifique à la plateforme en plus des [espaces de noms pris en charge par toutes les plateformes \(p. 679\)](#). L'espace de noms `aws:elasticbeanstalk:container:golang:staticfiles` vous permet de définir des options qui mappe des chemins d'accès sur votre application web vers des dossiers dans le groupe source de votre application incluant le contenu statique.

Par exemple, ce [fichier de configuration \(p. 737\)](#) indique au serveur proxy de servir les fichiers dans le dossier `staticimages` du chemin `/images` :

Example `.ebextensions/go-settings.config`

```
option_settings:
  aws:elasticbeanstalk:container:golang:staticfiles:
    /html: statichtml
    /images: staticimages
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide

de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Configuration du processus de l'application avec un Procfile

Pour spécifier des commandes personnalisées pour démarrer une application Go, incluez un fichier nommé `Procfile` à la racine de votre groupe source.

Pour plus d'informations sur l'écriture et l'utilisation d'un `Procfile`, développez la section `Buildfile` et `Procfile` dans [the section called "Extension des plateformes Linux" \(p. 34\)](#).

### Example Procfile

```
web: bin/server
queue_process: bin/queue_processor
foo: bin/fooapp
```

Vous devez appeler l'application principale `web` et la répertorier comme première commande dans votre fichier `Procfile`. Elastic Beanstalk expose la principale application `web` sur l'URL racine de l'environnement ; par exemple, <http://my-go-env.elasticbeanstalk.com>.

Elastic Beanstalk exécute également n'importe quelle application dont le nom n'a pas le préfixe `web_`, mais ces applications ne sont pas disponibles à l'extérieur de votre instance.

Elastic Beanstalk s'attend à ce que les processus s'exécutant à partir du fichier `Procfile` le fassent en continu. Elastic Beanstalk surveille ces applications et redémarre tout processus qui s'arrête. Pour les processus de courte durée, utilisez une commande [Buildfile \(p. 105\)](#).

## Utilisation d'un Procfile sur AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Go utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

### Transmission de port

Elastic Beanstalk configure le proxy nginx pour transmettre les demandes à votre application sur le numéro de port indiqué dans la [propriété d'environnement \(p. 102\)](#) `PORT` de votre application. Votre application doit toujours écouter sur ce port. Vous pouvez accéder à cette variable dans votre application en appelant la méthode `os.Getenv("PORT")`.

Elastic Beanstalk utilise le numéro de port spécifié dans la propriété d'environnement `PORT` pour le port de la première application dans le fichier `Procfile`. Il augmente ensuite le numéro de port de 100 pour chaque application ultérieure dans le fichier `Procfile`. Si la propriété d'environnement `PORT` n'est pas définie, Elastic Beanstalk utilise 5000 pour le port initial.

Dans l'exemple précédent, la propriété d'environnement `PORT` pour l'application `web` est 5000, l'application `queue_process` est 5100 et l'application `foo` est 5200.

Vous pouvez spécifier le port initial en définissant l'option `PORT` avec l'espace de noms [aws:elasticbeanstalk:application:environment \(p. 697\)](#), comme illustré dans l'exemple suivant.

```
option_settings:
  - namespace: aws:elasticbeanstalk:application:environment
    option_name: PORT
    value: <first_port_number>
```

Pour de plus amples informations sur la définition des propriétés d'environnement pour votre application, veuillez consulter [Paramètres d'option \(p. 738\)](#).

## Développement d'un exécutable sur le serveur avec un Buildfile

Pour spécifier une commande personnalisée de configuration et de build pour votre application Go, incluez un fichier nommé `Buildfile` à la racine de votre groupe source. Le nom de fichier est sensible à la casse. Utilisez le format suivant pour le `Buildfile`:

```
<process_name>: <command>
```

La commande dans votre `Buildfile` doit correspondre à l'expression régulière suivante : `^[A-Za-zA-9_]+:\s*.*$`.

Elastic Beanstalk ne surveille pas l'application exécutée avec un fichier `Buildfile`. Utilisez un `Buildfile` pour les commandes qui s'exécutent pendant de courtes durées et s'arrêtent après avoir terminé leurs tâches. Pour les processus d'applications longue durée qui ne doivent pas se fermer, utilisez plutôt le [Procfile \(p. 104\)](#).

Dans l'exemple suivant d'un `Buildfile`, `build.sh` est un script shell qui se trouve à la racine du bundle de fichiers source :

```
make: ./build.sh
```

Tous les chemins d'accès dans le `Buildfile` sont par rapport à la racine du groupe source. Si vous savez à l'avance où les fichiers résident sur l'instance, vous pouvez inclure des chemins d'accès absolus dans le `Buildfile`.

## Configuration du proxy inverse

Elastic Beanstalk utilise nginx comme proxy inverse pour mapper votre application à votre équilibrEUR de charge Elastic Load Balancing sur le port 80. Elastic Beanstalk fournit une configuration nginx par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Par défaut, Elastic Beanstalk configure le serveur proxy nginx pour transmettre les demandes à votre application sur le port 5000. Vous pouvez remplacer le port par défaut en définissant la [propriété d'environnement \(p. 102\)](#) `PORT` sur le port que votre application écoute.

### Note

Le port que votre application écoute n'affecte pas le port que le serveur nginx écoute pour recevoir des demandes de l'équilibrEUR de charge.

Toutes les plateformes Amazon Linux 2 prennent en charge une configuration de proxy uniforme. Pour obtenir des détails sur la configuration du serveur proxy sur les nouvelles versions de plateforme Amazon Corretto qui exécutent Amazon Linux 2, développez la section Configuration du proxy inverse dans [the section called "Extension des plateformes Linux" \(p. 34\)](#).

## Configuration du proxy sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Go utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations de cette section.

### Extension et remplacement de la configuration du proxy par défaut

Elastic Beanstalk utilise nginx comme proxy inverse pour mapper votre application vers votre équilibrEUR de charge sur le port 80. Si vous souhaitez fournir votre propre configuration nginx, vous pouvez remplacer la configuration par défaut fournie par Elastic Beanstalk en incluant le fichier `.ebextensions/nginx/`

`nginx.conf` dans votre bundle de fichiers source. Si ce fichier est présent, Elastic Beanstalk utilise à la place le fichier de configuration par défaut nginx.

Si vous souhaitez inclure des directives en plus de celles du bloc `nginx.conf http`, vous pouvez également fournir des fichiers de configuration supplémentaires dans le répertoire `.ebextensions/nginx/conf.d/` de votre bundle de fichiers source. Tous les fichiers de ce répertoire doivent avoir l'extension `.conf`.

Pour tirer parti des fonctionnalités fournies par Elastic Beanstalk, comme [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#), mappages d'application automatiques et fichiers statiques, vous devez inclure la ligne suivante dans le bloc `server` de votre fichier de configuration nginx :

```
include conf.d/elasticbeanstalk/*.conf;
```

## Déploiement d'une application Go sur Elastic Beanstalk

Ce didacticiel vous guide tout au long du processus de génération d'une application Go et de son déploiement dans un environnement AWS Elastic Beanstalk.

### Sections

- [Prerequisites \(p. 106\)](#)
- [Création d'une application Go \(p. 106\)](#)
- [Déploiement de votre application Go avec l'interface de ligne de commande \(CLI\) EB \(p. 107\)](#)
- [Nettoyage \(p. 109\)](#)

## Prerequisites

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

Ce tutoriel utilise également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter [Installation de l'interface de ligne de commande EB \(p. 1028\)](#) et [Configuration de l'interface de ligne de commande EB \(p. 1036\)](#).

## Création d'une application Go

Créez un répertoire de projet.

```
~$ mkdir eb-go
~$ cd eb-go
```

Créez ensuite une application qui vous allez déployer à l'aide d'Elastic Beanstalk. Nous allons créer un service web RESTful « Hello World ».

Cet exemple affiche un message d'accueil personnalisé qui varie selon le chemin d'accès utilisé pour accéder au service.

Créez un fichier texte dans le répertoire nommé `application.go` avec le contenu suivant :

Example `~/eb-go/application.go`

```
package main

import (
    "fmt"
    "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
    if r.URL.Path == "/" {
        fmt.Fprintf(w, "Hello World! Append a name to the URL to say hello. For example, use %s/Mary to say hello to Mary.", r.Host)
    } else {
        fmt.Fprintf(w, "Hello, %s!", r.URL.Path[1:])
    }
}

func main() {
    http.HandleFunc("/", handler)
    http.ListenAndServe(":5000", nil)
}
```

## Déploiement de votre application Go avec l'interface de ligne de commande (CLI) EB

Créez ensuite votre environnement d'applications et déployez votre application configurée avec Elastic Beanstalk.

Création d'un environnement et déploiement de votre application Go

1. Initialisez votre référentiel de la CLI EB avec la commande `eb init` :

```
~/eb-go$ eb init -p go-go-tutorial --region us-east-2
Application go-tutorial has been created.
```

Cette commande crée une application appelée `go-tutorial` et configure votre référentiel local de façon à créer des environnements avec la dernière version de plateforme Go.

2. (facultatif) Exécutez à nouveau la commande `eb init` pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l'instance EC2 qui exécute votre application.

```
~/eb-go$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
```

2) [ Create new KeyPair ]

Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez eb init -i.

3. Créez un environnement et déployez-y votre application avec eb create. Elastic Beanstalk génère automatiquement un fichier binaire pour votre application et le démarre sur le port 5000.

```
~/eb-go$ eb create go-env
```

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

#### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.

- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

Lorsque le processus de création de l'environnement est terminé, ouvrez votre site web avec la commande : eb open.

```
~/eb-go$ eb open
```

Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application.

Si vous ne voyez pas votre application en cours d'exécution ou si vous obtenez un message d'erreur, consultez [Déploiements \(p. 1142\)](#) afin d'obtenir de l'aide concernant la façon de déterminer la cause de l'erreur.

Si vous voyez effectivement votre application en cours d'exécution, alors félicitations, cela signifie que vous avez déployé votre première application Go avec Elastic Beanstalk !

## Nettoyage

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Ou, avec l'interface de ligne de commande (CLI) EB, procédez comme suit.

```
~/eb-go$ eb terminate
```

# Création et déploiement d'applications Java sur Elastic Beanstalk

AWS Elastic Beanstalk prend en charge deux plateformes pour les applications Java.

- Apache Tomcat : plateforme basée sur Apache Tomcat, conteneur web open source pour les applications qui utilisent des servlets Java et des pages JSP (Java Server Pages) pour traiter les demandes HTTP. Tomcat facilite le développement d'applications web en fournissant une configuration de sécurité multithread déclarative, ainsi qu'une personnalisation étendue. Elastic Beanstalk possède des branches de plateforme pour chacune des versions majeures actuelles de Tomcat. Pour plus d'informations, consultez [Plateforme Tomcat \(p. 118\)](#).
- Java SE : plateforme pour les applications qui n'utilisent pas de conteneur Web, ou qui en utilisent un autre que Tomcat, comme Jetty ou GlassFish. Vous pouvez inclure n'importe quelle bibliothèque Java Archives (JAR) utilisée par votre application dans le groupe source que vous déployez sur Elastic Beanstalk. Pour plus d'informations, consultez [Plateforme Java SE \(p. 129\)](#).

Les branches récentes des plateformes Tomcat et Java SE sont basées sur Amazon Linux 2 et utilisent Corretto, la distribution AWS Java SE. Les noms de ces branches dans les listes de plateformes incluent le mot Corretto au lieu de Java, par exemple `Corretto 11 with Tomcat 8.5`.

Pour obtenir la liste des versions actuelles de la plateforme, veuillez consulter [Tomcat](#) et [Java SE](#) dans le guide Plateformes AWS Elastic Beanstalk.

AWS fournit plusieurs outils pour travailler avec Java et Elastic Beanstalk. Quelle que soit la version de plateforme que vous choisissez, vous pouvez utiliser [AWS SDK pour Java \(p. 117\)](#) pour utiliser d'autres services AWS dans votre application Java. Le kit AWS SDK pour Java est un ensemble de bibliothèques qui vous permet d'utiliser des API AWS depuis votre code d'application sans écrire les appels HTTP bruts de bout en bout.

Si vous utilisez l'environnement de développement intégré (IDE) Eclipse pour développer votre application Java, vous pouvez également obtenir [AWS Toolkit for Eclipse \(p. 142\)](#). AWS Toolkit for Eclipse est un plug-in open source qui vous permet de gérer les ressources AWS, y compris les environnements et les applications Elastic Beanstalk, à partir de l'IDE Eclipse.

Si la ligne de commande vous convient mieux, installez l'[interface de ligne de commande \(CLI\) Elastic Beanstalk \(p. 1027\)](#) et utilisez-la pour créer, surveiller et gérer vos environnements à partir de la ligne de commande. Si vous exécutez plusieurs environnements pour votre application, l'interface de ligne de commande (CLI) EB s'intègre avec Git pour vous permettre d'associer chacun de vos environnements à une branche Git différente.

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

## Rubriques

- [Démarrer avec Java sur Elastic Beanstalk \(p. 111\)](#)
- [Configuration de votre environnement de développement Java \(p. 116\)](#)
- [Utilisation de la plateforme Elastic Beanstalk Tomcat \(p. 118\)](#)
- [Utilisation de la plateforme Java SE Elastic Beanstalk \(p. 129\)](#)
- [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Java \(p. 136\)](#)
- [Utilisation d'AWS Toolkit for Eclipse \(p. 142\)](#)
- [Resources \(p. 157\)](#)

## Démarrer avec Java sur Elastic Beanstalk

Pour commencer à utiliser des applications Java sur AWS Elastic Beanstalk, il vous suffit de charger le [bundle source \(p. 415\)](#) d'une application en tant que première version de l'application et de la déployer dans un environnement. Lorsque vous créez un environnement, Elastic Beanstalk alloue toutes les ressources AWS nécessaires pour exécuter une application web évolutive.

## Lancement d'un environnement avec un exemple d'application Java

Elastic Beanstalk fournit des exemples d'applications à page unique pour chaque plateforme, ainsi que des exemples plus complexes qui illustrent l'utilisation de ressources AWS supplémentaires, comme Amazon RDS, des fonctions propres aux différents langages ou plateformes, et des API.

Les exemples de page unique correspondent au même code que celui que vous obtenez lorsque vous créez un environnement sans fournir votre propre code source. Les exemples plus complexes sont hébergés sur GitHub. Vous pouvez être amené à les compiler ou à les créer avant de les déployer dans un environnement Elastic Beanstalk.

### Samples

Nom	Versions prises en charge	Type d'environnement	Source	Description
Tomcat (toutes les branches de plateforme Tomcat avec Corretto unique)	Toutes les branches de plateforme Tomcat (page unique)	Server web Nœuds	Tomcat.zip	<p>Application web Tomcat avec une page unique (<code>index.jsp</code>) configurée pour s'afficher à la racine du site web.</p> <p>Pour les <a href="#">environnements de travail (p. 521)</a>, cet exemple inclut un fichier <code>cron.yaml</code> qui configure une tâche planifiée afin d'appeler <code>scheduled.jsp</code> une fois par minute. Lorsque <code>scheduled.jsp</code> est appelé, il écrit dans un fichier journal à l'emplacement <code>/tmp/sample-app.log</code>. Enfin, un fichier de configuration est inclus dans <code>.ebextensions</code> et copie les journaux depuis <code>/tmp/</code> vers les emplacements lus par Elastic Beanstalk lorsque vous demandez les journaux de l'environnement.</p> <p>Si vous <a href="#">activez l'intégration à X-Ray (p. 638)</a> dans un environnement exécutant cet exemple, l'application affiche un contenu supplémentaire concernant X-Ray et fournit une option permettant de générer des informations de débogage que vous pouvez afficher dans la console X-Ray.</p>

Nom	Versions prises en charge	Type	Source d'environnement	Description
Corretto (page unique)	Corretto 8	Serveur web	<a href="#">corretto.zip</a>	<p>Application Corretto avec fichiers de configuration <code>Buildfile</code> et <code>Procfile</code>.</p> <p>Si vous <a href="#">activez l'intégration à X-Ray (p. 638)</a> dans un environnement exécutant cet exemple, l'application affiche un contenu supplémentaire concernant X-Ray et fournit une option permettant de générer des informations de débogage que vous pouvez afficher dans la console X-Ray.</p>

Nom	Versions prises en charge	Type d'environnement	Source	Description
Scorekeep	Java 8	Serveur web	Ozonez le référentiel sur GitHub.com	<p>Scorekeep est une API web RESTful qui utilise l'infrastructure Spring pour fournir une interface permettant de créer et de gérer des utilisateurs, des sessions et des jeux. Cette API est regroupée avec une application web Angular 1.5 qui utilise l'API sur HTTP.</p> <p>L'application utilise des fonctionnalités de la plateforme Java SE pour télécharger des dépendances et des instances intégrées, ce qui réduit la taille du bundle de fichiers source. L'application inclut également des fichiers de configuration nginx qui remplacent la configuration par défaut pour servir l'application web frontale de manière statique sur le port 80 via le proxy et acheminer les requêtes vers des chemins sous /api vers l'API s'exécutant sur localhost:5000.</p> <p>Scorekeep inclut également une branche xray qui montre comment instrumenter une application Java à utiliser avec AWS X-Ray. La branche montre l'instrumentation des requêtes HTTP entrantes avec un filtre de servlets, l'instrumentation automatique et manuelle du client SDK AWS, la configuration de l'enregistreur, et l'instrumentation des requêtes HTTP sortantes et des clients SQL.</p> <p>Pour obtenir des instructions, consultez le fichier readme ou utilisez le <a href="#">Tutorial de mise en route AWS X-Ray</a> pour essayer l'application avec X-Ray.</p>

Nom	Versions prises en charge	Type d'environnement	Source	Description
DoesTomcat 8 avec Java 8 it Have Snakes?		Serveur web	Créez le référentiel sur GitHub.com	Does it Have Snakes? est une application web Tomcat qui décrit l'utilisation des fichiers de configuration Elastic Beanstalk, Amazon RDS, JDBC, PostgreSQL, Servlets, JSP, Simple Tag Support, Tag Files, Log4J, Bootstrap et Jackson.  Le code source de ce projet comprend un script de construction minimal qui compile les servlets et les modèles dans des fichiers de classe, et qui regroupe les fichiers requis dans une archive web que vous pouvez déployer dans un environnement Elastic Beanstalk. Pour obtenir des instructions complètes, consultez le fichier readme dans le référentiel du projet.
LocustJava 8 Load Generator		Serveur web	Créez le référentiel sur GitHub.com	Application web qui vous permet de tester le chargement d'une autre application web exécutée dans un autre environnement Elastic Beanstalk. Elle décrit l'utilisation des fichiers Buildfile et Procfile, de DynamoDB et de Locust, un outil open source de test de charge.

Téléchargez l'un des exemples d'application et déployez-le dans Elastic Beanstalk en procédant comme suit :

Pour lancer un environnement avec un exemple d'application (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le volet de navigation, choisissez Applications, puis le nom d'une application existante dans la liste ou [créez-en un \(p. 406\)](#).
3. Sur la page de présentation de l'application, choisissez Create a new environment (Créer un nouvel environnement).

Environment name	Health	Date created	Last modified	URL	Platform
GettingStartedApp-env	<span>OK</span>	2020-01-28 12:06:50 UTC-0800	2020-01-30 15:02:35 UTC-0800	GettingStartedApp-env.bn7dx222kw.us-east-2.elasticbeanstalk.com	Tomcat running Linux
GettingStartedApp-Worker	<span>OK</span>	2020-01-28 16:34:29 UTC-0800	2020-01-28 16:38:20 UTC-0800	GettingStartedApp-Worker.bn7dx222kw.us-east-2.elasticbeanstalk.com	IIS 10.0 Windows

4. Ensuite, pour le niveau d'environnement, choisissez l'environnement de serveur web ou le [niveau d'environnement de travail \(p. 14\)](#). Vous ne pouvez pas modifier le niveau d'un environnement après sa création.

Note

La plateforme [.NET sur Windows Server \(p. 192\)](#) ne prend pas en charge le niveau d'environnement worker.

**Select environment tier**

AWS Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web server environments run a website, web application, or web API that serves HTTP requests. Worker environments run a background processing task that listens for messages on an Amazon SQS queue. Worker applications post those messages to your application by using HTTP.

**Web server environment**  
Run a website, web application, or web API that serves HTTP requests.  
[Learn more](#)

**Worker environment**  
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule.  
[Learn more](#)

5. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

Note

Elastic Beanstalk prend en charge plusieurs [versions \(p. 31\)](#) pour la plupart des plateformes répertoriées. Par défaut, la console sélectionne la version recommandée pour la plateforme et la branche de plateforme que vous choisissez. Si votre application nécessite

une version différente, vous pouvez la sélectionner ici, ou choisir Configurer plus d'options, selon les instructions de l'étape 7. Pour plus d'informations sur les versions de plateforme prises en charge, consultez [the section called “Plateformes prises en charge” \(p. 31\)](#).

6. Pour l'option Code de l'application, choisissez Exemple d'application.
7. Pour personnaliser davantage votre environnement, choisissez Configurer plus d'options. Les options suivantes peuvent être définies uniquement lors de la création de l'environnement :
  - Nom de l'environnement
  - Nom de domaine
  - Version de plateforme
  - VPC
  - Palier

Vous pouvez modifier les paramètres suivants après la création de l'environnement, mais ils requièrent la mise en œuvre de nouvelles instances ou d'autres ressources et leur application peut prendre du temps :

- Type d'instance, volume racine, paire de clés et rôle AWS Identity and Access Management (IAM)
- Base de données interne Amazon RDS
- Equilibreur de charge

Pour de plus amples informations sur tous les paramètres disponibles, veuillez consulter [Assistant de création d'un environnement \(p. 442\)](#).

8. Choisissez Create environment.

## Étapes suivantes

Une fois que vous disposez d'un environnement exécutant une application, vous pouvez [déployer une nouvelle version \(p. 476\)](#) de l'application ou une application totalement différente à tout moment. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2.

Une fois que vous avez déployé un ou deux exemples d'application et que vous êtes prêt à développer et à exécuter des applications Java localement, consultez la [section suivante \(p. 116\)](#) afin de configurer un environnement de développement Java avec tous les outils et les bibliothèques dont vous avez besoin.

## Configuration de votre environnement de développement Java

Configurez un environnement de développement Java pour tester votre application localement avant de le déployer dans AWS Elastic Beanstalk. Cette rubrique décrit les étapes de configuration de l'environnement de développement et des liens vers les pages d'installation pour des outils utiles.

Pour accéder aux outils et aux étapes de configuration courants qui s'appliquent à toutes les langues, veuillez consulter [Configuration de votre machine de développement \(p. 1024\)](#).

### Sections

- [Installation du kit de développement Java \(p. 117\)](#)
- [Installation d'un conteneur web \(p. 117\)](#)
- [Téléchargement de bibliothèques \(p. 117\)](#)
- [Installation du kit SDK AWS pour Java \(p. 117\)](#)

- Installation d'un IDE ou d'un éditeur de texte (p. 117)
- Installation d'AWS Toolkit for Eclipse (p. 118)

## Installation du kit de développement Java

Installez le kit de développement Java (JDK). Si vous n'avez pas de préférence, téléchargez la dernière version. Téléchargez le JDK sur [oracle.com](http://oracle.com)

Le JDK inclut le compilateur Java, qui vous permet de créer vos fichiers source dans des fichiers de classe qui peuvent être exécutés sur un serveur web Elastic Beanstalk.

## Installation d'un conteneur web

Si vous n'avez pas déjà une autre infrastructure ou un autre conteneur web, installez la version appropriée de Tomcat :

- Téléchargez Tomcat 8 (nécessite Java 7 ou version ultérieure)
- Téléchargez Tomcat 7 (nécessite Java 6 ou version ultérieure)

## Téléchargement de bibliothèques

Les plateformes Elastic Beanstalk incluent peu de bibliothèques par défaut. Téléchargez des bibliothèques que votre application va utiliser et enregistrez-les dans votre dossier de projet pour un déploiement dans le groupe source de votre application.

Si vous avez installé Tomcat localement, vous pouvez copier l'API servlet et les bibliothèques d'API Java Server Pages (JSP) du dossier d'installation. Si vous déployez sur une version de plateforme Tomcat, vous n'avez pas besoin d'inclure ces fichiers dans votre groupe source, mais vous devez les avoir dans votre classpath pour compiler toutes les classes qui les utilisent.

JUnit, Google Guava et Apache Commons offrent plusieurs bibliothèques utiles. Visitez leurs pages d'accueil pour en savoir plus :

- Téléchargez JUnit
- Téléchargez Google Guava
- Téléchargez Apache Commons

## Installation du kit SDK AWS pour Java

Si vous avez besoin gérer les ressources AWS à partir de votre application, installez le kit SDK AWS pour Java. Par exemple, avec le AWS SDK for Java, vous pouvez utiliser Amazon DynamoDB (DynamoDB) pour partager les états de session des applications Apache Tomcat sur plusieurs serveurs Web. Pour de plus amples informations, veuillez consulter la section relative à la [gestion de l'état de session Tomcat avec Amazon DynamoDB](#) dans la documentation du kit SDK AWS pour Java.

Visitez la [page d'accueil SDK AWS pour Java](#) pour de plus amples informations et des instructions d'installation.

## Installation d'un IDE ou d'un éditeur de texte

Les environnements de développement intégré (IDE) offrent un large éventail de fonctionnalités qui facilitent le développement d'applications. Si vous n'avez pas utilisé un IDE pour le développement Java, testez Eclipse et IntelliJ puis évaluez voir ce qui vous convient le mieux.

- Installation d'IDE Eclipse pour Java EE Developers
- Installation d'IntelliJ

#### Note

Un IDE peut ajouter des fichiers dans votre dossier de projet que vous pouvez ne pas souhaiter engager sur le contrôle de code source. Pour empêcher la validation de ces fichiers de contrôle de code source, utilisez `.gitignore` ou l'équivalent de votre outil de contrôle de source.

Si vous souhaitez simplement commencer le codage et que vous n'avez pas besoin de toutes les fonctionnalités d'un IDE, pensez à [installer Sublime Text](#).

## Installation d'AWS Toolkit for Eclipse

[AWS Toolkit for Eclipse \(p. 142\)](#) est un plug-in open source pour l'IDE Eclipse Java destiné aux développeurs afin de faciliter le développement, le débogage et le déploiement d'applications Java en utilisant AWS. Visitez la [page d'accueil AWS Toolkit for Eclipse](#) pour obtenir les instructions d'installation.

## Utilisation de la plateforme Elastic Beanstalk Tomcat

#### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [the section called "Mise à niveau vers Amazon Linux 2" \(p. 508\)](#).

La plateforme Tomcat AWS Elastic Beanstalk est un ensemble de [versions de plate-forme](#) pour les applications web Java qui peuvent s'exécuter dans un conteneur web Tomcat. Tomcat s'exécute derrière un serveur proxy nginx. Chaque configuration correspond à une version majeure de Tomcat, telle que Java 8 avec Tomcat 8.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour [modifier la configuration d'un environnement en cours d'exécution \(p. 671\)](#). Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des [configurations enregistrées \(p. 779\)](#) pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des [fichiers de configuration \(p. 737\)](#). Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

La plateforme Tomcat Elastic Beanstalk inclut un proxy inverse qui transmet les demandes à votre application. Vous pouvez utiliser les [options de configuration \(p. 121\)](#) pour configurer le serveur proxy de manière à traiter les ressources statiques à partir d'un dossier de votre code source afin de réduire la charge sur votre application. Pour les scénarios avancés, vous pouvez [inclure vos propres .conf fichiers \(p. 126\)](#) dans votre bundle de fichiers source afin d'étendre la configuration proxy Elastic Beanstalk ou de la remplacer complètement.

#### Note

Elastic Beanstalk prend en charge `nginx` (valeur par défaut) et `Apache HTTP Server` en tant que serveurs proxy sur la plateforme Tomcat. Si votre environnement Elastic Beanstalk Tomcat utilise une branche de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), vous avez

également la possibilité d'utiliser [Apache HTTP Server Version 2.2](#). Apache (dernière version) est la valeur par défaut sur ces anciennes branches de plateforme.

Vous devez compresser les applications Java dans un fichier WAR (Web application ARchive) avec une structure spécifique. Pour de plus amples informations sur la structure à respecter et sur sa liaison à la structure de votre répertoire de projet, veuillez consulter [Structuration de votre dossier de projet \(p. 123\)](#).

Pour exécuter plusieurs applications sur le même serveur web, vous pouvez [regrouper plusieurs fichiers WAR \(p. 122\)](#) dans un bundle source unique. Chaque application d'un bundle source multiple s'exécute soit sur le chemin d'accès racine (`ROOT.war` s'exécute dans `myapp.elasticbeanstalk.com/`), soit sur un chemin d'accès situé juste en dessous (`app2.war` s'exécute dans `myapp.elasticbeanstalk.com/app2/`), selon le nom du fichier WAR. Dans un bundle source WAR unique, l'application s'exécute toujours sur le chemin d'accès racine.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour de plus amples informations sur la priorité et les autres méthodes de modification des paramètres, veuillez consulter [Options de configuration \(p. 658\)](#).

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter [the section called "Extension des plateformes Linux" \(p. 34\)](#).

#### Rubriques

- [Configuration de votre environnement Tomcat \(p. 119\)](#)
- [Espaces de noms de la configuration Tomcat \(p. 121\)](#)
- [Plateforme Tomcat d'AMI Amazon Linux \(antérieure à Amazon Linux 2\) \(p. 122\)](#)
- [Création d'une offre groupée de plusieurs fichiers WAR pour les environnements Tomcat \(p. 122\)](#)
- [Structuration de votre dossier de projet \(p. 123\)](#)
- [Configuration du serveur proxy de votre environnement Tomcat \(p. 126\)](#)

## Configuration de votre environnement Tomcat

La plateforme Tomcat Elastic Beanstalk fournit quelques options propres à la plateforme en plus des options standard présentes sur toutes les plateformes. Ces options vous permettent de configurer la machine virtuelle Java (JVM) qui s'exécute sur les serveurs web de votre environnement et de définir les propriétés système qui fournissent les chaînes de configuration d'information à votre application.

Vous pouvez utiliser la console Elastic Beanstalk pour activer la rotation de journal sur Amazon S3 et configurer les variables que votre application peut lire depuis l'environnement.

Pour configurer votre environnement Tomcat dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

## Options du conteneur

Vous pouvez spécifier ces options spécifiques à la plateforme :

- Proxy server (Serveur proxy) – Serveur proxy à utiliser sur vos instances d'environnement. Le serveur nginx est utilisé par défaut.

## Options du conteneur JVM

La taille du tas sur la machine virtuelle Java (JVM) détermine combien d'objets peuvent être créés en mémoire par votre application avant un [nettoyage de la mémoire](#). Vous pouvez modifier les arguments Initial JVM Heap Size (argument -Xms) et Maximum JVM Heap Size (argument -Xmx). Plus la taille initiale du tas est importante, plus le nombre d'objets pouvant être créés avant le nettoyage de la mémoire sera élevé. Toutefois, cela signifie également que le récupérateur de mémoire mettra plus de temps pour compacter le tas. La taille de tas maximale indique la quantité maximale de mémoire que la machine virtuelle Java peut allouer lorsqu'elle augmente le tas dans le cadre d'une activité intensive.

### Note

La mémoire disponible dépend du type d'instance Amazon EC2. Pour de plus amples informations sur les types d'instances EC2 disponibles pour votre environnement Elastic Beanstalk, veuillez consulter [Types d'instances](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud pour les instances Linux.

La génération permanente est une section du tas de la machine virtuelle Java qui stocke les définitions de classe et les métadonnées associées. Pour modifier la taille de la génération permanente, saisissez la nouvelle taille dans le champ Maximum JVM PermGen Size (-XX:MaxPermSize argument). Ce paramètre s'applique uniquement à Java 7 et versions antérieures.

## Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

## Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour de plus amples informations sur la configuration des fichiers statiques à l'aide de la console Elastic Beanstalk, veuillez consulter [the section called "Fichiers statiques" \(p. 790\)](#).

## Propriétés de l'environnement

Dans la section Environment Properties (Propriétés de l'environnement), vous pouvez spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

La plateforme Tomcat définit une propriété d'espace réservée nommée `JDBC_CONNECTION_STRING` pour les environnements Tomcat, permettant de transmettre une chaîne de connexion à une base de données externe.

#### Note

Si vous attachez une instance de base de données RDS à votre environnement, construisez la chaîne de connexion JDBC dynamiquement à partir des propriétés d'environnement Amazon Relational Database Service (Amazon RDS) fournies par Elastic Beanstalk. Utilisez `JDBC_CONNECTION_STRING` uniquement pour les instances de base de données qui ne sont pas mises en service via Elastic Beanstalk.

Pour de plus amples informations sur l'utilisation d'Amazon RDS avec votre application Java, veuillez consulter [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Java \(p. 136\)](#).

Dans l'environnement Tomcat en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de `System.getProperty()`. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
String endpoint = System.getProperty("API_ENDPOINT");
```

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Espaces de noms de la configuration Tomcat

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

La plateforme Tomcat prend en charge les options des espaces de noms suivants en plus des [options prises en charge pour tous les environnements Elastic Beanstalk \(p. 679\)](#) :

- `aws:elasticbeanstalk:container:tomcat:jvmoptions` : modifier les paramètres de la JVM. Les options de cet espace de noms correspondent aux options de la console de gestion, comme suit :
  - `Xms` – JVM command line options (Options de ligne de commande de la JVM)
  - `JVM Options` – JVM command line options (Options de ligne de commande de la JVM)
- `aws:elasticbeanstalk:environment:proxy` : choisissez le serveur proxy de l'environnement.

L'exemple de fichier de configuration suivant illustre l'utilisation d'options de configuration spécifiques à Tomcat :

#### Example .ebextensions/tomcat-settings.config

```
option_settings:  
  aws:elasticbeanstalk:container:tomcat:jvmoptions:  
    Xms: 512m  
    JVM Options: '-Xmn128m'  
  aws:elasticbeanstalk:application:environment:  
    API_ENDPOINT: mywebapi.zkpexsjtmd.us-west-2.elasticbeanstalk.com  
  aws:elasticbeanstalk:environment:proxy:  
    ProxyServer: apache
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide

de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Plateforme Tomcat d'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Tomcat utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

### Espaces de noms de la configuration Tomcat

La plateforme Tomcat d'AMI Amazon Linux prend en charge des options supplémentaires dans les espaces de noms suivants :

- `aws:elasticbeanstalk:container:tomcat:jvmoptions` : en plus des options mentionnées précédemment sur cette page pour cet espace de noms, les anciennes versions de la plateforme AMI Amazon Linux prennent également en charge les éléments suivants :
  - `XX:MaxPermSize` – Maximum JVM permanent generation size (Taille maximum de génération permanente de la JVM)
  - `aws:elasticbeanstalk:environment:proxy` : en plus de choisir le serveur proxy, configurez également la compression de réponse.

L'exemple de fichier de configuration suivant illustre l'utilisation d'options de configuration de l'espace de noms du proxy.

### Example `.ebextensions/tomcat-settings.config`

```
option_settings:  
  aws:elasticbeanstalk:environment:proxy:  
    GzipCompression: 'true'  
    ProxyServer: nginx
```

### Inclusion des fichiers de configuration Elastic Beanstalk

Pour déployer les fichiers de configuration `.ebextensions`, incluez-les dans la source de votre application. S'il s'agit d'une seule application, ajoutez `.ebextensions` à un fichier WAR compressé en exécutant la commande suivante :

### Example

```
zip -ur your_application.war .ebextensions
```

Pour une application nécessitant plusieurs fichiers WAR, veuillez consulter [Création d'une offre groupée de plusieurs fichiers WAR pour les environnements Tomcat \(p. 122\)](#) pour de plus amples informations.

## Création d'une offre groupée de plusieurs fichiers WAR pour les environnements Tomcat

Si votre application web comporte plusieurs composants d'application web, vous pouvez simplifier les déploiements et réduire les coûts d'exploitation en exécutant les composants dans un environnement unique, au lieu d'exécuter un environnement distinct pour chaque composant. Cette stratégie est efficace pour les applications légères qui ne nécessitent pas beaucoup de ressources, ainsi que pour les environnements de développement et de test.

Pour déployer plusieurs applications web dans votre environnement, regroupez les fichiers WAR (web application archive) de chaque composant dans un même [bundle source \(p. 415\)](#).

Pour créer un bundle de fichiers source d'application qui contienne plusieurs fichiers WAR, organisez ces fichiers à l'aide de la structure suivante :

```
MyApplication.zip
### .ebextensions
### .platform
### foo.war
### bar.war
### ROOT.war
```

Lorsque vous déployez un groupe source contenant plusieurs fichiers WAR dans un environnement AWS Elastic Beanstalk, chaque application est accessible depuis un chemin d'accès différent du nom de domaine racine. L'exemple précédent comprend trois applications : `foo`, `bar` et `ROOT`. `ROOT.war` est un nom de fichier spécial qui demande à Elastic Beanstalk d'exécuter cette application au niveau du domaine racine, afin que les trois applications soient disponibles aux emplacements suivants : `http://MyApplication.elasticbeanstalk.com/`, `http://MyApplication.elasticbeanstalk.com/bar` et `http://MyApplication.elasticbeanstalk.com`.

Le bundle source peut inclure des fichiers WAR, un dossier `.ebextensions` facultatif et un dossier `.platform` facultatif. Pour de plus amples informations sur ces dossiers de configuration facultatifs, veuillez consulter [the section called "Extension des plateformes Linux" \(p. 34\)](#).

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk avec ce lien préconfiguré : [console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Plateforme, sélectionnez la plateforme et la branche de plateforme qui correspondent à la langue utilisée par votre application, ou la plateforme Docker pour les applications basées sur des conteneurs.
3. Pour Application code (Code d'application), choisissez Upload your code (Charger votre code).
4. Choisissez Local file (Fichier local), Choose file (Choisir un fichier), puis ouvrez le bundle source.
5. Choisissez Vérifier et lancer.
6. Vérifiez les paramètres disponibles et choisissez Créer une application.

Pour plus d'informations sur la création de groupes sources, consultez [Création d'une solution groupée d'application \(p. 415\)](#).

## Structuration de votre dossier de projet

Pour fonctionner lorsqu'ils sont déployés sur un serveur Tomcat, les fichiers WAR (web application archives) Java EE (Java Platform Enterprise Edition) compilés doivent être structurés selon certaines [directives](#). Votre répertoire de projet n'a pas besoin de respecter les mêmes normes, mais nous vous recommandons de le structurer de la même façon afin de simplifier la compilation et l'empaquetage. De plus, si vous structurez votre dossier de projet comme le contenu du fichier WAR, vous comprendrez mieux comment les fichiers sont liés et comment ils se comportent sur un serveur web.

Dans la hiérarchie recommandée ci-dessous, le code source de l'application web est placé dans un répertoire `src` afin de l'isoler du script de génération et du fichier WAR qu'il génère.

```
~/workspace/my-app/
|-- build.sh           - Build script that compiles classes and creates a WAR
```

```

|--- README.MD           - Readme file with information about your project, notes
|--- ROOT.war             - Source bundle artifact created by build.sh
`-- src                   - Source code folder
    |-- WEB-INF            - Folder for private supporting files
    |   |-- classes          - Compiled classes
    |   |-- lib               - JAR libraries
    |   |-- tags              - Tag files
    |   |-- tlds              - Tag Library Descriptor files
    |   `-- web.xml           - Deployment Descriptor
    |-- com                  - Uncompiled classes
    |-- css                  - Style sheets
    |-- images               - Image files
    |-- js                   - JavaScript files
    `-- default.jsp          - JSP (JavaServer Pages) webpage

```

Le contenu du dossier `src` correspond à ce que vous allez empaqueter et déployer sur le serveur, à l'exception du dossier `com`. Le dossier `com` contient vos classes non compilées (fichiers `.java`). Ils doivent être compilés et placés dans le répertoire `WEB-INF/classes` afin d'être accessibles à partir du code de votre application.

Le répertoire `WEB-INF` contient le code et les configurations qui ne sont pas publiquement traités sur le serveur web. Les autres dossiers situés à la racine du répertoire source (`css`, `images` et `js`) sont publiquement disponibles à l'emplacement correspondant sur le serveur web.

L'exemple suivant est identique au répertoire de projet précédent, sauf qu'il contient davantage de fichiers et de sous-répertoires. Cet exemple de projet inclut des balises simples, des classes de modèle et de support, et un fichier Java Server Pages (JSP) pour une ressource `record`. Il inclut également une feuille de style et JavaScript pour [Bootstrap](#), un fichier JSP par défaut et une page d'erreur pour les erreurs 404.

`WEB-INF/lib` inclut un fichier JAR (Java Archive) contenant le pilote JDBC (Java Database Connectivity) pour PostgreSQL. `WEB-INF/classes` est vide, car les fichiers de classe n'ont pas encore été compilés.

```

~/workspace/my-app/
|-- build.sh
|-- README.MD
|-- ROOT.war
`-- src
    |-- WEB-INF
    |   |-- classes
    |   |-- lib
    |   |   `-- postgresql-9.4-1201.jdbc4.jar
    |   |-- tags
    |   |   `-- header.tag
    |   |-- tlds
    |   |   `-- records.tld
    |   `-- web.xml
    |-- com
    |   `-- myapp
    |       |-- model
    |       |   `-- Record.java
    |       |-- web
    |       |   `-- ListRecords.java
    |-- css
    |   |-- bootstrap.min.css
    |   `-- myapp.css
    |-- images
    |   `-- myapp.png
    |-- js
    |   `-- bootstrap.min.js
    |-- 404.jsp
    |-- default.jsp
    `-- records.jsp

```

## Création d'un fichier WAR avec un script Shell

`build.sh` est un script shell très simple qui compile les classes Java, crée un fichier WAR et le copie dans le répertoire `webapps` de Tomcat pour les tests en local.

```
cd src
javac -d WEB-INF/classes com/myapp/model/Record.java
javac -classpath WEB-INF/lib/*:WEB-INF/classes -d WEB-INF/classes com/myapp/model/
Record.java
javac -classpath WEB-INF/lib/*:WEB-INF/classes -d WEB-INF/classes com/myapp/web/
ListRecords.java

jar -cvf ROOT.war *.jsp images css js WEB-INF
cp ROOT.war /Library/Tomcat/webapps
mv ROOT.war ../
```

Dans le fichier WAR, vous trouvez la même structure que celle du répertoire `src` figurant dans l'exemple précédent, à l'exception du dossier `src/com`. La commande `jar` crée automatiquement le fichier `META-INF/MANIFEST.MF`.

```
~/workspace/my-app/ROOT.war
|-- META-INF
|   '-- MANIFEST.MF
|-- WEB-INF
|   '-- classes
|       '-- com
|           '-- myapp
|               '-- model
|                   '-- Records.class
|               '-- web
|                   '-- ListRecords.class
|   '-- lib
|       '-- postgresql-9.4-1201.jdbc4.jar
|   '-- tags
|       '-- header.tag
|   '-- tlds
|       '-- records.tld
|   '-- web.xml
|-- css
|   '-- bootstrap.min.css
|   '-- myapp.css
|-- images
|   '-- myapp.png
|-- js
|   '-- bootstrap.min.js
|-- 404.jsp
|-- default.jsp
`-- records.jsp
```

## Utiliser `.gitignore`

Pour éviter de valider les fichiers de classe compilés et les fichiers WAR dans votre référentiel Git, ou de voir des messages les concernant lorsque vous exécutez les commandes Git, ajoutez les types de fichiers appropriés à un fichier nommé `.gitignore` dans votre dossier de projet.

```
~/workspace/myapp/.gitignore
```

```
*.zip
*.class
```

## Configuration du serveur proxy de votre environnement Tomcat

La plateforme Tomcat utilise [nginx](#) (par défaut) ou [Apache HTTP Server](#) comme proxy inverse pour relayer les requêtes du port 80 de l'instance vers votre conteneur web Tomcat écoutant sur le port 8080. Elastic Beanstalk fournit une configuration de proxy par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Toutes les plateformes Amazon Linux 2 prennent en charge une configuration de proxy uniforme. Pour obtenir des détails sur la configuration du serveur proxy sur les versions de plateforme Tomcat exécutant Amazon Linux 2, développez la section relative à la configuration de proxy inverse dans [the section called "Extension des plateformes Linux" \(p. 34\)](#).

### Configuration du proxy sur la plateforme Tomcat d'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Tomcat utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

#### Choisir un serveur proxy pour votre environnement Tomcat

Les versions de la plateforme Tomcat basées sur l'AMI Amazon Linux (antérieure à Amazon Linux 2) utilisent [Apache 2.4](#) pour le proxy par défaut. Vous pouvez choisir d'utiliser [Apache 2.2](#) ou [nginx](#) en incluant un [fichier de configuration \(p. 737\)](#) dans votre code source. L'exemple suivant configure Elastic Beanstalk pour utiliser nginx.

#### Example .ebextensions/nginx-proxy.config

```
option_settings:  
  aws:elasticbeanstalk:environment:proxy:  
    ProxyServer: nginx
```

#### Migration d'Apache 2.2 vers Apache 2.4

Si votre application a été développée pour [Apache 2.2](#), lisez cette section afin d'en savoir plus sur la migration vers [Apache 2.4](#).

À compter de la version 3.0.0 de la plateforme Tomcat, qui a été publiée avec la [mise à jour de la plateforme Java avec Tomcat le 24 mai 2018](#), Apache 2.4 est le proxy par défaut de la plateforme Tomcat. Les fichiers Apache 2.4 .conf sont pour la plupart, mais pas dans leur totalité, rétrocompatibles avec ceux d'Apache 2.2. Elastic Beanstalk inclut les fichiers .conf par défaut qui fonctionnent correctement avec chaque version Apache. Si votre application ne personnalise pas la configuration d'Apache, comme expliqué dans [Extension et remplacement de la configuration Apache par défaut \(p. 127\)](#), elle devrait migrer vers Apache 2.4 sans aucun problème.

Si votre application étend ou remplace la configuration d'Apache, il se peut que vous ayez à apporter certaines modifications pour migrer vers Apache 2.4. Pour plus d'informations, consultez [Mise à jour de la version 2.2 vers la version 2.4](#) sur le site The Apache Software Foundation. Temporairement, tant que vous n'avez pas réussi à migrer vers Apache 2.4, vous pouvez choisir d'utiliser Apache 2.2 avec votre application en incluant le [fichier de configuration \(p. 737\)](#) suivant dans votre code source.

#### Example .ebextensions/apache-legacy-proxy.config

```
option_settings:  
  aws:elasticbeanstalk:environment:proxy:  
    ProxyServer: apache/2.2
```

Pour une solution rapide, vous pouvez également sélectionner le serveur proxy dans la console Elastic Beanstalk.

Pour sélectionner le proxy dans votre environnement Tomcat dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

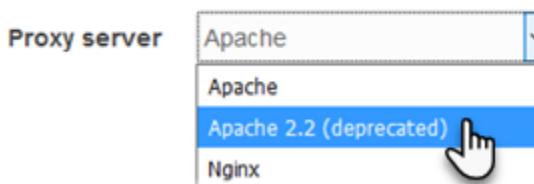
Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Pour Serveur proxy, choisissez Apache 2.2 (deprecated).
6. Choisissez Apply.

## Modify software

### Container Options

The following settings control container behavior and let you pass key-value pairs in as OS environment variables. [Learn more](#)



### Extension et remplacement de la configuration Apache par défaut

Vous pouvez étendre la configuration Apache Elastic Beanstalk par défaut avec vos fichiers de configuration supplémentaires. Sinon, vous pouvez remplacer complètement la configuration Apache Elastic Beanstalk par défaut.

Pour étendre la configuration Apache Elastic Beanstalk par défaut, ajoutez les fichiers de configuration .conf à un dossier nommé .ebextensions/httpd/conf.d dans le bundle de fichiers source de votre application. La configuration Apache Elastic Beanstalk par défaut inclut automatiquement les fichiers .conf dans ce dossier.

```
~/workspace/my-app/
|-- .ebextensions
|   -- httpd
|       -- conf.d
|           -- myconf.conf
|           -- ssl.conf
-- index.jsp
```

Par exemple, la configuration Apache 2.4 suivante ajoute un écouteur sur le port 5000 :

Example .ebextensions/httpd/conf.d/port5000.conf

```
listen 5000
```

```
<VirtualHost *:5000>
<Proxy *>
    Require all granted
</Proxy>
ProxyPass / http://localhost:8080/ retry=0
ProxyPassReverse / http://localhost:8080/
ProxyPreserveHost on

ErrorLog /var/log/httpd/elasticbeanstalk-error_log
</VirtualHost>
```

Pour remplacer complètement la configuration Apache Elastic Beanstalk par défaut, incluez une configuration dans votre bundle de fichiers source sur `.ebextensions/httpd/conf/httpd.conf`.

```
~/workspace/my-app/
|-- .ebextensions
|   '-- httpd
|       '-- conf
|           '-- httpd.conf
`-- index.jsp
```

Pour remplacer la configuration Apache Elastic Beanstalk par défaut, ajoutez les lignes suivantes à votre fichier `httpd.conf` afin d'extraire les configurations Elastic Beanstalk pour [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#), la compression des réponses et les fichiers statiques.

```
IncludeOptional conf.d/*.conf
IncludeOptional conf.d/elasticbeanstalk/*.conf
```

Si votre environnement utilise Apache 2.2 comme proxy, remplacez les directives `IncludeOptional` par `Include`. Pour plus d'informations sur le comportement de ces deux directives dans les deux versions d'Apache, consultez [Directive Include dans Apache 2.4](#), [Directive IncludeOptional dans Apache 2.4](#) et [Directive Include dans Apache 2.2](#).

#### Note

Pour remplacer l'écouteur par défaut sur le port 80, incluez un fichier nommé `00_application.conf` dans `.ebextensions/httpd/conf.d/elasticbeanstalk/` afin de remplacer la configuration Elastic Beanstalk.

Pour obtenir un exemple concret, veuillez consulter le fichier de configuration par défaut Elastic Beanstalk dans `/etc/httpd/conf/httpd.conf` sur une instance de votre environnement. Tous les fichiers du dossier `.ebextensions/httpd` de votre bundle de fichiers source sont copiés dans `/etc/httpd` au cours des déploiements.

#### Extension de la configuration nginx par défaut

Pour étendre la configuration nginx par défaut d'Elastic Beanstalk, ajoutez les fichiers de configuration `.conf` dans un dossier nommé `.ebextensions/nginx/conf.d/` dans le bundle de fichiers source de votre application. La configuration nginx d'Elastic Beanstalk inclut automatiquement les fichiers `.conf` dans ce dossier.

```
~/workspace/my-app/
|-- .ebextensions
|   '-- nginx
|       '-- conf.d
|           |-- elasticbeanstalk
|               |   '-- my-server-conf.conf
|               '-- my-http-conf.conf
`-- index.jsp
```

Les fichiers dotés de l'extension .conf du dossier conf.d sont inclus dans le bloc http de la configuration par défaut. Les fichiers du dossier conf.d/elasticbeanstalk sont inclus dans le bloc server au sein du bloc http.

Pour remplacer complètement la configuration nginx par défaut d'Elastic Beanstalk, incluez une configuration dans votre bundle de fichiers source à l'emplacement .ebextensions/nginx/nginx.conf.

```
~/workspace/my-app/  
|-- .ebextensions  
|  '-- nginx  
|    '-- nginx.conf  
`-- index.jsp
```

Pour remplacer la configuration nginx d'Elastic Beanstalk, ajoutez la ligne suivante au bloc server de votre configuration afin d'extraire les configurations Elastic Beanstalk pour l'écouteur du port 80, la compression des réponses et les fichiers statiques.

```
include conf.d/elasticbeanstalk/*.conf;
```

#### Note

Pour remplacer l'écouteur par défaut sur le port 80, incluez un fichier nommé 00\_application.conf dans .ebextensions/nginx/conf.d/elasticbeanstalk/ afin de remplacer la configuration Elastic Beanstalk.

Incluez également la ligne suivante dans le bloc http de votre configuration afin d'extraire les configurations Elastic Beanstalk pour [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#) et la journalisation.

```
include      conf.d/*.*.conf;
```

Pour obtenir un exemple concret, veuillez consulter le fichier de configuration par défaut Elastic Beanstalk dans /etc/nginx/nginx.conf sur une instance de votre environnement. Tous les fichiers du dossier .ebextensions/nginx de votre bundle de fichiers source sont copiés dans /etc/nginx au cours des déploiements.

## Utilisation de la plateforme Java SE Elastic Beanstalk

#### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [the section called “Mise à niveau vers Amazon Linux 2” \(p. 508\)](#).

La plateforme Java SE AWS Elastic Beanstalk est un ensemble de [versions de plateforme](#) pour les applications Web Java qui peuvent s'exécuter seules à partir d'un fichier JAR compilé. Vous pouvez compiler votre application en local ou télécharger le code source à l'aide d'un script de compilation pour le compiler sur l'instance. Les versions de plateforme Java SE sont regroupées en branches de plateforme, chacune correspondant à une version majeure de Java, par exemple Java 8 et Java 7.

#### Note

Elastic Beanstalk n'analyse pas le fichier JAR de votre application. Conservez les fichiers dont Elastic Beanstalk a besoin en dehors du fichier JAR. Par exemple, incluez le fichier cron.yaml

d'un [environnement de travail \(p. 521\)](#) à la racine du bundle source de votre application, en regard du fichier JAR.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour [modifier la configuration d'un environnement en cours d'exécution \(p. 671\)](#). Pour éviter de perdre la configuration de votre environnement en le résilient, vous pouvez utiliser des [configurations enregistrées \(p. 779\)](#) pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des [fichiers de configuration \(p. 737\)](#). Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

La plateforme Java SE Elastic Beanstalk comprend un serveur [nginx](#) qui agit comme un proxy inverse, traitant le contenu statique mis en cache et passant les demandes à votre application. La plateforme fournit également des options de configuration qui vous permettent de configurer le serveur proxy de manière à traiter les ressources statiques à partir d'un dossier de votre code source afin de réduire la charge sur votre application. Pour les scénarios avancés, vous pouvez [inclure vos propres fichiers .conf \(p. 134\)](#) dans votre bundle source afin d'étendre la configuration proxy d'Elastic Beanstalk ou de la remplacer complètement.

Si vous fournissez un seul fichier JAR pour votre source d'application (seul, pas dans un bundle de fichiers source), Elastic Beanstalk renomme votre fichier JAR en `application.jar`, puis l'exécute à l'aide de `java -jar application.jar`. Pour configurer les processus qui s'exécutent sur les instances de serveur dans votre environnement, incluez un [fichier Procfile \(p. 133\)](#) facultatif dans votre bundle de fichiers source. Un fichier `Procfile` est requis si vous avez plusieurs fichiers JAR à la racine de votre bundle de fichiers source, ou si vous souhaitez personnaliser la commande Java pour définir des options JVM.

Nous vous recommandons de toujours fournir un `Procfile` dans le bundle source avec votre application. De cette façon, vous contrôlez précisément les processus Elastic Beanstalk qui s'exécutent pour votre application et les arguments que ces processus reçoivent.

Pour compiler les classes Java et exécuter d'autres commandes de génération sur les instances EC2 dans votre environnement au moment du déploiement, notamment un [Buildfile \(p. 132\)](#) dans le bundle source de votre application. Un `Buildfile` vous permet de déployer votre code source en l'état et de le développer sur le serveur au lieu de compiler des JAR localement. La plateforme Java SE inclut des outils de développement courants pour vous permettre de développer sur le serveur.

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter [the section called "Extension des plateformes Linux" \(p. 34\)](#).

## Configuration de votre environnement Java SE

Les paramètres de la plateforme Java SE vous permettent d'affiner le comportement de vos instances Amazon EC2. Vous pouvez modifier la configuration des instances Amazon EC2 de l'environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour permettre la rotation des journaux sur Amazon S3 et configurer des variables que votre application peut lire à partir de l'environnement.

Pour configurer votre environnement Java SE dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

## Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

## Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour de plus amples informations sur la configuration des fichiers statiques à l'aide de la console Elastic Beanstalk, veuillez consulter [the section called “Fichiers statiques” \(p. 790\)](#).

## Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

Dans l'environnement Java SE en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de `System.getenv()`. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
String endpoint = System.getenv("API_ENDPOINT");
```

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Espaces de noms de la configuration Java SE

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

La plate-forme Java SE ne définit aucun espace de noms spécifique à la plate-forme. Vous pouvez configurer le proxy pour qu'il traite les fichiers statiques à l'aide de l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles`. Pour plus de détails et un exemple, reportez-vous à la section [the section called “Fichiers statiques” \(p. 790\)](#).

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## La plate-forme Java SE d'AMI Linux Amazon (antérieure à Amazon Linux 2)

Si votre environnement Java SE Elastic Beanstalk utilise une version de plate-forme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

### Espaces de noms de la configuration Java SE

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

La plate-forme Java SE prend en charge l'espace de noms de configuration spécifique à la plate-forme en plus des espaces de noms [pris en charge par toutes les plateformes \(p. 679\)](#). L'espace de noms `aws:elasticbeanstalk:container:java:staticfiles` vous permet de définir des options qui mappe des chemins d'accès sur votre application web vers des dossiers dans le groupe source de votre application incluant le contenu statique.

Par exemple, cet extrait [option\\_settings \(p. 738\)](#) définit deux options dans l'espace de noms de fichiers statiques. La première mappe le chemin d'accès `/public` dans un dossier nommé `public` et la deuxième mappe le chemin d'accès `/images` dans un dossier nommé `img` :

```
option_settings:  
  aws:elasticbeanstalk:container:java:staticfiles:  
    /html: statichtml  
    /images: staticimages
```

Les dossiers que vous mappez à l'aide de cet espace de noms doivent être de véritables dossiers à la racine de votre groupe source. Vous ne pouvez pas mapper un chemin d'accès à un dossier dans un fichier JAR.

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Développement de JAR sur le serveur avec un Buildfile

Vous pouvez créer les JAR et les fichiers de classe de votre application sur les instances EC2 dans votre environnement en appelant une commande Build depuis un fichier `Buildfile` dans votre groupe source.

Les commandes dans un `Buildfile` sont exécutées une seule fois et doivent se terminer à la fin, tandis que les commandes dans un [Procfile \(p. 133\)](#) sont censées s'exécuter pendant la durée de vie de l'application et seront redémarrées si elles s'arrêtent. Pour exécuter les JAR dans votre application, utilisez un `Procfile`.

Pour plus d'informations sur le placement et la syntaxe d'un `Buildfile`, développez la section `Buildfile` et `Procfile` dans [the section called "Extension des plateformes Linux" \(p. 34\)](#).

L'exemple de `Buildfile` ci-dessous exécute Apache Maven pour générer une application web à partir du code source. Pour un exemple d'application qui utilise cette fonctionnalité, consultez [Exemples d'applications web Java \(p. 111\)](#).

### Example Buildfile

```
build: mvn assembly:assembly -DdescriptorId=jar-with-dependencies
```

La plateforme Java SE inclut les outils de génération suivants que vous pouvez appeler à partir de votre script de build :

- `javac` : compilateur Java
- `ant` : Apache Ant
- `mvn` : Apache Maven
- `gradle` : Gradle

## Configuration du processus de l'application avec un Procfile

Si vous avez plus d'un fichier JAR à la racine du groupe source de votre application, vous devez inclure un fichier `Procfile` qui indique à Elastic Beanstalk quels JAR exécuter. Vous pouvez également inclure un fichier `Procfile` pour une seule application JAR pour configurer la machine virtuelle Java (JVM) qui exécute votre application.

Nous vous recommandons de toujours fournir un `Procfile` dans le bundle source avec votre application. De cette façon, vous contrôlez précisément les processus Elastic Beanstalk qui s'exécutent pour votre application et les arguments que ces processus reçoivent.

Pour plus d'informations sur l'écriture et l'utilisation d'un `Procfile`, développez la section Buildfile et `Procfile` dans [the section called “Extension des plateformes Linux” \(p. 34\)](#).

### Example Procfile

```
web: java -jar server.jar -Xms256m
cache: java -jar mycache.jar
web_foo: java -jar other.jar
```

La commande qui exécute le JAR principal dans votre application doit être appelée `web`, et il doit s'agir de la première commande figurant dans votre `Procfile`. Le serveur nginx transmet à cette application toutes les requêtes HTTP qu'il reçoit de l'équilibrage de charge de votre environnement.

Elastic Beanstalk part du principe que toutes les entrées dans `Procfile` doivent s'exécuter en permanence et redémarre automatiquement toute application définie dans le `Procfile` qui s'arrête. Pour exécuter des commandes qui s'arrêteront et ne devraient pas être redémarrées, utilisez un [Buildfile \(p. 132\)](#).

## Utilisation d'un `Procfile` sur AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Java SE Elastic Beanstalk utilise une version de plate-forme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

### Transmission de port

Par défaut, Elastic Beanstalk configure le serveur proxy nginx pour transmettre les demandes à votre application sur le port 5000. Vous pouvez remplacer le port par défaut en définissant la [propriété d'environnement \(p. 130\)](#) `PORT` sur le port que votre application écoute.

Si vous utilisez un `Procfile` pour exécuter plusieurs applications, Elastic Beanstalk sur les versions de plateforme AMI Amazon Linux s'attend à ce que chaque application supplémentaire écoute sur un port 100 supérieur au précédent. Elastic Beanstalk définit la variable `PORT` accessible à partir de chaque application

avec le port sur lequel il s'attend à ce que l'application s'exécute. Vous pouvez accéder à cette variable dans votre code d'application en appelant `System.getenv("PORT")`.

Dans l'exemple `Procfile` précédent, l'application web écoute sur le port 5000, cache écoute sur le port 5100 et `web_foo` écoute sur le port 5200. `web` configure son port d'écoute en lisant la variable `PORT` et ajoute 100 à ce nombre pour déterminer sur quel port `cache` écoute afin de pouvoir lui envoyer des demandes.

## Configuration du proxy inverse

Elastic Beanstalk utilise `nginx` comme proxy inverse pour mapper votre application à votre équilibrEUR de charge Elastic Load Balancing sur le port 80. Elastic Beanstalk fournit une configuration `nginx` par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Par défaut, Elastic Beanstalk configure le serveur proxy `nginx` pour transmettre les demandes à votre application sur le port 5000. Vous pouvez remplacer le port par défaut en définissant la [propriété d'environnement](#) (p. 130) `PORT` sur le port que votre application écoute.

### Note

Le port que votre application écoute n'affecte pas le port que le serveur `nginx` écoute pour recevoir des demandes de l'équilibrEUR de charge.

Toutes les plateformes Amazon Linux 2 prennent en charge une configuration de proxy uniforme. Pour obtenir des détails sur la configuration du serveur proxy sur les nouvelles versions de plateforme Amazon Corretto qui exécutent Amazon Linux 2, développez la section Configuration du proxy inverse dans [the section called "Extension des plateformes Linux"](#) (p. 34).

## Configuration du proxy sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Java SE Elastic Beanstalk utilise une version de plate-forme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires dans cette section.

### Extension et remplacement de la configuration du proxy par défaut

Pour étendre la configuration `nginx` par défaut d'Elastic Beanstalk, ajoutez les fichiers de configuration `.conf` dans un dossier nommé `.ebextensions/nginx/conf.d/` dans le bundle de fichiers source de votre application. La configuration `nginx` Elastic Beanstalk inclut automatiquement les fichiers `.conf` dans ce dossier.

```
~/workspace/my-app/
|-- .ebextensions
|   '-- nginx
|       '-- conf.d
|           '-- myconf.conf
`-- web.jar
```

Pour remplacer complètement la configuration `nginx` Elastic Beanstalk par défaut, incluez une configuration dans votre bundle de fichiers source su `.ebextensions/nginx/nginx.conf`:

```
~/workspace/my-app/
|-- .ebextensions
|   '-- nginx
|       '-- nginx.conf
`-- web.jar
```

Si vous substituez une configuration `nginx` Elastic Beanstalk, ajoutez la ligne suivante à votre `nginx.conf` pour extraire les configurations Elastic Beanstalk pour [Surveillance et création de rapports d'intégrité améliorée](#) (p. 837), les mappages d'application automatiques et les fichiers statiques.

```
include conf.d/elasticbeanstalk/*.conf;
```

L'exemple de configuration suivant provenant de l'[exemple d'application Scorekeep](#) remplace la configuration Elastic Beanstalk par défaut pour servir une application web statique depuis le sous-répertoire public de /var/app/current, où la plateforme Java SE copie le code source de l'application. L'emplacement /api achemine le trafic vers des routes sous /api/ jusqu'à l'application Spring à l'écoute sur le port 5000. Le reste du trafic est servi par l'application web au chemin racine.

#### Example .ebextensions/nginx/nginx.conf

```
user                      nginx;
error_log                /var/log/nginx/error.log warn;
pid                      /var/run/nginx.pid;
worker_processes          auto;
worker_rlimit_nofile      33282;

events {
    worker_connections 1024;
}

http {
    include       /etc/nginx/mime.types;
    default_type  application/octet-stream;

    log_format  main  '$remote_addr - $remote_user [$time_local] "$request" '
                      '$status $body_bytes_sent "'.$http_referer" '
                      '"$http_user_agent" "$http_x_forwarded_for"';

    include       conf.d/*.conf;

    map $http_upgrade $connection_upgrade {
        default      "upgrade";
    }

    server {
        listen       80 default_server;
        root /var/app/current/public;

        location / {

        }

        location /api {
            proxy_pass          http://127.0.0.1:5000;
            proxy_http_version  1.1;

            proxy_set_header    Connection      $connection_upgrade;
            proxy_set_header    Upgrade        $http_upgrade;
            proxy_set_header    Host          $host;
            proxy_set_header    X-Real-IP     $remote_addr;
            proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
        }

        access_log    /var/log/nginx/access.log main;

        client_header_timeout 60;
        client_body_timeout   60;
        keepalive_timeout     60;
        gzip                 off;
        gzip_comp_level      4;

        # Include the Elastic Beanstalk generated locations
        include conf.d/elasticbeanstalk/01_static.conf;
        include conf.d/elasticbeanstalk/healthd.conf;
    }
}
```

```
}
```

## Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Java

Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données que votre application recueille et modifie. La base de données peut être associée à votre environnement et gérée par Elastic Beanstalk, ou créée et gérée en externe.

Si vous utilisez Amazon RDS pour la première fois, ajoutez une instance de base de données à un environnement de test à l'aide de la console Elastic Beanstalk et assurez-vous que votre application peut s'y connecter.

Pour ajouter une instance DB à votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Choisissez Apply.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).

Nom de la propriété	Description	Valeur de la propriété
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour de plus amples informations sur la configuration d'une instance de base de données interne, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#). Pour de plus amples informations sur la configuration d'une base de données externe à utiliser avec Elastic Beanstalk, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#).

Pour vous connecter à la base de données, ajoutez le fichier JAR du pilote approprié à votre application, chargez la classe de pilote dans votre code et créez un objet de connexion avec les propriétés d'environnement fournies par Elastic Beanstalk.

#### Sections

- [Téléchargement du pilote JDBC \(p. 137\)](#)
- [Connexion à une base de données \(plateformes Java SE\) \(p. 138\)](#)
- [Connexion à une base de données \(plateformes Tomcat\) \(p. 138\)](#)
- [Résolution des problèmes de connexion à une base de données \(p. 140\)](#)

## Téléchargement du pilote JDBC

Vous avez besoin du fichier JAR du pilote JDBC pour le moteur de base de données que vous choisissez. Enregistrez le fichier JAR dans votre code source et incluez-le dans votre chemin de classe lorsque vous compilez la classe qui crée les connexions à la base de données.

Vous trouverez le pilote le plus récent pour votre moteur de base de données dans les emplacements suivants :

- MySQL : [MySQL Connector/J](#)
- Oracle SE-1 : [pilote Oracle JDBC](#)
- Postgres : [pilote PostgreSQL JDBC](#)
- SQL Server : [pilote Microsoft JDBC](#)

Pour utiliser le pilote JDBC, appelez `Class.forName()` afin de le charger avant de créer la connexion avec `DriverManager.getConnection()` dans votre code.

JDBC utilise une chaîne de connexion au format suivant :

```
jdbc:driver://hostname:port/dbName?user=userName&password=password
```

Vous pouvez récupérer le nom d'hôte, le port, le nom de la base de données, le nom d'utilisateur et le mot de passe dans les variables d'environnement qu'Elastic Beanstalk fournit à votre application. Le nom du pilote est spécifique à votre type de base de données et à la version de votre pilote. Voici des exemples de noms de pilote :

- `mysql` pour MySQL
- `postgresql` pour PostgreSQL
- `oracle:thin` pour Oracle Thin
- `oracle:oci` pour Oracle OCI
- `oracle:oci8` pour Oracle OCI 8

- oracle:kprb pour Oracle KPRB
- sqlserver pour SQL Server

## Connexion à une base de données (plateformes Java SE)

Dans un environnement Java SE, utilisez `System.getenv()` pour lire les variables de connexion à partir de l'environnement. L'exemple de code suivant montre une classe qui crée une connexion à une base de données PostgreSQL.

```
private static Connection getRemoteConnection() {  
    if (System.getenv("RDS_HOSTNAME") != null) {  
        try {  
            Class.forName("org.postgresql.Driver");  
            String dbName = System.getenv("RDS_DB_NAME");  
            String userName = System.getenv("RDS_USERNAME");  
            String password = System.getenv("RDS_PASSWORD");  
            String hostname = System.getenv("RDS_HOSTNAME");  
            String port = System.getenv("RDS_PORT");  
            String jdbcUrl = "jdbc:postgresql://" + hostname + ":" + port + "/" + dbName + "?  
user=" + userName + "&password=" + password;  
            logger.trace("Getting remote connection with connection string from environment  
variables.");  
            Connection con = DriverManager.getConnection(jdbcUrl);  
            logger.info("Remote connection successful.");  
            return con;  
        }  
        catch (ClassNotFoundException e) { logger.warn(e.toString());}  
        catch (SQLException e) { logger.warn(e.toString());}  
    }  
    return null;  
}
```

## Connexion à une base de données (plateformes Tomcat)

Dans un environnement Tomcat, les propriétés de l'environnement sont fournies sous la forme de propriétés système auxquelles vous pouvez accéder via `System.getProperty()`.

L'exemple de code suivant montre une classe qui crée une connexion à une base de données PostgreSQL.

```
private static Connection getRemoteConnection() {  
    if (System.getProperty("RDS_HOSTNAME") != null) {  
        try {  
            Class.forName("org.postgresql.Driver");  
            String dbName = System.getProperty("RDS_DB_NAME");  
            String userName = System.getProperty("RDS_USERNAME");  
            String password = System.getProperty("RDS_PASSWORD");  
            String hostname = System.getProperty("RDS_HOSTNAME");  
            String port = System.getProperty("RDS_PORT");  
            String jdbcUrl = "jdbc:postgresql://" + hostname + ":" + port + "/" + dbName + "?  
user=" + userName + "&password=" + password;  
            logger.trace("Getting remote connection with connection string from environment  
variables.");  
            Connection con = DriverManager.getConnection(jdbcUrl);  
            logger.info("Remote connection successful.");  
            return con;  
        }  
        catch (ClassNotFoundException e) { logger.warn(e.toString());}  
        catch (SQLException e) { logger.warn(e.toString());}  
    }  
    return null;  
}
```

```
}
```

Si vous rencontrez des difficultés pour obtenir une connexion ou exécuter des instructions SQL, essayez d'insérer le code suivant dans un fichier JSP. Ce code se connecte à une instance de base de données, crée une table et écrit dedans.

```
<%@ page import="java.sql.*" %>
<%
// Read RDS connection information from the environment
String dbName = System.getProperty("RDS_DB_NAME");
String userName = System.getProperty("RDS_USERNAME");
String password = System.getProperty("RDS_PASSWORD");
String hostname = System.getProperty("RDS_HOSTNAME");
String port = System.getProperty("RDS_PORT");
String jdbcUrl = "jdbc:mysql://" + hostname + ":" +
port + "/" + dbName + "?user=" + userName + "&password=" + password;

// Load the JDBC driver
try {
    System.out.println("Loading driver...");
    Class.forName("com.mysql.jdbc.Driver");
    System.out.println("Driver loaded!");
} catch (ClassNotFoundException e) {
    throw new RuntimeException("Cannot find the driver in the classpath!", e);
}

Connection conn = null;
Statement setupStatement = null;
Statement readStatement = null;
ResultSet resultSet = null;
String results = "";
int numresults = 0;
String statement = null;

try {
    // Create connection to RDS DB instance
    conn = DriverManager.getConnection(jdbcUrl);

    // Create a table and write two rows
    setupStatement = conn.createStatement();
    String createTable = "CREATE TABLE Beanstalk (Resource char(50));";
    String insertRow1 = "INSERT INTO Beanstalk (Resource) VALUES ('EC2 Instance');";
    String insertRow2 = "INSERT INTO Beanstalk (Resource) VALUES ('RDS Instance');";

    setupStatement.addBatch(createTable);
    setupStatement.addBatch(insertRow1);
    setupStatement.addBatch(insertRow2);
    setupStatement.executeBatch();
    setupStatement.close();

} catch (SQLException ex) {
    // Handle any errors
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
} finally {
    System.out.println("Closing the connection.");
    if (conn != null) try { conn.close(); } catch (SQLException ignore) {}
}

try {
    conn = DriverManager.getConnection(jdbcUrl);

    readStatement = conn.createStatement();
```

```
resultSet = readStatement.executeQuery("SELECT Resource FROM Beanstalk;");

resultSet.first();
results = resultSet.getString("Resource");
resultSet.next();
results += ", " + resultSet.getString("Resource");

resultSet.close();
readStatement.close();
conn.close();

} catch (SQLException ex) {
    // Handle any errors
    System.out.println("SQLException: " + ex.getMessage());
    System.out.println("SQLState: " + ex.getSQLState());
    System.out.println("VendorError: " + ex.getErrorCode());
} finally {
    System.out.println("Closing the connection.");
    if (conn != null) try { conn.close(); } catch (SQLException ignore) {}
}
%>
```

Pour afficher les résultats, insérez le code suivant dans le corps de la partie HTML du fichier JSP.

```
<p>Established connection to RDS. Read first two rows: <%= results %></p>
```

## Résolution des problèmes de connexion à une base de données

Si vous rencontrez des problèmes pour vous connecter à une base de données à partir de votre application, consultez la base de données et le journal du conteneur web.

### Examen des journaux

Vous pouvez afficher tous les journaux de votre environnement Elastic Beanstalk depuis Eclipse. Si la vue « AWS Explorer » (Explorateur AWS) n'est pas ouverte, sélectionnez la flèche à côté de l'icône AWS orange dans la barre d'outils, puis choisissez Show AWS Explorer View (Afficher la vue Explorateur AWS). Développez AWS Elastic Beanstalk et le nom de votre environnement, puis ouvrez le menu contextuel (clic droit) pour le serveur. Choisissez Open in WTP Server Editor (Ouvrir dans WTP Server Editor).

Choisissez l'onglet Log (Journal) de la vue Server (Serveur) afin de visualiser les journaux cumulés provenant de votre environnement. Pour ouvrir les journaux les plus récents, cliquez sur le bouton Refresh (Actualiser) dans l'angle supérieur droit de la page.

Faites défiler la page afin de trouver les journaux Tomcat dans /var/log/tomcat7/catalina.out. Si vous avez chargé plusieurs fois la page web issue de notre exemple précédent, les données ci-dessous devraient s'afficher :

```
-----
/var/log/tomcat7/catalina.out
-----
INFO: Server startup in 9285 ms
Loading driver...
Driver loaded!
SQLException: Table 'Beanstalk' already exists
SQLState: 42S01
VendorError: 1050
Closing the connection.
Closing the connection.
```

Toutes les informations envoyées par l'application à la sortie standard s'affichent dans le journal du conteneur web. Dans l'exemple précédent, l'application tente de créer la table à chaque chargement de

la page. Cela se traduit par l'interception d'une exception SQL pour chaque chargement de page après le premier.

L'exemple qui précède est acceptable. Toutefois, dans des applications réelles, vous devez conserver vos définitions de base de données dans les objets de schéma, effectuer les transactions à partir des classes de modèle et coordonner les demandes avec des servlets contrôleur.

## Connexion à une instance DB RDS

Vous pouvez vous connecter directement à l'instance de base de données RDS dans votre environnement Elastic Beanstalk à l'aide de l'application cliente MySQL.

Commencez par ouvrir le groupe de sécurité dans votre instance DB RDS afin d'autoriser le trafic provenant de votre ordinateur.

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. À côté de l'option Endpoint (Point de terminaison), choisissez le lien de la console Amazon RDS.
6. Sur la page des détails de l'instance RDS Dashboard (Tableau de bord RDS), sous Security and Network (Sécurité et réseau), sélectionnez le groupe de sécurité commençant par rds- en regard de Security Groups (Groupes de sécurité).

### Note

La base de données peut comporter plusieurs entrées associées au libellé Security Groups (Groupes de sécurité). Utilisez le premier, qui commence par awseb, uniquement si vous avez un compte plus ancien qui n'a pas d'[Amazon Virtual Private Cloud](#) (Amazon VPC) par défaut.

7. Dans Security group details (Détails du groupe de sécurité), cliquez sur l'onglet Inbound (Entrant), puis choisissez Edit (Modifier).
8. Ajoutez une règle pour MySQL (port 3306) qui autorise le trafic provenant de votre adresse IP, spécifiée au format CIDR.
9. Choisissez Enregistrer. Les modifications prennent effet immédiatement.

Revenez sur la page des détails de configuration Elastic Beanstalk pour votre environnement et notez le point de terminaison. Vous devrez utiliser le nom de domaine pour vous connecter à l'instance DB RDS.

Installez le client MySQL et lancez une connexion à la base de données sur le port 3306. Sous Windows, installez MySQL Workbench à partir de la page d'accueil MySQL et suivez les instructions.

Sous Linux, installez le client MySQL en utilisant le gestionnaire de package correspondant à votre distribution. L'exemple suivant fonctionne sur Ubuntu et d'autres dérivés Debian.

```
// Install MySQL client
$ sudo apt-get install mysql-client-5.5
...
// Connect to database
```

```
$ mysql -h aas839jo2vwhwb.cnubrrfwfka8.us-west-2.rds.amazonaws.com -u username -ppassword
      ebdb
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 117
Server version: 5.5.40-log Source distribution
...
...
```

Une fois connecté, vous pouvez exécuter les commandes SQL pour afficher l'état de la base de données, savoir si vos tables et vos lignes ont été créées, et obtenir d'autres informations.

```
mysql> SELECT Resource from Beanstalk;
+-----+
| Resource      |
+-----+
| EC2 Instance  |
| RDS Instance  |
+-----+
2 rows in set (0.01 sec)
```

## Utilisation d'AWS Toolkit for Eclipse

AWS Toolkit for Eclipse intègre les fonctionnalités de gestion d'AWS Elastic Beanstalk dans votre environnement de développement Tomcat afin de faciliter la création de l'environnement, la configuration et le déploiement du code. Cette boîte à outils inclut la prise en charge de plusieurs comptes AWS, la gestion des environnements existants et la connexion directe aux instances dans votre environnement pour les opérations de dépannage.

### Note

AWS Toolkit for Eclipse ne prend en charge que les projets qui utilisent la plateforme Java avec Tomcat, et non la plateforme Java SE.

Pour de plus amples informations sur les conditions préalables et l'installation d'AWS Toolkit for Eclipse, accédez à <https://aws.amazon.com/eclipse>. Vous pouvez également visionner notre vidéo sur [l'utilisation d'AWS Elastic Beanstalk avec AWS Toolkit for Eclipse](#). Cette rubrique fournit également des informations utiles sur les outils, des didacticiels et des ressources supplémentaires pour les développeurs Java.

## Importation d'environnements existants dans Eclipse

Vous pouvez importer dans Eclipse des environnements existants que vous avez créés dans la console de gestion AWS.

Pour importer des environnements existants, développez le nœud AWS Elastic Beanstalk et double-cliquez sur un environnement dans AWS Explorer (Explorateur AWS), dans Eclipse. Vous pouvez désormais déployer vos applications Elastic Beanstalk dans cet environnement.

## Gestion des environnements d'application Elastic Beanstalk

### Rubriques

- [Modification des paramètres de configuration de l'environnement \(p. 143\)](#)
- [Changement de type d'environnement \(p. 144\)](#)
- [Configuration des instances de serveur EC2 à l'aide de AWS Toolkit for Eclipse \(p. 144\)](#)
- [Configuration d'Elastic Load Balancing via AWS Toolkit for Eclipse \(p. 146\)](#)

- Configuration d'Auto Scaling via AWS Toolkit for Eclipse (p. 150)
- Configuration des notifications via AWS Toolkit for Eclipse (p. 152)
- Configuration des conteneurs Java via AWS Toolkit for Eclipse (p. 152)
- Configuration des propriétés du système avec AWS Toolkit for Eclipse (p. 154)

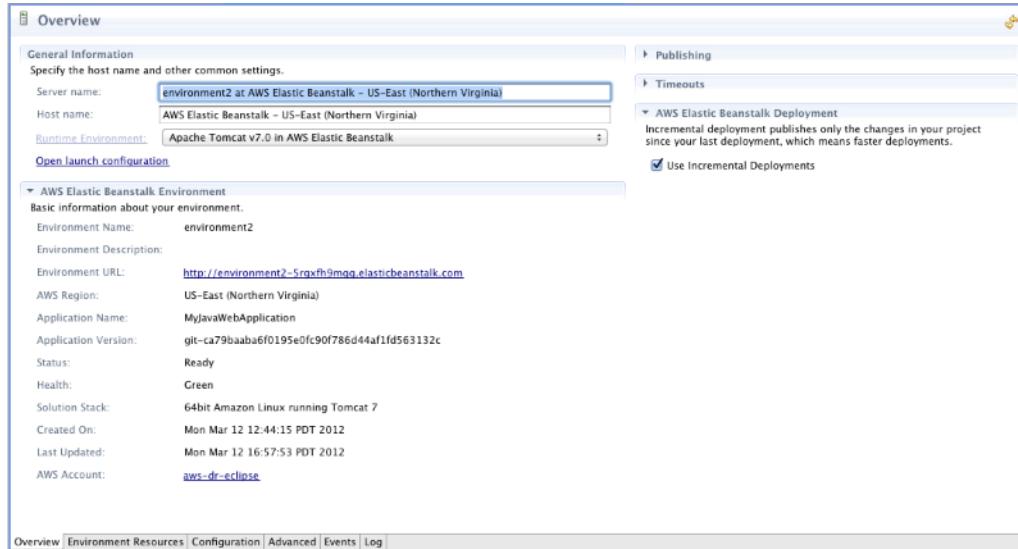
Avec AWS Toolkit for Eclipse, vous pouvez modifier l'approvisionnement et la configuration des ressources AWS qui sont utilisées par vos environnements d'application. Pour plus d'informations sur la façon de gérer les environnements de votre application à l'aide de la console de gestion AWS, consultez [Gestion des environnements \(p. 428\)](#). Cette section décrit les paramètres de service spécifiques que vous pouvez modifier dans AWS Toolkit for Eclipse dans le cadre de la configuration de votre environnement d'application. Pour en savoir plus sur AWS Toolkit for Eclipse, consultez le [Guide de démarrage de AWS Toolkit for Eclipse](#).

## Modification des paramètres de configuration de l'environnement

Lorsque vous déployez votre application, Elastic Beanstalk configure un certain nombre de services AWS de cloud computing. Vous pouvez contrôler la façon dont ces services individuels sont configurés à l'aide d'AWS Toolkit for Eclipse.

Pour modifier les paramètres d'environnement d'une application

1. Si Eclipse n'affiche pas la vue AWS Explorer (Explorateur AWS), cliquez sur Window (Fenêtre), Show View (Afficher la vue), AWS Explorer (Explorateur AWS) dans le menu. Développez le noeud Elastic Beanstalk et le noeud de votre application.
2. Dans AWS Explorer (Explorateur AWS), double-cliquez sur votre environnement Elastic Beanstalk.
3. En bas du volet, cliquez sur l'onglet Configuration.

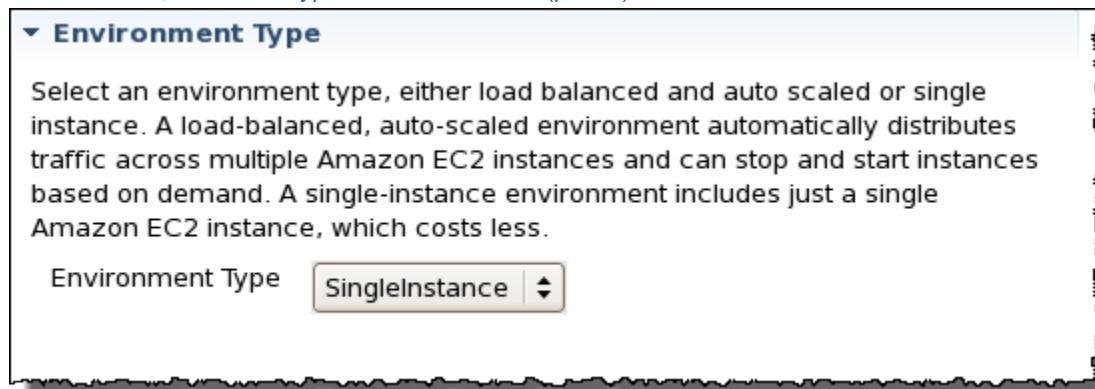


Vous pouvez à présent configurer des paramètres pour les éléments suivants :

- Instances de serveur EC2
- Equilibreur de charge
- Auto Scaling
- Notifications
- Types d'environnement
- Propriétés de l'environnement

## Changement de type d'environnement

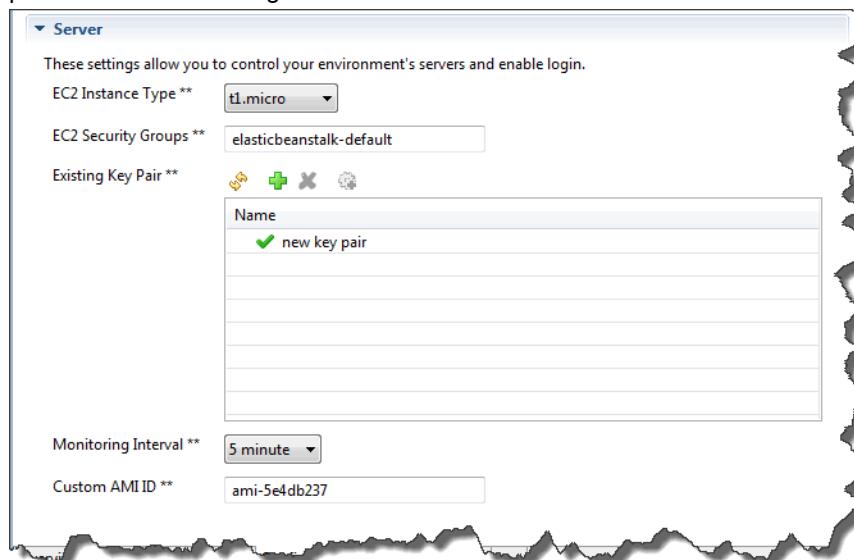
Dans AWS Toolkit for Eclipse, la section Environment Type (Type d'environnement) de l'onglet Configuration de votre environnement vous permet de sélectionner un environnement de type Load balanced, auto scaled (Équilibrage de charge, scalabilité automatique) ou Single instance (Instance unique), selon les exigences de l'application que vous déployez. Si l'application doit être mise à l'échelle, sélectionnez Load balanced, auto scaled (Équilibrage de charge, scalabilité automatique). Si l'application est simple et génère un faible trafic, sélectionnez Single instance (Instance unique). Pour plus d'informations, consultez [Types d'environnement \(p. 519\)](#).



## Configuration des instances de serveur EC2 à l'aide de AWS Toolkit for Eclipse

Amazon Elastic Compute Cloud (EC2) est un service web permettant de lancer et de gérer des instances de serveur dans les centres de données Amazon. Vous pouvez utiliser des instances de serveur Amazon EC2 à tout moment, aussi longtemps que vous le souhaitez et pour tout motif (dans le cadre d'une utilisation légale). Les instances sont disponibles dans différentes tailles et configurations. Pour de plus amples informations, veuillez consulter la [page produit Amazon EC2](#).

Sous Server (Serveur), sous l'onglet Configuration de votre environnement dans Toolkit for Eclipse, vous pouvez modifier la configuration de l'instance Amazon EC2 de l'environnement Elastic Beanstalk.



## Types d'instances Amazon EC2

Instance type (Type d'instance) affiche les types d'instance disponibles pour votre application Elastic Beanstalk. Changez le type d'instance pour sélectionner un serveur dont les caractéristiques (y compris la

taille de la mémoire et la puissance d'UC) sont les mieux adaptées à votre application. Par exemple, les applications exécutant des opérations intensives et de longue durée peuvent nécessiter plus de puissance de calcul et de mémoire.

Pour de plus amples informations sur les types d'instance Amazon EC2 disponibles pour votre application Elastic Beanstalk, veuillez consulter [Types d'instances](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

## Groupes de sécurité Amazon EC2

Vous pouvez contrôler l'accès à votre application Elastic Beanstalk par le biais d'un groupe de sécurité Amazon EC2. Un groupe de sécurité définit les règles de pare-feu de vos instances. Ces règles déterminent le trafic réseau d'entrée (c'est à dire, entrant) doit être acheminé vers votre instance. Tout autre trafic d'entrée sera ignoré. Vous pouvez modifier les règles pour un groupe à la fois. Les nouvelles règles sont appliquées automatiquement pour toutes les instances en cours d'exécution et les instances lancées par la suite.

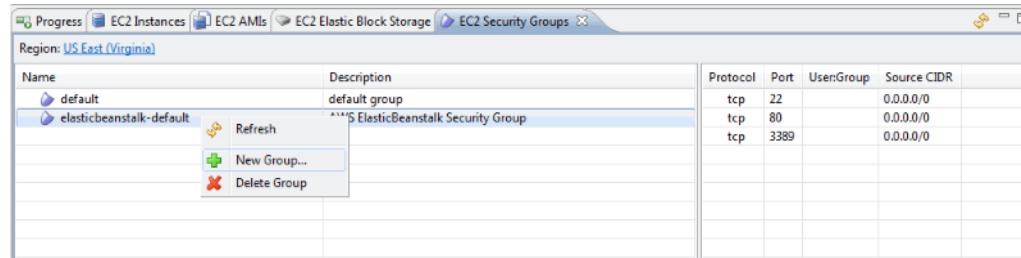
Vous pouvez configurer vos groupes de sécurité Amazon EC2 via la console de gestion AWS ou à l'aide de AWS Toolkit for Eclipse. Pour spécifier les groupes de sécurité Amazon EC2 qui contrôlent l'accès à votre application Elastic Beanstalk, saisissez les noms d'un ou de plusieurs groupes de sécurité Amazon EC2 (séparés par des virgules) dans la zone de texte EC2 Security Groups (Groupes de sécurité EC2).

### Note

Si vous exécutez votre application à l'aide d'un type de conteneur hérité, assurez-vous que le port 80 (HTTP) est accessible depuis 0.0.0.0/0 comme plage d'adresses CIDR source si vous souhaitez activer les vérifications de l'état pour votre application. Pour plus d'informations sur les vérifications de l'état, consultez [Vérifications de l'état \(p. 148\)](#). Pour vérifier si vous utilisez un type de conteneur hérité, consultez [the section called "Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?" \(p. 507\)](#)

Pour créer un groupe de sécurité à l'aide de AWS Toolkit for Eclipse

1. Dans l'AWS Toolkit for Eclipse, cliquez sur l'onglet AWS Explorer (Explorateur AWS). Développez le nœud Amazon EC2, puis double-cliquez sur Security Groups (Groupes de sécurité).
2. Cliquez avec le bouton droit sur le tableau de gauche, puis cliquez sur New Group (Nouveau groupe).



3. Dans la boîte de dialogue Security Group (Groupe de sécurité), saisissez le nom et la description du groupe de sécurité, puis cliquez sur OK.

Pour de plus amples informations sur les groupes de sécurité Amazon EC2, veuillez consulter [Utilisation des groupes de sécurité](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

## Paires de clés Amazon EC2

Vous pouvez vous connecter en toute sécurité aux instances Amazon EC2 allouées pour votre application Elastic Beanstalk avec une paire de clés Amazon EC2.

### Important

Avant de pouvoir accéder à vos instances Amazon EC2 allouées par Elastic Beanstalk, vous devez créer une paire de clés Amazon EC2 et configurer vos instances Amazon EC2 allouées

par Elastic Beanstalk pour utiliser la paire de clés Amazon EC2. Vous pouvez créer votre paire de clés avec Publish to Beanstalk Wizard (Assistant de publication sur Beanstalk) dans AWS Toolkit for Eclipse lorsque vous déployez votre application dans Elastic Beanstalk. Sinon, vous pouvez configurer vos paires de clés Amazon EC2 via la [console de gestion AWS](#). Pour obtenir des instructions sur la création d'une paire de clés pour Amazon EC2, veuillez consulter le [Guide de démarrage Amazon Elastic Compute Cloud](#).

Pour de plus amples informations sur les paires de clés Amazon EC2, accédez à [Utilisation des informations d'identification Amazon EC2](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud. Pour de plus amples informations sur la connexion à des instances Amazon EC2, accédez à [Connexion aux instances](#) et à [Connexion à votre instance Linux/UNIX à partir de Windows à l'aide de PuTTY](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

### Métriques CloudWatch

Par défaut, seules les métriques de base d'Amazon Cloudwatch sont activées. Elles renvoient des données toutes les cinq minutes. Vous pouvez activer des métriques CloudWatch plus détaillées en une minute en sélectionnant 1 minute pour la Monitoring Interval (Intervalle de surveillance) dans la section Server (Serveur) de l'onglet Configuration correspondant à votre environnement dans AWS Toolkit for Eclipse.

#### Note

Des frais de service Amazon CloudWatch peuvent s'appliquer aux métriques d'intervalle d'une minute. Pour de plus amples informations, veuillez consulter [Amazon CloudWatch](#).

### ID d'AMI personnalisé

Vous pouvez remplacer l'AMI par défaut utilisée pour vos instances Amazon EC2 par votre propre AMI personnalisée en saisissant l'identifiant de cette dernière dans la zone Custom AMI ID (ID d'AMI personnalisée) de la section Server (Serveur) de l'onglet Configuration correspondant à votre environnement dans AWS Toolkit for Eclipse.

#### Important

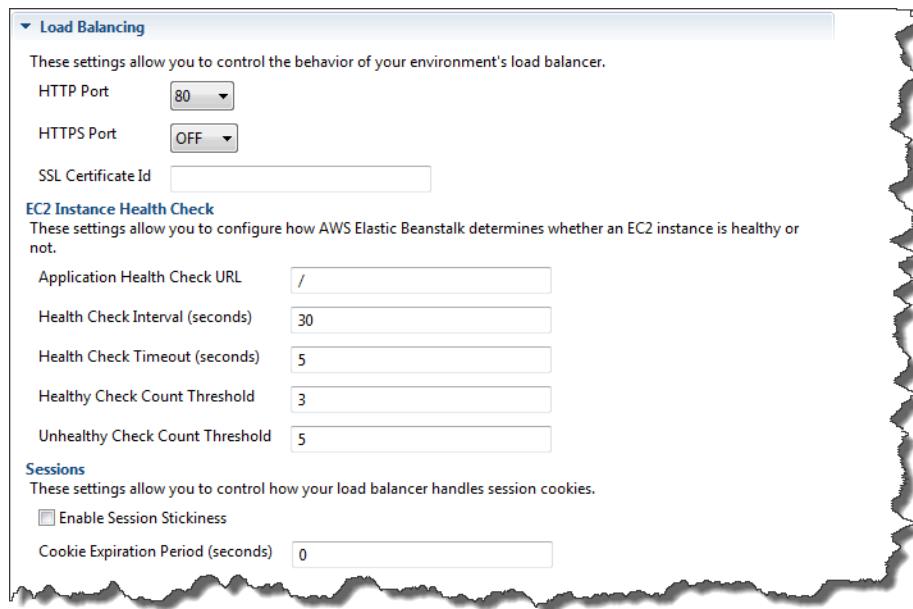
L'utilisation de votre propre image AMI est une tâche avancée qui doit être effectuée avec soin. Si vous avez besoin d'une AMI personnalisée, nous vous recommandons de démarrer par l'AMI Elastic Beanstalk par défaut, puis de la modifier. Pour être considérées saines par Elastic Beanstalk, les instances Amazon EC2 doivent respecter un ensemble de conditions, y compris disposer d'un gestionnaire hôte en cours d'exécution. Si ces conditions ne sont pas satisfaites, il se peut que votre environnement ne fonctionne pas correctement.

## Configuration d'Elastic Load Balancing via AWS Toolkit for Eclipse

Elastic Load Balancing est un service d'Amazon Web Services qui améliore la disponibilité et l'évolutivité de votre application. Avec Elastic Load Balancing, vous pouvez répartir les charges applicatives entre deux instances Amazon EC2 ou plus. Elastic Load Balancing améliore la disponibilité par la redondance et prend en charge l'augmentation du trafic pour votre application.

Elastic Load Balancing répartit et équilibre automatiquement le trafic applicatif entrant sur toutes les instances de serveur EC2 que vous exécutez. Il vous permet également d'ajouter aisément de nouvelles instances lorsque vous avez besoin d'augmenter la capacité de votre application.

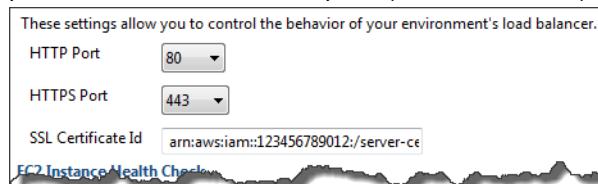
Elastic Beanstalk fournit automatiquement Elastic Load Balancing lorsque vous déployez une application. Sous Load Balancing (Équilibrage de charge), sous l'onglet Configuration de votre environnement dans Toolkit for Eclipse, vous pouvez modifier la configuration d'équilibrage de charge de l'environnement Elastic Beanstalk.



Les sections suivantes décrivent les paramètres Elastic Load Balancing que vous pouvez configurer pour votre application.

## Ports

L'équilibrEUR de charge alloué pour gérer les demandes pour votre application Elastic Beanstalk envoie des demandes aux instances Amazon EC2 qui exécutent votre application. L'équilibrEUR de charge alloué peut écouter les demandes sur les ports HTTP et HTTPS, et les acheminer vers les instances Amazon EC2 dans votre application AWS Elastic Beanstalk. Par défaut, l'équilibrEUR de charge gère les demandes sur le port HTTP. Au moins un des ports (HTTP ou HTTPS) doit être activé.



### Important

Assurez-vous que le port spécifié n'est pas verrouillé. S'il l'est, les utilisateurs ne pourront pas se connecter à votre application Elastic Beanstalk.

## Contrôle du port HTTP

Pour désactiver le port HTTP, définissez HTTP Listener Port (Port d'écoute HTTP) sur « OFF ». Pour activer le port HTTP, sélectionnez un port HTTP (par exemple, 80).

### Note

Pour accéder à votre environnement à l'aide d'un port autre que le port 80, par exemple le port 8080, vous pouvez ajouter un écouteur à l'équilibrEUR de charge existant et configurer le nouvel écouteur de sorte qu'il écoute sur ce port.

Par exemple, en utilisant la [AWS CLI for Classic load balancers](#) (CLI pour les équilibrEURS de charge Classic Load Balancer), tapez la commande suivante en remplaçant **LOAD\_BALANCER\_NAME** par le nom de votre équilibrEUR de charge pour Elastic Beanstalk.

```
aws elb create-load-balancer-listeners --load-balancer-name LOAD_BALANCER_NAME
--listeners "Protocol=HTTP, LoadBalancerPort=8080, InstanceProtocol=HTTP,
InstancePort=80"
```

Par exemple, en utilisant la [AWS CLI for Application Load Balancers](#) (CLI pour les équilibreurs de charge Application Load Balancer), tapez la commande suivante en remplaçant **LOAD\_BALANCER\_ARN** par l'ARN de votre équilibrEUR de charge pour Elastic Beanstalk.

```
aws elbv2 create-listener --load-balancer-arn LOAD_BALANCER_ARN --protocol HTTP --
port 8080
```

Si vous souhaitez que Elastic Beanstalk surveille votre environnement, ne supprimez pas l'écouteur sur le port 80.

## Contrôle du port HTTPS

Elastic Load Balancing prend en charge le protocole HTTPS/TLS pour activer le chiffrement du trafic pour les connexions client à l'équilibrEUR de charge. Les connexions entre l'équilibrEUR de charge et les instances EC2 sont effectuées à l'aide d'un texte brut. Par défaut, le port HTTPS est désactivé.

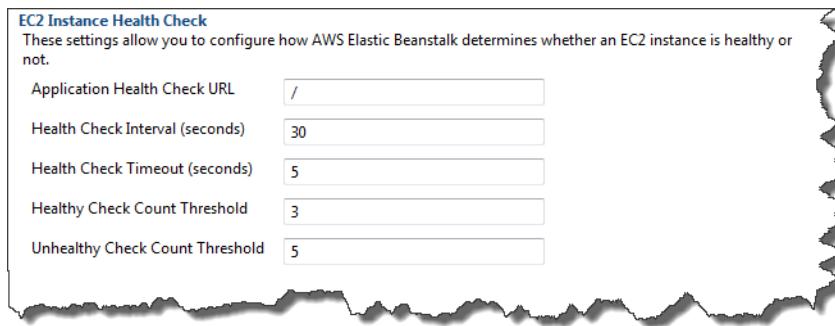
### Pour activer le port HTTPS

1. Créez un nouveau certificat à l'aide d'AWS Certificate Manager (ACM) ou téléchargez un certificat et une clé dans AWS Identity and Access Management (IAM). Pour plus d'informations sur une demande de certificat ACM, consultez [Request a Certificate](#) (Demande de certificat) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur d'AWS Certificate Manager). Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez [Importing Certificates](#) (Importation de certificats) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur d'AWS Certificate Manager). Si ACM n'est pas [disponible dans votre région AWS](#), utilisez AWS Identity and Access Management (IAM) pour télécharger un certificat tiers. Les services ACM et IAM stockeront le certificat et fourniront un Amazon Resource Name (ARN) pour le certificat SSL. Pour de plus amples informations sur la création et le chargement des certificats dans IAM, veuillez consulter [Utilisation des certificats de serveur](#) dans le Guide de l'utilisateur IAM.
2. Spécifiez le port HTTPS en sélectionnant un port dans la liste déroulante HTTPS Listener Port (Port d'écoute HTTPS).
3. Dans la zone de texte SSL Certificate ID (ID du certificat SSL), saisissez l'ARN (Amazon Resources Name) de votre certificat SSL. Par exemple, `arn:aws:iam::123456789012:server-certificate/abc/certs/build` ou `arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`. Utilisez le certificat SSL que vous avez créé et téléchargé à l'étape 1.

Pour désactiver le port HTTPS, sélectionnez OFF (désactivé) pour HTTPS Listener Port (Port d'écoute HTTPS).

## Vérifications de l'état

Vous pouvez contrôler les paramètres de vérification de l'état via la section Vérification de l'état de l'instance EC2 du panneau Équilibrage de charge.



La liste suivante décrit les paramètres de vérification de l'état que vous pouvez définir pour votre application.

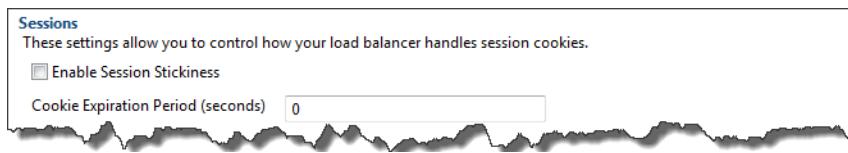
- Pour déterminer l'état de l'instance, Elastic Beanstalk recherche un code de réponse 200 sur une URL qu'il interroge. Par défaut, Elastic Beanstalk recherche TCP:80 pour les conteneurs non hérités et HTTP:80 pour les conteneurs hérités. Vous pouvez remplacer l'URL par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/index.jsp`) en entrant celle-ci dans le champ URL de vérification de l'état de l'application. Si vous remplacez l'URL par défaut, Elastic Beanstalk utilise HTTP pour interroger la ressource. Pour vérifier si vous utilisez un type de conteneur hérité, consultez [the section called "Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?" \(p. 507\)](#)
- Pour Health Check Interval (Intervalle de vérification de l'état), saisissez le nombre de secondes entre les vérifications de l'état des instances Amazon EC2.
- Pour Délai de vérification de l'état, spécifiez le laps de temps, en secondes, durant lequel Elastic Load Balancing doit attendre une réponse avant de considérer qu'une instance ne répond pas.
- Dans les champs Seuil du nombre de vérifications de l'état saines et Seuil du nombre de vérifications de l'état non saines, indiquez le nombre d'analyses d'URL réussies et non réussies consécutives avant qu'Elastic Load Balancing ne modifie l'état de l'instance. Par exemple, si vous définissez à 5 le Seuil du nombre de vérifications de l'état non saines, l'URL doit afficher un message d'erreur ou une expiration du délai cinq fois de suite avant qu'Elastic Load Balancing ne considère que la vérification de l'état est un échec.

## Sessions

Par défaut, un équilibrEUR de charge achemine chaque demande de façon indépendante à l'instance de serveur ayant la plus petite charge. Par comparaison, une session permanente lie la session d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la session soient envoyées à la même instance de serveur.

Elastic Beanstalk utilise des cookies HTTP générés par l'équilibrEUR de charge lorsque des sessions permanentes sont activées pour une application. L'équilibrEUR de charge utilise un cookie spécial généré par l'équilibrEUR de charge pour suivre l'instance d'application pour chaque demande. Lorsque l'équilibrEUR de charge reçoit une demande, il vérifie d'abord si ce cookie est présent dans la demande. Si tel est le cas, la demande est envoyée à l'instance d'application spécifiée dans le cookie. S'il ne trouve aucun cookie, l'équilibrEUR de charge choisit une instance d'application à partir de l'algorithme d'équilibrage de charge existant. Un cookie est inséré dans la réponse pour lier les demandes suivantes provenant du même utilisateur à cette instance d'application. La configuration de la stratégie définit l'expiration d'un cookie, ce qui établit la durée de validité de chaque cookie.

Sous ÉquilibrEUR de charge, dans la section Sessions, indiquez si l'équilibrEUR de charge pour votre application autorise la permanence des sessions, ainsi que la durée pour chaque cookie.



Pour de plus amples informations sur Elastic Load Balancing, veuillez consulter le [Manuel du développeur Elastic Load Balancing](#).

## Configuration d'Auto Scaling via AWS Toolkit for Eclipse

Amazon EC2 Auto Scaling est un service web Amazon conçu pour lancer ou résilier automatiquement les instances Amazon EC2 en fonction de déclencheurs définis par l'utilisateur. Les utilisateurs peuvent configurer des groupes Auto Scaling et y associer des déclencheurs afin de mettre à l'échelle automatiquement les ressources de calcul selon des métriques comme l'utilisation de la bande passante ou l'utilisation de l'UC. Amazon EC2 Auto Scaling fonctionne avec Amazon CloudWatch afin de récupérer des métriques pour les instances de serveur exécutant votre application.

Amazon EC2 Auto Scaling vous permet de récupérer un groupe d'instances Amazon EC2 et de définir différents paramètres pour que ce groupe augmente ou diminue automatiquement en nombre. Amazon EC2 Auto Scaling peut ajouter ou supprimer des instances Amazon EC2 de ce groupe pour vous aider à gérer facilement l'évolution du trafic vers votre application.

De plus, Amazon EC2 Auto Scaling surveille l'état de chaque instance Amazon EC2 qu'il lance. Si une instance est résiliée de façon inattendue, Amazon EC2 Auto Scaling détecte cette résiliation et lance une instance de remplacement. Cette fonctionnalité vous permet de maintenir automatiquement un nombre fixe et souhaité d'instances Amazon EC2.

Elastic Beanstalk met en service Amazon EC2 Auto Scaling pour votre application. Sous Auto Scaling, sous l'onglet Configuration de votre environnement dans Toolkit for Eclipse, vous pouvez modifier la configuration Auto Scaling de l'environnement Elastic Beanstalk.

Auto-scaling automatically launches or terminates EC2 instances based on defined metrics and thresholds called triggers. Auto-scaling will also launch a new EC2 instance in the event of a failure. These settings allow you to control auto-scaling behavior.

Scaling Trigger

Trigger Measurement	NetworkOut
Trigger Statistic	Average
Unit of Measurement	Bytes
Measurement Period (seconds)	5
Breach Duration (seconds)	5
Upper Threshold	6000000
Scale-up Increment	1
Lower Threshold	2000000
Scale-down Increment	-1

Les sections suivantes expliquent comment configurer les paramètres Auto Scaling pour votre application.

## Configuration de lancement

Vous pouvez modifier la configuration de lancement pour contrôler la façon dont votre application Elastic Beanstalk alloue des ressources Amazon EC2 Auto Scaling.

Utilisez les zones Minimum Instance Count (Nombre minimum d'instances) et Maximum Instance Count (Nombre maximum d'instances) pour spécifier les tailles maximale et minimale du groupe Auto Scaling utilisé par votre application Elastic Beanstalk.

The screenshot shows a configuration interface for an Auto Scaling group. It includes fields for 'Minimum Instance Count' (set to 1), 'Maximum Instance Count' (set to 4), 'Availability Zones' (set to 'Any 1'), and 'Scaling Cooldown Time (seconds)' (set to 360). The background of the interface has a decorative, wavy pattern.

Minimum Instance Count	1
Maximum Instance Count	4
Availability Zones	Any 1
Scaling Cooldown Time (seconds)	360

### Note

Pour maintenir un nombre fixe d'instances Amazon EC2, indiquez la même valeur dans les zones de texte Minimum Instance Count (Nombre minimum d'instances) et Maximum Instance Count (Nombre maximum d'instances)

Pour Availability Zones (Zones de disponibilité), spécifiez le nombre de zones de disponibilité dans lesquelles vous souhaitez inclure vos instances Amazon EC2. Il est important que vous définissiez ce nombre si vous souhaitez créer des applications tolérantes aux pannes : si une zone de disponibilité est défaillante, vos instances seront toujours exécutées dans vos autres zones de disponibilité.

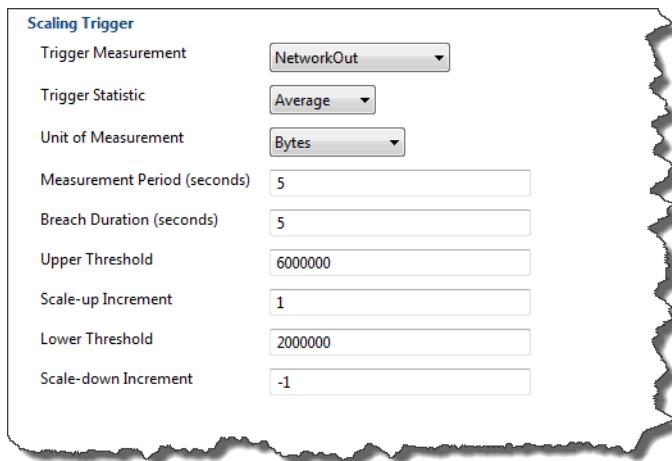
### Note

Il est actuellement impossible de spécifier la zone de disponibilité dans laquelle se situera votre instance.

## Triggers

Un déclencheur est un mécanisme Amazon EC2 Auto Scaling que vous définissez pour indiquer au système quand augmenter (monter en puissance) et diminuer (diminuer en puissance) le nombre d'instances. Vous pouvez configurer les déclencheurs pour qu'ils soient activés en fonction de n'importe quelle métrique publiée dans Amazon CloudWatch, telle que l'utilisation de l'UC, et pour déterminer si les conditions spécifiées sont réunies. Lorsque vos seuils inférieurs ou supérieurs pour la métrique ont été dépassés pendant la période spécifiée, le déclencheur lance un processus de longue durée que nous appelons activité de mise à l'échelle.

Vous pouvez définir un déclencheur de mise à l'échelle pour votre application Elastic Beanstalk avec AWS Toolkit for Eclipse.



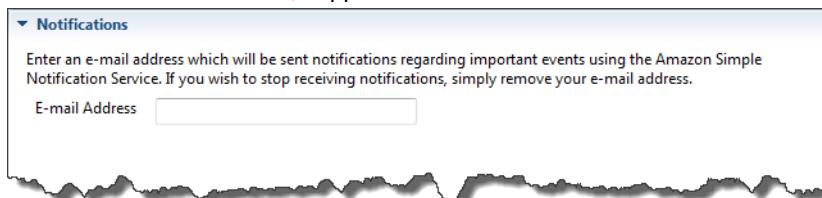
Vous pouvez configurer la liste suivante de paramètres de déclencheur dans la section Scaling Trigger (Déclencheur de dimensionnement) de l'onglet Configuration pour votre environnement, dans Toolkit for Eclipse.

- Pour Mesure du déclencheur, spécifiez la métrique relative à votre déclencheur.
- Pour Trigger Statistic (Statistique du déclencheur), spécifiez la statistique qui sera utilisée par le déclencheur : **Minimum**, **Maximum**, **Sum** ou **Average**.
- Pour Unité de mesure, spécifiez les unités de mesure du déclencheur.
- Pour Durée de mesure, spécifiez la fréquence à laquelle Amazon CloudWatch mesure les métriques pour votre déclencheur. Pour Durée de la faille, indiquez pendant combien de temps une métrique peut se situer au-delà de la limite définie (dans Seuil supérieur et Seuil inférieur) avant l'activation du déclencheur.
- Pour Scale-up Increment (Incrément d'augmentation) et Scale-down Increment (Incrément de diminution), spécifiez le nombre d'instances Amazon EC2 à ajouter ou supprimer lorsque vous effectuez une activité de mise à l'échelle.

Pour de plus amples informations sur Amazon EC2 Auto Scaling, veuillez consulter Amazon EC2 Auto Scaling dans la [documentation Amazon Elastic Compute Cloud](#).

## Configuration des notifications via AWS Toolkit for Eclipse

Elastic Beanstalk utilise Amazon Simple Notification Service (Amazon SNS) pour vous informer des événements importants qui concernent votre application. Pour activer les notifications Amazon SNS, il vous suffit de saisir votre adresse e-mail dans la zone Email Address (Adresse e-mail) sous Notifications, dans l'onglet Configuration correspondant à votre environnement dans Toolkit for Eclipse. Pour désactiver les notifications Amazon SNS, supprimez votre adresse e-mail de la zone de texte.



## Configuration des conteneurs Java via AWS Toolkit for Eclipse

Le panneau Container/JVM Options (Options de conteneur/JVM) vous permet d'ajuster le comportement de la machine virtuelle Java sur vos instances Amazon EC2, et d'activer ou de désactiver la rotation des

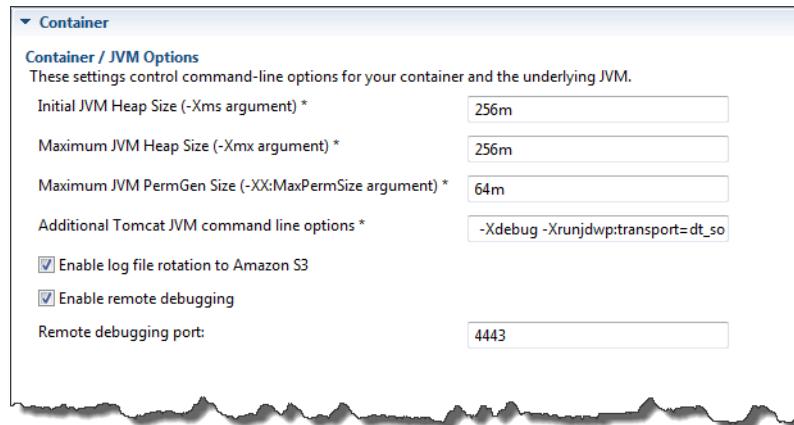
journaux Amazon S3. Vous pouvez utiliser AWS Toolkit for Eclipse pour configurer vos informations de conteneur. Pour plus d'informations sur les options disponibles pour les environnements Tomcat, consultez [the section called "Configuration de votre environnement Tomcat" \(p. 119\)](#).

#### Note

Pour modifier vos paramètres de configuration sans aucune interruption, échangez le CNAME pour vos environnements. Pour plus d'informations, consultez [Déploiements bleu/vert avec Elastic Beanstalk \(p. 486\)](#).

Pour accéder au panneau des options de conteneur/machine virtuelle Java de votre application Elastic Beanstalk

1. Si Eclipse n'affiche pas la vue AWS Explorer (Explorateur AWS), cliquez sur Window (Fenêtre), Show View (Afficher la vue), AWS Explorer (Explorateur AWS) dans le menu. Développez le nœud Elastic Beanstalk et le nœud de votre application.
2. Dans AWS Explorer (Explorateur AWS), double-cliquez sur votre environnement Elastic Beanstalk.
3. En bas du volet, cliquez sur l'onglet Configuration.
4. Sous Container (Conteneur), vous pouvez configurer les options de conteneur.



#### Débogage à distance

Pour tester votre application à distance, vous pouvez l'exécuter en mode de débogage.

Pour activer le débogage à distance

1. Sélectionnez Enable remote debugging (Activer le débogage à distance).
2. Pour Remote debugging port (Port de débogage à distance), spécifiez le numéro de port à utiliser pour le débogage à distance.

Le paramètre Additional Tomcat JVM command line options (Options de ligne de commande de la JVM Tomcat supplémentaires) est renseigné automatiquement.

Pour démarrer le débogage à distance

1. Dans le menu AWS Toolkit for Eclipse, choisissez Window (Fenêtre), Show view (Afficher la vue), Other (Autre).
2. Développez le dossier Server (Serveur), puis choisissez Servers (Serveurs). Choisissez OK.
3. Dans le volet Servers (Serveurs), cliquez avec le bouton droit sur le serveur sur lequel votre application s'exécute, puis cliquez sur Restart in Debug (Redémarrer en débogage).

## Configuration des propriétés du système avec AWS Toolkit for Eclipse

L'exemple suivant définit la propriété système `JDBC_CONNECTION_STRING` dans AWS Toolkit for Eclipse. Une fois que vous avez défini ces propriétés, elles sont disponibles dans votre application Elastic Beanstalk en tant que propriétés système appelées `JDBC_CONNECTION_STRING`.

### Note

AWS Toolkit for Eclipse ne prend pas encore en charge la modification de la configuration d'environnement, y compris des propriétés système, pour les environnements exécutés dans un VPC. Vous devez utiliser la console de gestion AWS (décrite dans la section suivante) ou [l'interface de ligne de commande \(CLI\) EB \(p. 1027\)](#), sauf si vous disposez d'un compte plus ancien utilisant EC2 Classic.

### Note

Les paramètres de configuration de l'environnement peuvent contenir tout caractère ASCII, à l'exception de l'accent grave (', ASCII 96) et ne peuvent pas dépasser 200 caractères.

Pour définir les propriétés système de votre application Elastic Beanstalk

1. Si Eclipse n'affiche pas la vue AWS Explorer (Explorateur AWS), cliquez sur Window (Fenêtre), Show View (Afficher la vue), Other (Autre). Développez AWS Toolkit, puis cliquez sur AWS Explorer.
2. Dans le volet AWS Explorer (Explorateur AWS), développez Elastic Beanstalk, développez le nœud de votre application, puis double-cliquez sur votre environnement Elastic Beanstalk.
3. En bas du volet correspondant à votre environnement, cliquez sur l'onglet Advanced (Avancé).
4. Sous `aws:elasticbeanstalk:application:environment`, cliquez sur `JDBC_CONNECTION_STRING`, puis saisissez une chaîne de connexion. Par exemple, la chaîne de connexion JDBC suivante vous connecterait à une instance de base de données MySQL sur le port 3306 de l'hôte local, avec le nom d'utilisateur `me` et le mot de passe `mypassword` :

```
jdbc:mysql://localhost:3306/mydatabase?user=me&password=mypassword
```

Cela sera accessible à votre application Elastic Beanstalk en tant que propriété système appelée `JDBC_CONNECTION_STRING`.

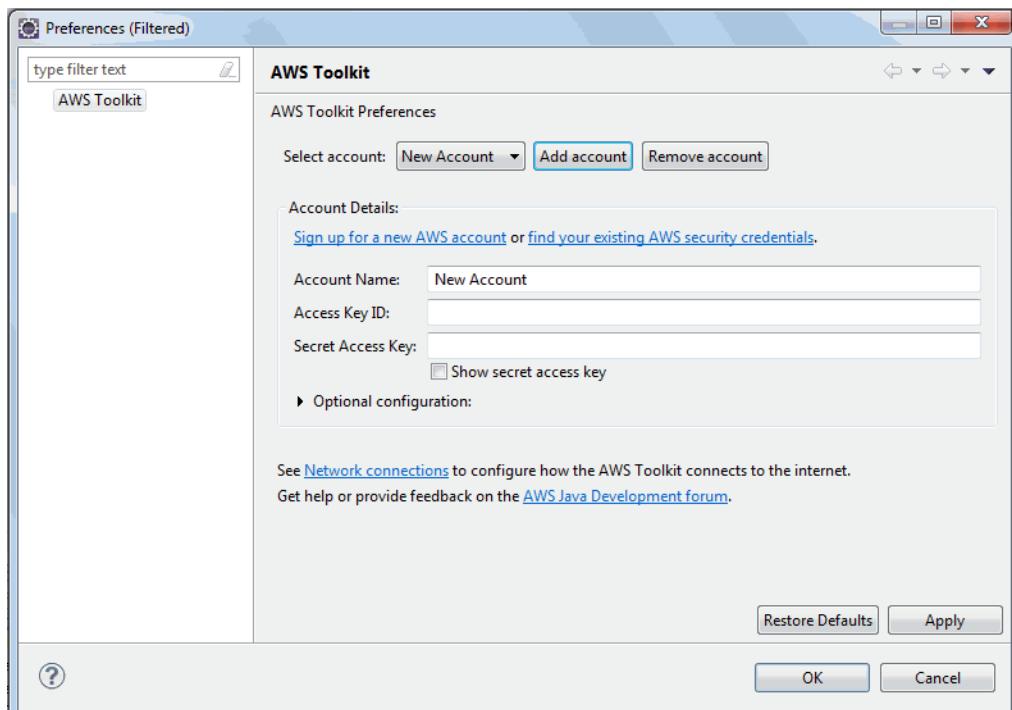
5. Appuyez sur Ctrl+C sur le clavier ou choisissez Fichier, Enregistrer pour enregistrer les modifications apportées à la configuration de l'environnement. Les changements apparaissent en une minute environ.

## Gestion de plusieurs comptes AWS

Si vous le souhaitez, vous pouvez configurer différents comptes AWS afin d'exécuter différentes tâches, telles que les tests, la mise en place et la production. AWS Toolkit for Eclipse vous permet d'ajouter, de modifier et de supprimer des comptes facilement.

Pour ajouter un compte AWS avec AWS Toolkit for Eclipse

1. Dans Eclipse, assurez-vous que la barre d'outils est visible. Dans la barre d'outils, cliquez sur la flèche à côté de l'icône AWS et sélectionnez Preferences (Préférences).
2. Cliquez sur Add account (Ajouter un compte).



3. Dans le champ Account Name (Nom du compte), saisissez le nom complet du compte.
4. Dans le champ Access Key ID (identifiant de la clé d'accès), saisissez votre ID de clé d'accès AWS.
5. Dans le champ Secret Access Key (clé d'accès secrète), saisissez votre clé d'accès secrète AWS.

Pour accéder à l'API, vous avez besoin d'un ID de clé d'accès et d'une clé d'accès secrète. Utilisez les clés d'accès utilisateur IAM au lieu des clés d'accès utilisateur racine du Compte AWS . Pour de plus amples informations sur la création de clés d'accès, veuillez consulter [Gestion des clés d'accès pour les utilisateurs IAM](#) dans le Guide de l'utilisateur IAM.

6. Cliquez sur OK.

Pour utiliser un autre compte afin de déployer une application dans Elastic Beanstalk

1. Dans la barre d'outils Eclipse, cliquez sur la flèche à côté de l'icône AWS et sélectionnez Preferences (Préférences).
2. Pour l'option Default Account (Compte par défaut), sélectionnez le compte que vous souhaitez utiliser pour déployer les applications dans Elastic Beanstalk.
3. Cliquez sur OK.
4. Dans le panneau Project Explorer (Explorateur de projet), cliquez avec le bouton droit sur l'application que vous souhaitez déployer, puis sélectionnez Amazon Web Services > Deploy to Elastic Beanstalk (Déployer dans Elastic Beanstalk).

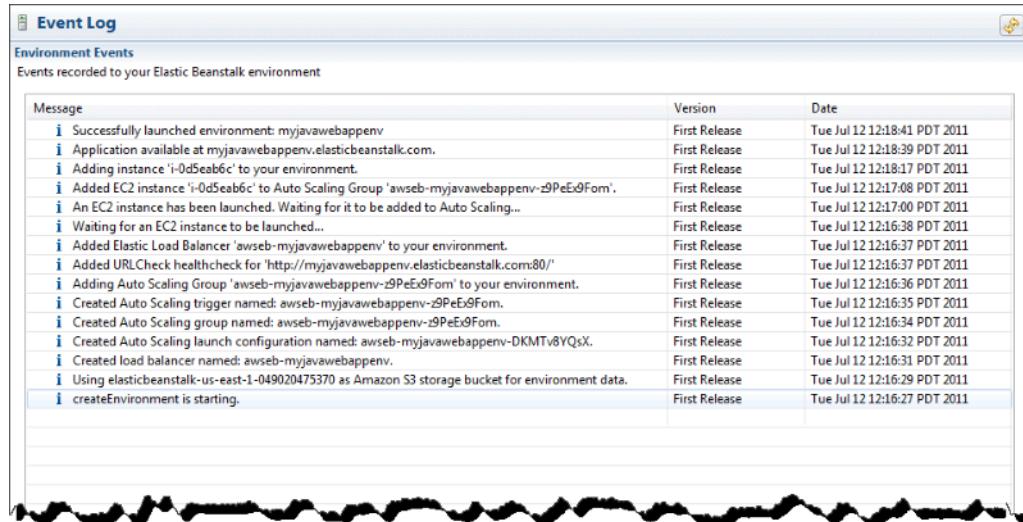
## Affichage d'événements

Vous pouvez utiliser AWS Toolkit for Eclipse pour accéder aux événements et aux notifications associés à votre application.

Pour afficher des événements d'application

1. Si Eclipse n'affiche pas la vue AWS Explorer (Explorateur AWS), cliquez sur Window (Fenêtre) > Show View (Afficher la vue) > AWS Explorer (Explorateur AWS) dans le menu. Développez le noeud Elastic Beanstalk et le noeud de votre application.
2. Dans AWS Explorer (Explorateur AWS), double-cliquez sur votre environnement Elastic Beanstalk.
3. En bas du volet, cliquez sur l'onglet Events (Événements).

La liste des événements pour tous les environnements de votre application s'affiche.



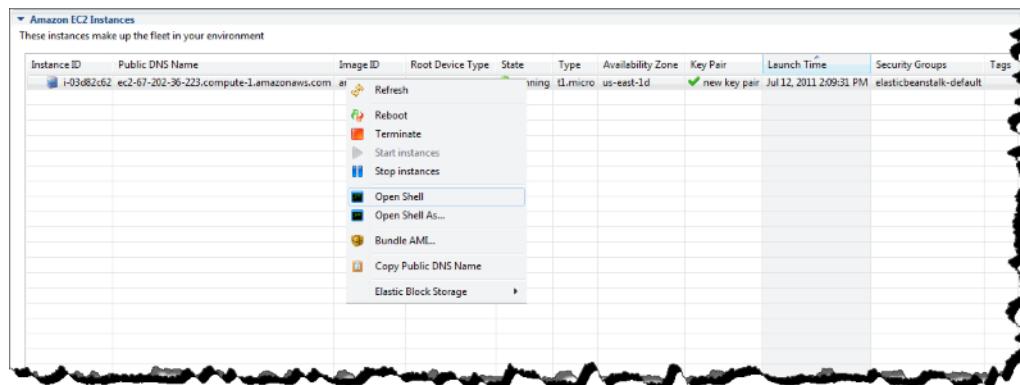
Message	Version	Date
i Successfully launched environment: myjavawebappenv	First Release	Tue Jul 12 12:18:41 PDT 2011
i Application available at <a href="http://myjavawebappenv.elasticbeanstalk.com">myjavawebappenv.elasticbeanstalk.com</a> .	First Release	Tue Jul 12 12:18:39 PDT 2011
i Adding instance 'i-0dSeab6c' to your environment.	First Release	Tue Jul 12 12:18:17 PDT 2011
i Added EC2 instance 'i-0dSeab6c' to Auto Scaling Group 'awseb-myjavawebappenv-9PeEx9Form'.	First Release	Tue Jul 12 12:17:08 PDT 2011
i An EC2 instance has been launched. Waiting for it to be added to Auto Scaling...	First Release	Tue Jul 12 12:17:00 PDT 2011
i Waiting for an EC2 instance to be launched...	First Release	Tue Jul 12 12:16:38 PDT 2011
i Added Elastic Load Balancer 'awseb-myjavawebappenv' to your environment.	First Release	Tue Jul 12 12:16:37 PDT 2011
i Added URLCheck healthcheck for ' <a href="http://myjavawebappenv.elasticbeanstalk.com:80/">http://myjavawebappenv.elasticbeanstalk.com:80/</a> '	First Release	Tue Jul 12 12:16:37 PDT 2011
i Adding Auto Scaling Group 'awseb-myjavawebappenv-9PeEx9Form' to your environment.	First Release	Tue Jul 12 12:16:36 PDT 2011
i Created Auto Scaling trigger named: awseb-myjavawebappenv-9PeEx9Form.	First Release	Tue Jul 12 12:16:35 PDT 2011
i Created Auto Scaling group named: awseb-myjavawebappenv-9PeEx9Form.	First Release	Tue Jul 12 12:16:34 PDT 2011
i Created Auto Scaling launch configuration named: awseb-myjavawebappenv-DKMTv8YQsX.	First Release	Tue Jul 12 12:16:32 PDT 2011
i Created load balancer named: awseb-myjavawebappenv.	First Release	Tue Jul 12 12:16:31 PDT 2011
i Using elasticbeanstalk-us-east-1-049020475370 as Amazon S3 storage bucket for environment data.	First Release	Tue Jul 12 12:16:29 PDT 2011
i createEnvironment is starting.	First Release	Tue Jul 12 12:16:27 PDT 2011

## Affichage de la liste des instances de serveur et connexion à ces instances

Vous pouvez afficher la liste des instances Amazon EC2 exécutant votre environnement d'applications Elastic Beanstalk via AWS Toolkit for Eclipse ou la console de gestion AWS. Vous pouvez vous connecter à ces instances via Secure Shell (SSH). Pour plus d'informations sur la façon d'obtenir la liste de vos instances de serveur et de vous y connecter via la console de gestion AWS, consultez [Affichage de la liste des instances de serveur et connexion à ces instances \(p. 881\)](#). La section suivante vous explique comment afficher vos instances de serveur et vous y connecter via AWS Toolkit for Eclipse.

Pour afficher les instances Amazon EC2 d'un environnement et s'y connecter

1. Dans AWS Toolkit for Eclipse, cliquez sur AWS Explorer (Explorateur AWS). Développez le noeud Amazon EC2 et double-cliquez sur Instances.
2. Dans la fenêtre des instances Amazon EC2, dans la colonne Instance ID (ID d'instance), cliquez avec le bouton droit sur l'option Instance ID (ID d'instance) qui correspond à l'instance Amazon EC2 exécutée dans l'équilibrage de charge de votre application. Ensuite, cliquez sur Open Shell (Ouvrir Shell).



Eclipse ouvre automatiquement le client SSH et établit la connexion à l'instance EC2.

Pour de plus amples informations sur la connexion à une instance Amazon EC2, veuillez consulter le [Manuel de mise en route Amazon Elastic Compute Cloud](#).

## Résiliation d'un environnement

Pour éviter de payer des frais pour des ressources AWS inutilisées, vous pouvez résilier un environnement en cours d'exécution via AWS Toolkit for Eclipse. Pour de plus amples informations sur l'arrêt de l'environnement, veuillez consulter [Arrêt d'un environnement Elastic Beanstalk \(p. 464\)](#).

Pour résilier un environnement

1. Dans AWS Toolkit for Eclipse, cliquez sur le volet AWS Explorer (Explorateur AWS). Développez le nœud Elastic Beanstalk.
2. Développez l'application Elastic Beanstalk et cliquez avec le bouton droit de la souris sur l'environnement Elastic Beanstalk.
3. Cliquez sur Terminate Environment (Résilier l'environnement). Il faudra quelques minutes à Elastic Beanstalk pour résilier les ressources AWS en cours d'exécution dans l'environnement.

## Resources

Il existe plusieurs endroits auxquels vous pouvez accéder pour obtenir une aide supplémentaire lors du développement de vos applications Java.

Ressource	Description
<a href="#">Forum de développement Java AWS</a>	Posez vos questions et obtenez des commentaires.
<a href="#">Centre pour développeurs Java</a>	Guichet unique pour l'exemple de code, la documentation, les outils et les ressources supplémentaires.

## Utilisation de .NET Core sous Linux

Cette section fournit des informations sur le déploiement d'applications .NET core sous Linux avec AWS Elastic Beanstalk.

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

#### Rubriques

- [Démarrez avec .NET Core sous Linux \(p. 158\)](#)
- [Configuration de votre environnement de développement .NET Core sous Linux \(p. 160\)](#)
- [Utilisation de la plateforme .NET Core sous Linux \(p. 161\)](#)
- [Tutorial : Déploiement d'une application ASP.NET Core sous Linux avec Elastic Beanstalk \(p. 165\)](#)
- [L'AWS Toolkit for Visual Studio - Utilisation de .Net Core \(p. 171\)](#)
- [Migration depuis .NET sur les plateformes Windows Server vers la plateforme .NET Core sous Linux \(p. 191\)](#)

## Démarrez avec .NET Core sous Linux

Pour commencer à utiliser des applications .NET Core sous Linux sur AWS Elastic Beanstalk, il vous suffit de charger la [solution groupée de fichiers source \(p. 415\)](#) d'une application comme première version de l'application et de la déployer dans un environnement. Lorsque vous créez un environnement, Elastic Beanstalk alloue toutes les ressources AWS nécessaires pour exécuter une application web hautement évolutive.

### Lancement d'un environnement avec un exemple d'application .NET Core sous Linux

Elastic Beanstalk fournit des exemples d'application d'une seule page pour chaque plateforme.

#### Samples

Configurations prises en charge	Type d'environnement	Solution groupée source	Description
.NET Core sous Linux	Serveur web	<a href="#">dotnet-core-linux.zip</a>	Application d'une seule page.
.NET Core sous Linux	Serveur web	<a href="#">dotnet-core-linux-multiple-apps.zip</a>	Deux applications Web qui s'exécutent sur le même serveur Web.

Téléchargez l'exemple d'application et déployez-le dans Elastic Beanstalk en procédant comme suit :

Pour lancer un environnement avec un exemple d'application (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le volet de navigation, choisissez Applications, puis le nom d'une application existante dans la liste ou [créez-en un \(p. 406\)](#).
3. Sur la page de présentation de l'application, choisissez Create a new environment (Créer un nouvel environnement).

Environment name	Health	Date created	Last modified	URL	Platform
GettingStartedApp-env	<span>OK</span>	2020-01-28 12:06:50 UTC-0800	2020-01-30 15:02:35 UTC-0800	GettingStartedApp-env.bn7dx222kw.us-east-2.elasticbeanstalk.com	Tomcat running Linux
GettingStartedApp-Windows	<span>OK</span>	2020-01-28 16:34:29 UTC-0800	2020-01-28 16:38:20 UTC-0800	GettingStartedApp-Windows.bn7dx222kw.us-east-2.elasticbeanstalk.com	IIS 10.0 Windows

4. Ensuite, pour le niveau d'environnement, choisissez l'environnement de serveur web ou le [niveau d'environnement de travail \(p. 14\)](#). Vous ne pouvez pas modifier le niveau d'un environnement après sa création.

Note

La plateforme [.NET sur Windows Server \(p. 192\)](#) ne prend pas en charge le niveau d'environnement worker.

**Select environment tier**

AWS Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web server environments run a website, web application, or web API that serves HTTP requests. Worker environments run a background processing task that listens for messages on an Amazon SQS queue. Worker applications post those messages to your application by using HTTP.

**Web server environment**  
Run a website, web application, or web API that serves HTTP requests.  
[Learn more](#)

**Worker environment**  
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule.  
[Learn more](#)

5. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

Note

Elastic Beanstalk prend en charge plusieurs [versions \(p. 31\)](#) pour la plupart des plateformes répertoriées. Par défaut, la console sélectionne la version recommandée pour la plateforme et la branche de plateforme que vous choisissez. Si votre application nécessite

une version différente, vous pouvez la sélectionner ici, ou choisir Configurer plus d'options, selon les instructions de l'étape 7. Pour plus d'informations sur les versions de plateforme prises en charge, consultez [the section called “Plateformes prises en charge” \(p. 31\)](#).

6. Pour l'option Code de l'application, choisissez Exemple d'application.
7. Pour personnaliser davantage votre environnement, choisissez Configurer plus d'options. Les options suivantes peuvent être définies uniquement lors de la création de l'environnement :
  - Nom de l'environnement
  - Nom de domaine
  - Version de plateforme
  - VPC
  - Palier

Vous pouvez modifier les paramètres suivants après la création de l'environnement, mais ils requièrent la mise en œuvre de nouvelles instances ou d'autres ressources et leur application peut prendre du temps :

- Type d'instance, volume racine, paire de clés et rôle AWS Identity and Access Management (IAM)
- Base de données interne Amazon RDS
- Equilibreur de charge

Pour de plus amples informations sur tous les paramètres disponibles, veuillez consulter [Assistant de création d'un environnement \(p. 442\)](#).

8. Choisissez Create environment.

## Étapes suivantes

Dès que vous disposez d'un environnement exécutant une application, vous pouvez à tout moment déployer une nouvelle version de l'application ou une application totalement différente. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire d'allouer ou de redémarrer les instances Amazon EC2.

Une fois que vous avez déployé un ou deux exemples d'application et que vous êtes prêt à développer et exécuter les applications .NET Core en local, veuillez consulter [Configuration de votre environnement de développement .NET Core sous Linux \(p. 160\)](#).

## Configuration de votre environnement de développement .NET Core sous Linux

Configurez un environnement de développement .NET Core pour tester votre application en local avant de la déployer dans AWS Elastic Beanstalk. Cette rubrique décrit les étapes de configuration de l'environnement de développement et des liens vers les pages d'installation pour des outils utiles.

Pour accéder aux outils et aux étapes de configuration courants qui s'appliquent à toutes les langues, veuillez consulter [Configuration de votre machine de développement pour une utilisation avec Elastic Beanstalk \(p. 1024\)](#).

### Sections

- [Installation du kit SDK .NET Core \(p. 161\)](#)
- [Installation d'un IDE \(p. 161\)](#)
- [Installation du AWS Toolkit for Visual Studio \(p. 161\)](#)

## Installation du kit SDK .NET Core

Vous pouvez utiliser le kit SDK .NET Core pour développer des applications qui s'exécutent sur Linux.

Consultez la [page de téléchargements .NET](#) pour télécharger et installer le kit SDK .NET Core.

## Installation d'un IDE

Les environnements de développement intégré (IDE) offrent un éventail de fonctions qui facilitent le développement d'applications. Si vous n'avez pas utilisé un IDE pour le développement .NET, essayez Visual Studio Community pour démarrer.

Veuillez consulter la page [Visual Studio Community](#) pour télécharger et installer Visual Studio Community.

## Installation du AWS Toolkit for Visual Studio

[AWS Toolkit for Visual Studio \(p. 224\)](#) est un plugin open source pour l'IDE Visual Studio. Il permet aux développeurs de développer, de déboguer et de déployer plus facilement des applications .NET utilisant AWS. Pour obtenir des instructions d'installation, veuillez consulter la [page d'accueil Toolkit for Visual Studio](#).

## Utilisation de la plateforme .NET Core sous Linux

La plateforme .NET Core sous Linux AWS Elastic Beanstalk est un ensemble de [versions de plateforme](#) pour les applications .NET Core qui s'exécutent sur le système d'exploitation Linux.

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter [the section called "Extension des plateformes Linux" \(p. 34\)](#). Voici quelques considérations spécifiques à la plateforme.

## Présentation de la plateforme .NET Core sous Linux

### Serveur proxy

La plateforme Elastic Beanstalk .NET Core sous Linux inclut un proxy inverse qui transmet les demandes à votre application. Par défaut, Elastic Beanstalk utilise [nginx](#) en tant que serveur proxy. Vous pouvez choisir de n'utiliser aucun serveur proxy et de configurer [Kestrel](#) en tant que serveur Web. Kestrel est inclus par défaut dans les modèles de projet ASP.NET Core.

### Structure d'application

Vous pouvez publier des applications dépendantes de l'exécution qui utilisent l'environnement d'exécution .NET Core fourni par Elastic Beanstalk. Vous pouvez également publier des applications autonomes qui incluent l'environnement d'exécution .NET Core et les dépendances de votre application dans le bundle de fichiers source. Pour en savoir plus, consultez la section [the section called "Regroupement d'applications" \(p. 163\)](#).

### Configuration de plateforme

Pour configurer les processus qui s'exécutent sur les instances de serveur dans votre environnement, incluez un [fichier Procfile \(p. 164\)](#) facultatif dans votre bundle de fichiers source. Un fichier [Procfile](#) est obligatoire si vous avez plus d'une application dans votre bundle de fichiers source.

Nous vous recommandons de toujours fournir un fichier [Procfile](#) dans le bundle de fichiers source avec votre application. De cette façon, vous contrôlez précisément les processus Elastic Beanstalk exécutés pour votre application.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour [modifier la configuration d'un environnement en cours d'exécution \(p. 671\)](#). Pour éviter de perdre la configuration

de votre environnement en le résilient, vous pouvez utiliser des [configurations enregistrées \(p. 779\)](#) pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des [fichiers de configuration \(p. 737\)](#). Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour de plus amples informations sur la priorité et les autres méthodes de modification des paramètres, veuillez consulter [Options de configuration \(p. 658\)](#).

## Configuration de votre environnement .NET Core sous Linux

Les paramètres de la plateforme .NET Core sous Linux vous permettent d'affiner le comportement de vos instances Amazon EC2. Vous pouvez modifier la configuration des instances Amazon EC2 de l'environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour permettre la rotation des journaux sur Amazon S3 et configurer des variables que votre application peut lire à partir de l'environnement.

Pour configurer votre environnement .NET Core sous Linux à l'aide de la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

## Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

## Propriétés de l'environnement

La section Environment Properties (Propriétés de l'environnement) vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

Dans l'environnement .NET Core sous Linux en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de `Environment.GetEnvironmentVariable("variable-name")`.

`name` ). Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
string endpoint = Environment.GetEnvironmentVariable("API_ENDPOINT");
```

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Espace de noms de la configuration .NET Core sous Linux

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

La plateforme .NET Core sous Linux prend en charge les options des espaces de noms suivants en plus des [options prises en charge pour tous les environnements Elastic Beanstalk \(p. 679\)](#) :

- `aws:elasticbeanstalk:environment:proxy` – Choisissez d'utiliser nginx ou de n'utiliser aucun serveur proxy. Les valeurs valides sont `nginx` ou `none`.

L'exemple de fichier de configuration suivant illustre l'utilisation d'options de configuration spécifiques à .NET Core sous Linux :

Example `.ebextensions/proxy-settings.config`

```
option_settings:  
  aws:elasticbeanstalk:environment:proxy:  
    ProxyServer: none
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Regroupement d'applications pour la plateforme .NET Core sous Linux

Vous pouvez exécuter à la fois des applications .NET Core dépendantes de l'exécution et autonomes sur AWS Elastic Beanstalk.

Une application dépendante de l'exécution utilise un environnement d'exécution .NET Core fourni par Elastic Beanstalk pour exécuter votre application. Elastic Beanstalk utilise le fichier `runtimeconfig.json` de votre bundle de fichiers source afin de déterminer l'environnement d'exécution à utiliser pour votre application. Elastic Beanstalk choisit l'environnement d'exécution compatible le plus récent disponible pour votre application.

Une application autonome inclut l'environnement d'exécution .NET Core, votre application et ses dépendances. Pour utiliser une version de l'environnement d'exécution .NET Core qu'Elastic Beanstalk n'inclut pas dans ses plateformes, fournissez une application autonome.

### Examples

Vous pouvez compiler à la fois des applications autonomes et dépendantes de l'environnement d'exécution à l'aide de la commande `dotnet publish`. Pour en savoir plus sur la publication d'applications .NET Core, veuillez consulter la [présentation de la publication d'applications .NET Core](#) dans la documentation .NET Core.

L'exemple de structure de fichiers suivant définit une application unique qui utilise un environnement d'exécution .NET Core fourni par Elastic Beanstalk.

```
### appsettings.Development.json
### appsettings.json
### dotnetcoreapp.deps.json
### dotnetcoreapp.dll
### dotnetcoreapp.pdb
### dotnetcoreapp.runtimeconfig.json
### web.config
### Procfile
### .ebextensions
### .platform
```

Vous pouvez inclure plusieurs applications dans votre bundle de fichiers source. L'exemple suivant définit deux applications à exécuter sur le même serveur Web. Pour exécuter plusieurs applications, vous devez inclure un [fichier Procfile \(p. 164\)](#) dans votre bundle de fichiers source. Pour un exemple d'application complet, veuillez consulter [dotnet-core-linux-multiple-apps.zip](#).

```
### DotnetMultipleApp1
# ### Amazon.Extensions.Configuration.SystemsManager.dll
# ### appsettings.Development.json
# ### appsettings.json
# ### AWSSDK.Core.dll
# ### AWSSDK.Extensions.NETCore.Setup.dll
# ### AWSSDK.SimpleSystemsManagement.dll
# ### DotnetMultipleApp1.deps.json
# ### DotnetMultipleApp1.dll
# ### DotnetMultipleApp1.pdb
# ### DotnetMultipleApp1.runtimeconfig.json
# ### Microsoft.Extensions.PlatformAbstractions.dll
# ### Newtonsoft.Json.dll
# ### web.config
### DotnetMultipleApp2
# ### Amazon.Extensions.Configuration.SystemsManager.dll
# ### appsettings.Development.json
# ### appsettings.json
# ### AWSSDK.Core.dll
# ### AWSSDK.Extensions.NETCore.Setup.dll
# ### AWSSDK.SimpleSystemsManagement.dll
# ### DotnetMultipleApp2.deps.json
# ### DotnetMultipleApp2.dll
# ### DotnetMultipleApp2.pdb
# ### DotnetMultipleApp2.runtimeconfig.json
# ### Microsoft.Extensions.PlatformAbstractions.dll
# ### Newtonsoft.Json.dll
# ### web.config
### Procfile
### .ebextensions
### .platform
```

## Utilisation d'un fichier Procfile pour configurer votre environnement .NET Core sous Linux

Pour exécuter plusieurs applications sur le même serveur web, vous devez inclure dans votre bundle de fichiers source un fichier `Procfile` qui indique à Elastic Beanstalk les applications à exécuter.

Nous vous recommandons de toujours fournir un fichier `Procfile` dans le bundle de fichiers source avec votre application. De cette façon, vous contrôlez précisément les processus Elastic Beanstalk qui s'exécutent pour votre application et les arguments que ces processus reçoivent.

L'exemple suivant utilise un fichier `Procfile` pour spécifier deux applications qu'Elastic Beanstalk doit exécuter sur le même serveur web.

#### Example Procfile

```
web: dotnet ./dotnet-core-app1/dotnetcoreapp1.dll
web2: dotnet ./dotnet-core-app2/dotnetcoreapp2.dll
```

Pour plus d'informations sur l'écriture et l'utilisation d'un `Procfile`, développez la section `Buildfile` et `Procfile` dans [the section called "Extension des plateformes Linux" \(p. 34\)](#).

## Configuration du serveur proxy pour votre environnement .NET Core sous Linux

AWS Elastic Beanstalk utilise [nginx](#) en tant que proxy inverse pour relayer les demandes à votre application. Elastic Beanstalk fournit une configuration nginx par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Par défaut, Elastic Beanstalk configure le serveur proxy nginx pour transmettre les demandes à votre application sur le port 5000. Vous pouvez remplacer le port par défaut en définissant la [propriété d'environnement \(p. 162\)](#) `PORT` sur le port que votre application écoute.

#### Note

Le port sur lequel votre application écoute n'affecte pas le port sur lequel le serveur nginx écoute les demandes de réception de l'équilibreur de charge.

Toutes les plateformes Amazon Linux 2 prennent en charge une configuration de proxy uniforme. Pour de plus amples informations sur la configuration du serveur proxy, développez la section Configuration du proxy inverse dans [the section called "Extension des plateformes Linux" \(p. 34\)](#).

L'exemple de fichier de configuration suivant étend la configuration nginx de votre environnement. La configuration dirige les demandes `/api` vers une deuxième application Web qui écoute sur le port 5200 du serveur Web. Par défaut, Elastic Beanstalk transmet les demandes à une seule application qui écoute sur le port 5000.

#### Example `01_custom.conf`

```
location /api {
    proxy_pass          http://127.0.0.1:5200;
    proxy_http_version  1.1;

    proxy_set_header    Upgrade $http_upgrade;
    proxy_set_header    Connection $http_connection;
    proxy_set_header    Host $host;
    proxy_cache_bypass $http_upgrade;
    proxy_set_header    X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header    X-Forwarded-Proto $scheme;
}
```

## Tutoriel : Déploiement d'une application ASP.NET Core sous Linux avec Elastic Beanstalk

Ce tutoriel décrit le processus de création d'une nouvelle application ASP.NET Core et de déploiement dans un environnement Amazon Linux 2 avec Elastic Beanstalk.

Dans ce didacticiel, vous utilisez d'abord l'outil de ligne de commande `dotnet` du SDK .NET Core pour effectuer les opérations suivantes :

- Générer une application qui sert les requêtes HTTP avec ASP.NET.
- Installer les dépendances d'exécution.
- Compiler et exécuter votre application Web localement.
- Publier vos artefacts d'application dans un répertoire de sortie. Les artefacts incluent le code source compilé, les dépendances d'exécution et les fichiers de configuration.

Ensuite, vous effectuez les opérations suivantes avec votre application nouvellement créée :

- Créez un bundle source d'application qui contient vos artefacts publiés.
- Créez un environnement Amazon Linux 2 et déployez votre application dans celui-ci avec Elastic Beanstalk.
- Ouvrez l'URL du site créé par Elastic Beanstalk pour exécuter votre application.

Le code source de l'application est disponible ici : [dotnet-core-linux-tutorial-source.zip](#).

Le bundle de fichiers source à déployer est disponible ici : [dotnet-core-linux-tutorial-bundle](#).

#### Sections

- [Prerequisites \(p. 166\)](#)
- [Générer un projet de base .NET en tant qu'application web \(p. 167\)](#)
- [Lancement d'un environnement Elastic Beanstalk et déploiement de votre application \(p. 169\)](#)
- [Cleanup \(p. 170\)](#)
- [Étapes suivantes \(p. 171\)](#)

## Prerequisites

Ce didacticiel utilise le kit SDK .NET Core pour générer une application web .NET Core de base, l'exécuter localement et créer un package pouvant être déployé.

#### Requirements

- .NET Core (x64) 2.1.19 ou ultérieur

Pour installer le kit SDK .NET core

1. Téléchargez le programme d'installation à partir de [microsoft.com/net/core](#). Choisissez votre plateforme de développement. Choisissez Télécharger le SDK .NET Core.
2. Exécutez le programme d'installation et suivez les instructions.

#### Note

Bien que les exemples de ce didacticiel soient des listes de la ligne de commande Windows, le SDK .NET Core prend en charge les plateformes de développement sur plusieurs systèmes d'exploitation. Les commandes `dotnet` affichées dans ce didacticiel sont cohérentes sur différentes plateformes de développement.

Ce tutoriel utilise un utilitaire ZIP de ligne de commande pour créer un bundle de fichiers source que vous pouvez déployer sur Elastic Beanstalk. Pour utiliser la commande `zip` dans Windows, vous pouvez installer `UnxUtils`. (`UnxUtils` est une collection légère d'utilitaires de ligne de commande utiles comme `zip` et `ls`.) Sinon, vous pouvez utiliser [l'Explorateur Windows \(p. 416\)](#) ou tout autre utilitaire ZIP pour créer des archives de bundle source.

### Pour installer UnxUtils

1. Téléchargez [UnxUtils](#).
2. Extrayez l'archive dans un répertoire local. Par exemple, C:\Program Files (x86).
3. Ajoutez le chemin d'accès aux fichiers binaires à votre variable utilisateur PATH sous Windows. Par exemple, C:\Program Files (x86)\UnxUtils\usr\local\wbin.
  - a. Appuyez sur la touche Windows et entrez **environment variables**.
  - b. Choisissez Modifier les variables d'environnement pour votre compte.
  - c. Choisissez PATH, puis Modifier.
  - d. Ajoutez des chemins d'accès dans le champ Valeur de la variable, en les séparant par des points virgules. Par exemple: **C:\item1\path;C:\item2\path**
  - e. Choisissez OK deux fois pour appliquer les nouveaux paramètres.
  - f. Fermez toutes les fenêtres d'invite de commande en cours d'exécution, puis rouvrez une fenêtre d'invite de commande.
4. Ouvrez une nouvelle fenêtre d'invite de commande et exécutez la commande zip pour vérifier qu'elle fonctionne.

```
> zip -h
Copyright (C) 1990-1999 Info-ZIP
Type 'zip "-L"' for software license.
...
```

## Générer un projet de base .NET en tant qu'application web

Utilisez l'outil de ligne de commande dotnet pour générer un nouveau projet d'application web C# .NET Core et l'exécuter localement. L'application web .NET Core par défaut affiche Hello World!.

### Pour générer un nouveau projet .NET core

1. Ouvrez une nouvelle fenêtre d'invite de commande et accédez à votre dossier utilisateur.

```
> cd %USERPROFILE%
```

2. Utilisez la commande dotnet new pour générer un nouveau projet .NET Core.

```
C:\Users\username> dotnet new web -o dotnet-core-tutorial
The template "ASP.NET Core Empty" was created successfully.

Processing post-creation actions...
Running 'dotnet restore' on dotnet-core-tutorial\dotnet-core-tutorial.csproj...
  Determining projects to restore...
  Restored C:\Users\username\dotnet-core-tutorial\dotnet-core-tutorial.csproj (in 154
ms).

Restore succeeded.
```

### Pour exécuter le site web localement

1. Utilisez la commande dotnet restore pour installer les dépendances.

```
C:\Users\username> cd dotnet-core-tutorial
C:\Users\username\dotnet-core-tutorial> dotnet restore
```

```
Determining projects to restore...
All projects are up-to-date for restore.
```

- Utilisez la commande `dotnet run` pour créer et exécuter l'application localement.

```
C:\Users\username\dotnet-core-tutorial> dotnet run
info: Microsoft.Hosting.Lifetime[0]
      Now listening on: https://localhost:5001
info: Microsoft.Hosting.Lifetime[0]
      Now listening on: http://localhost:5000
info: Microsoft.Hosting.Lifetime[0]
      Application started. Press Ctrl+C to shut down.
info: Microsoft.Hosting.Lifetime[0]
      Hosting environment: Development
info: Microsoft.Hosting.Lifetime[0]
      Content root path: C:\Users\username\dotnet-core-tutorial
```

- Ouvrez `localhost:5000` pour afficher le site à partir de votre navigateur web par défaut.

L'application retourne `Hello World!`, qui est affiché sur votre navigateur web.

Pour exécuter l'application sur un serveur web, vous devez regrouper le code source compilé avec un fichier de configuration `web.config` et les dépendances d'exécution. L'outil `dotnet` fournit une commande `publish` qui rassemble ces fichiers dans un répertoire basé sur la configuration dans `dotnet-core-tutorial.csproj`.

#### Pour créer votre site web

- Utilisez la commande `dotnet publish` pour générer la sortie du code compilé et des dépendances dans un dossier nommé `site`.

```
C:\users\username\dotnet-core-tutorial> dotnet publish -o site
Microsoft (R) Build Engine version 16.7.0-preview-20360-03+188921e2f for .NET
Copyright (C) Microsoft Corporation. All rights reserved.

Determining projects to restore...
All projects are up-to-date for restore.
dotnet-core-tutorial -> C:\Users\username\dotnet-core-tutorial\bin\Debug\netcoreapp3.1\dotnet-core-tutorial.dll
dotnet-core-tutorial -> C:\Users\username\dotnet-core-tutorial\site\
```

#### Pour créer un bundle de fichiers source

- Utilisez la commande `zip` pour créer un bundle de fichiers source nommé `dotnet-core-tutorial.zip`.

Le bundle source contient tous les fichiers publiés dans le dossier du site.

#### Note

Si vous utilisez un utilitaire ZIP différent, veillez à ajouter tous les fichiers au dossier racine de l'archive ZIP résultante. Ceci est requis pour un déploiement réussi de l'application dans votre environnement Elastic Beanstalk.

```
C:\users\username\dotnet-core-tutorial> cd site
C:\users\username\dotnet-core-tutorial\site>zip ../dotnet-core-tutorial.zip *
```

```
adding: appsettings.Development.json (164 bytes security) (deflated 38%)
adding: appsettings.json (164 bytes security) (deflated 39%)
adding: dotnet-core-tutorial.deps.json (164 bytes security) (deflated 93%)
adding: dotnet-core-tutorial.dll (164 bytes security) (deflated 58%)
adding: dotnet-core-tutorial.exe (164 bytes security) (deflated 57%)
adding: dotnet-core-tutorial.pdb (164 bytes security) (deflated 48%)
adding: dotnet-core-tutorial.runtimeconfig.json (164 bytes security) (deflated 33%)
adding: web.config (164 bytes security) (deflated 41%)
```

#### Note

Dans ce didacticiel, vous n'exécutez qu'une seule application sur le serveur web, de sorte qu'un fichier Procfile n'est pas requis dans votre bundle source. Toutefois, pour déployer plusieurs applications sur le même serveur Web, vous devez inclure un fichier Procfile. Pour de plus amples informations, veuillez consulter [Utilisation d'un fichier Procfile pour configurer votre environnement .NET Core sous Linux \(p. 164\)](#).

## Lancement d'un environnement Elastic Beanstalk et déploiement de votre application

Utilisez la console Elastic Beanstalk pour lancer un environnement Elastic Beanstalk et déployer le bundle de fichiers source.

Vous pouvez télécharger le bundle de fichiers source ici : [dotnet-core-linux-tutorial-bundle](#)

Pour lancer un environnement et déployer votre code (console)

1. Ouvrez la console Elastic Beanstalk avec ce lien préconfiguré : [console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez .NET Core on Linux (.NET Core sous Linux).
3. Choisissez Local file (Fichier local), Choose file (Choisir un fichier), puis ouvrez le bundle source.
4. Choisissez Vérifier et lancer.
5. Vérifiez les paramètres disponibles et choisissez Créer une application. L'application écrit simplement le message `Hello World!` à la réponse et le renvoie.

Il faut environ 10 minutes pour créer l'environnement et déployer votre code.

Le lancement d'un environnement crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

#### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021. Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk. Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibrer de charge – Équilibrer de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrer de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrer de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrer de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme [\*sous-domaine.région.elasticbeanstalk.com\*](#).

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibrers de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

## Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez peut-être gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger sur la console Elastic Beanstalk manuellement. [L'interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de l'interface de ligne de commande.

Si vous utilisez Visual Studio pour développer votre application, vous pouvez également utiliser AWS Toolkit for Visual Studio afin de déployer des modifications dans votre code, gérer vos environnements Elastic Beanstalk et gérer d'autres ressources AWS. Pour plus d'informations, consultez [. AWS Toolkit for Visual Studio \(p. 224\)](#).

À des fins de développement et de test, vous pouvez envisager de tirer parti de la fonction de déploiement d'Elastic Beanstalk pour ajouter directement une instance de base de données gérée à votre environnement. Pour savoir comment configurer une base de données dans votre environnement, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, nous vous recommandons de [configurer un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et d'activer [HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## L'AWS Toolkit for Visual Studio - Utilisation de .Net Core

L'AWS Toolkit for Visual Studio est un plugin pour l'IDE Visual Studio. Avec la boîte à outils, vous pouvez déployer et gérer des applications dans Elastic Beanstalk pendant que vous travaillez dans votre environnement Visual Studio.

Cette rubrique explique comment vous pouvez effectuer les tâches suivantes à l'aide de AWS Toolkit for Visual Studio:

- Créer une application web ASP.NET Core à l'aide d'un modèle Visual Studio.
- Créer un environnement Elastic Beanstalk Amazon Linux.
- Déployer l'application web ASP.NET Core dans le nouvel environnement Amazon Linux.

Cette rubrique explore également la façon dont vous pouvez utiliser l'AWS Toolkit for Visual Studio pour gérer vos environnements applicatifs Elastic Beanstalk et contrôler l'état de votre application.

### Sections

- [Prerequisites \(p. 172\)](#)
- [Créer un nouveau projet d'application \(p. 172\)](#)
- [Création d'un environnement Elastic Beanstalk et déploiement de votre application \(p. 173\)](#)
- [Résiliation d'un environnement \(p. 177\)](#)
- [Gestion de vos environnements d'application Elastic Beanstalk \(p. 178\)](#)
- [Surveillance de l'intégrité d'une application \(p. 189\)](#)

## Prerequisites

Avant de commencer ce didacticiel, vous devez installer le AWS Toolkit for Visual Studio. Pour obtenir des instructions, veuillez consulter [Configuration de l'AWS Toolkit for Visual Studio](#).

Si vous n'avez jamais utilisé la boîte à outils, vous devez l'installer, puis y enregistrer vos informations d'identification AWS. Pour de plus amples informations sur ce point, veuillez consulter [Fournire les informations d'identification AWS](#).

## Créer un nouveau projet d'application

Si vous n'avez pas de projet d'application .NET Core dans Visual Studio, vous pouvez facilement en créer un en utilisant l'un des modèles de projet Visual Studio.

Pour créer un projet d'application web ASP.NET Core

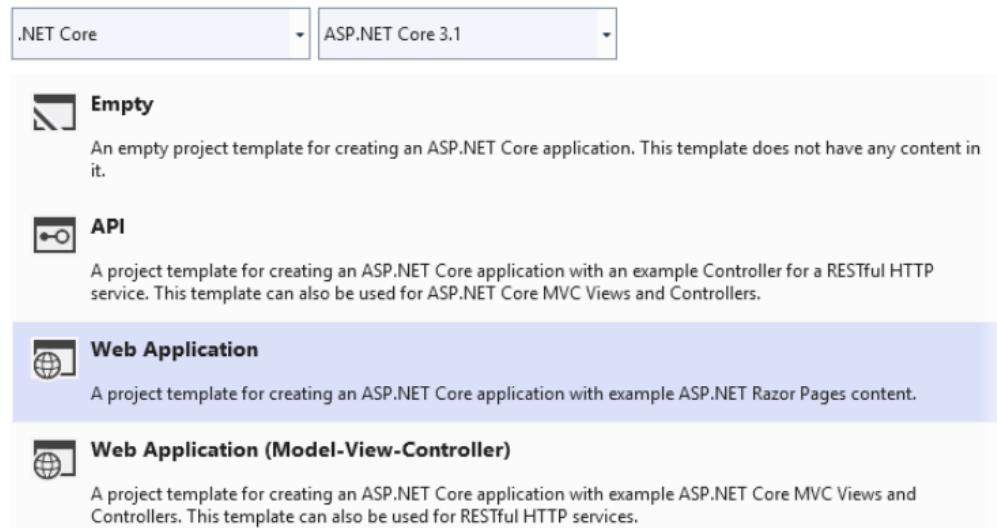
1. Dans Visual Studio, dans le menu File (Fichier), choisissez New (Nouveau), puis choisissez Project (Projet).
2. Dans la boîte de dialogue Create a new project (Créer un nouveau projet) sélectionnez C#, Linux, puis Cloud.
3. Dans la liste des modèles de projet qui s'affiche, sélectionnez ASP.NET Core Web Application (Application web ASP.NET Core), puis sélectionnez Next (Suivant).

### Note

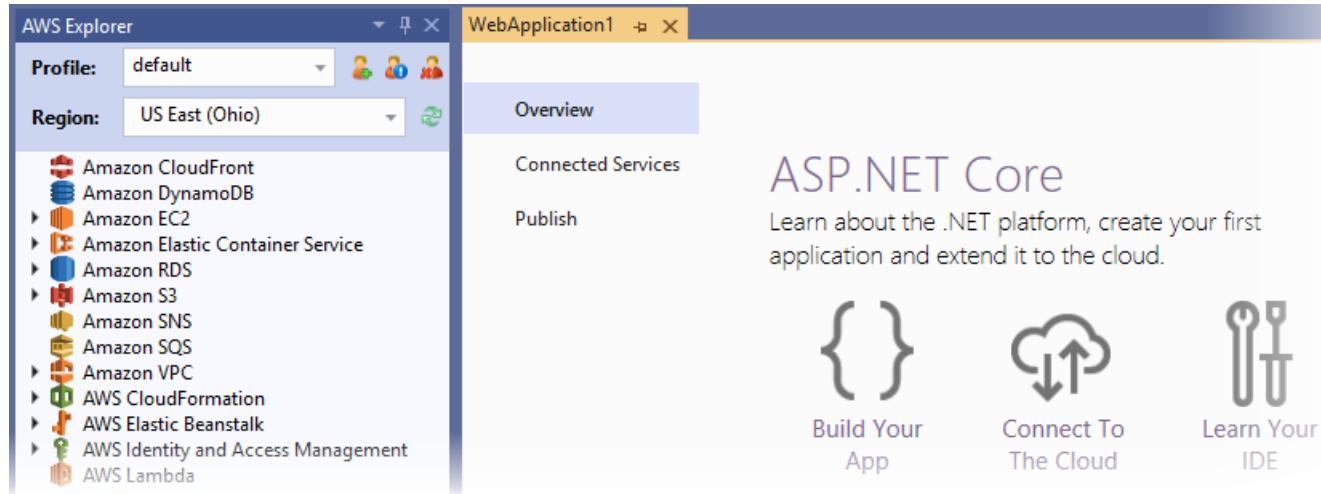
Si vous ne voyez pas l'application web ASP.NET Core répertoriée dans les modèles de projet, vous pouvez l'installer lorsque vous êtes dans Visual Studio.

1. Faites défiler jusqu'au bas de la liste des modèles et sélectionnez le lien Installer d'autres outils et fonctionnalités qui se trouve sous la liste des modèles.
2. Si vous êtes invité à autoriser l'application Visual Studio à apporter des modifications à votre appareil, sélectionnez Yes (Oui).
3. Choisissez l'onglet Workloads (Charges de travail), puis sélectionnez ASP.NET and web development (ASP.NET et développement web).
4. Sélectionnez le bouton Modify (Modifier). Le Visual Studio Installer (programme d'installation de Visual Studio) installe le modèle de projet.
5. Une fois le programme d'installation terminé, quittez le panneau pour revenir à l'endroit où vous vous êtes arrêté dans Visual Studio.
4. Dans la boîte de dialogue Configure your new project (Configurer votre nouveau projet) saisissez un nom de projet. Le nom de la solution est par défaut le nom de votre projet. Ensuite, choisissez Créer.
5. Dans la boîte de dialogue Create a new ASP.NET Core web application (Créer une nouvelle application web ASP.NET Core) sélectionnez .NET Core, puis ASP.NET Core 3.1. Dans la liste des types d'application affichée, sélectionnez Web Application (Application web), puis cliquez sur le bouton Create (Créer).

## Create a new ASP.NET Core web application



Visual Studio affiche la boîte de dialogue Creating Project (Création du projet) au moment de la création de votre application. Une fois la génération de votre application terminée, Visual Studio affiche un panneau avec le nom de votre application.



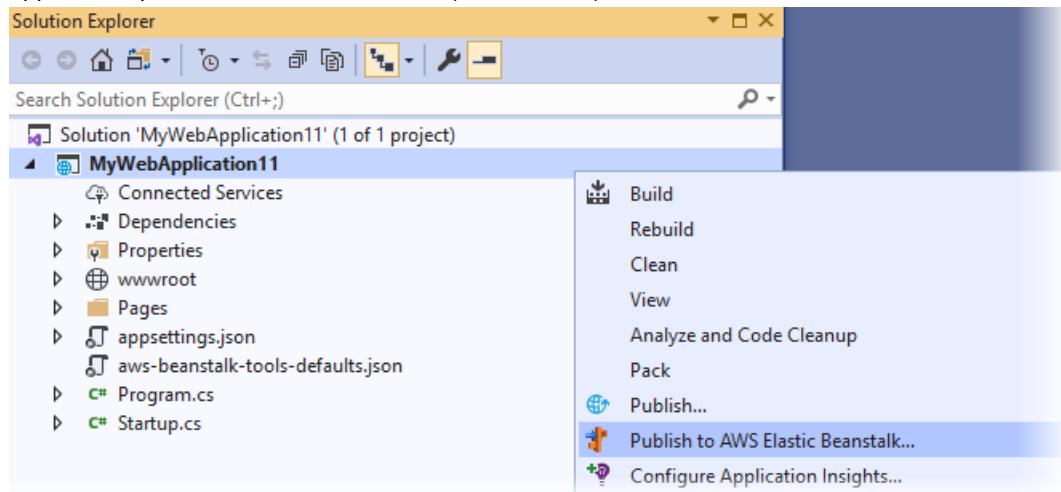
## Création d'un environnement Elastic Beanstalk et déploiement de votre application

Cette section décrit comment créer un environnement Elastic Beanstalk pour votre application et déployer votre application dans cet environnement.

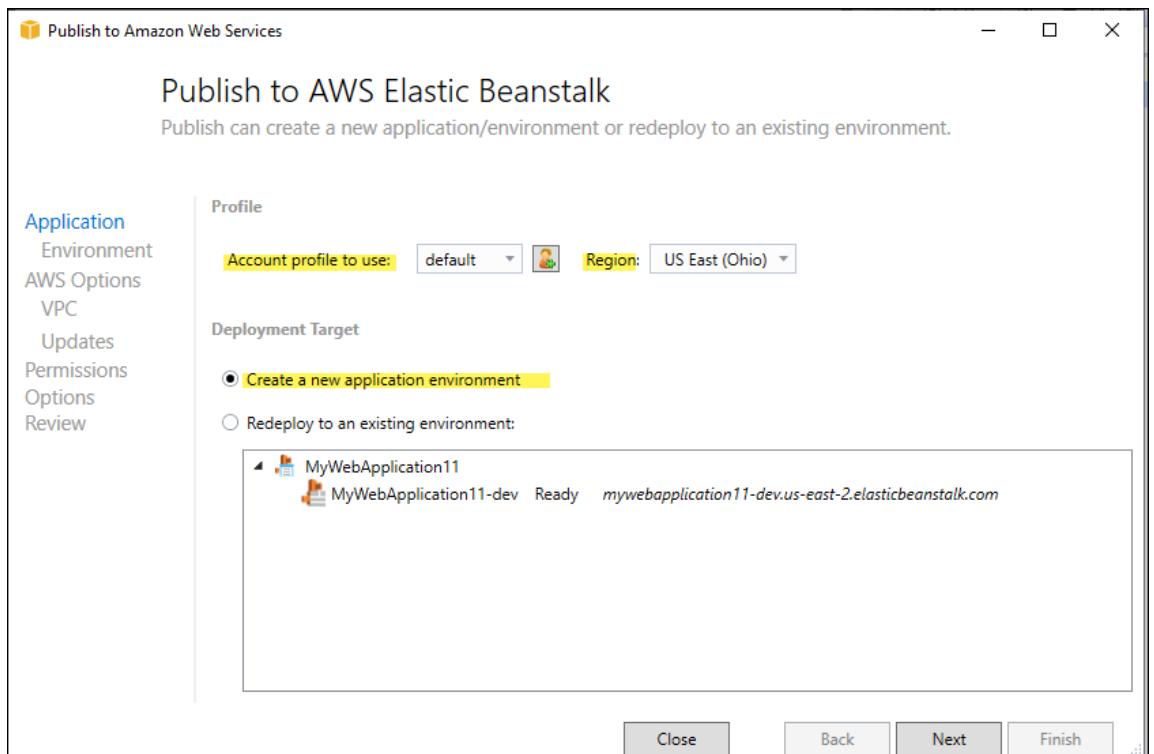
Pour créer un environnement et déployer votre application

1. Dans Visual Studio, sélectionnez View (Afficher), puis Solution Explorer (Explorateur de solutions).

2. Dans Solution Explorer (Explorateur de solutions), ouvrez le menu contextuel (clic droit) de votre application, puis sélectionnez Publish to (Publier dans) AWS Elastic Beanstalk.



3. Dans l'assistant Publish to AWS Elastic Beanstalk (Publier dans AWS), saisissez les informations de votre compte.
  - a. Pour Account profile to use (Profil de compte à utiliser), sélectionnez votre compte par défaut ou cliquez sur l'icône Add another account (Ajouter un autre compte) pour entrer les informations d'un nouveau compte.
  - b. Pour Region, sélectionnez la région où vous souhaitez déployer votre application. Pour de plus amples informations sur les régions AWS disponibles, veuillez consulter [AWS Elastic Beanstalk Endpoints and Quotas](#) (Points de terminaison et quotas AWS Elastic Beanstalk) dans la référence générale AWS. Si vous sélectionnez une région qui n'est pas prise en charge par Elastic Beanstalk, l'option de déploiement sur Elastic Beanstalk devient indisponible.
  - c. Sélectionnez Create a new application environment (Créer un nouvel environnement d'application), puis choisissez Next (Suivant).

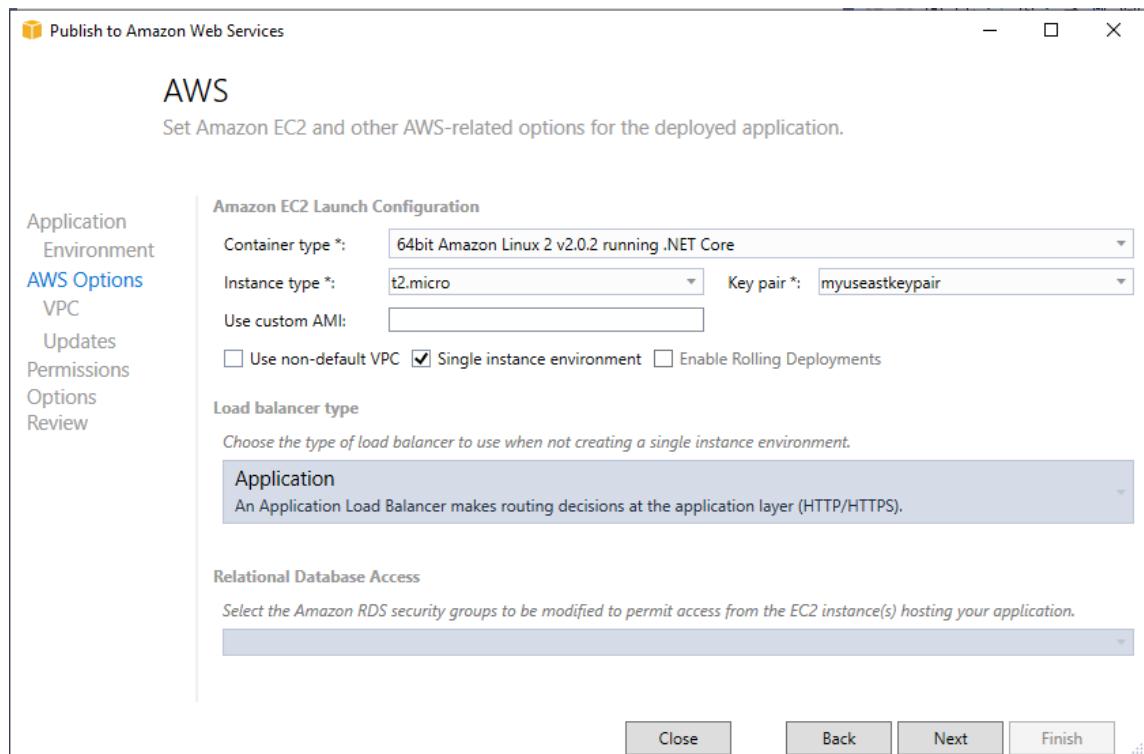


4. Dans la boîte de dialogue Application Environment (Environnement d'application) entrez les détails de votre nouvel environnement d'application.
5. Dans la boîte de dialogue des options AWS suivante, définissez les options Amazon EC2 et d'autres options associées à AWS pour l'application déployée.
  - a. Pour le Container type (Type de conteneur), sélectionnez 64bit Amazon Linux 2 v<n.n.n> exécutant .NET Core.

#### Note

Nous vous recommandons de sélectionner la version actuelle de la plateforme de Linux. Cette version contient les derniers correctifs de sécurité et de bogues inclus dans notre dernière Amazon Machine Image (AMI).

- b. Pour Instance Type (Type d'instance), sélectionnez t2.micro. (Le choix d'un type d'instance micro réduit le coût associé à l'exécution de l'instance.)
- c. Pour Key pair (Paire de clés), sélectionnez Create new key pair (Créer une paire de clés). Entrez un nom pour la nouvelle paire de clés, puis choisissez OK. (Dans cet exemple, nous utilisons **myuseastkeypair**.) Une paire de clés permet un accès bureau à distance à vos instances Amazon EC2. Pour de plus amples informations sur les paires de clés Amazon EC2, veuillez consulter [Utilisation des informations d'identification](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.
- d. Si l'application est simple et génère un faible trafic, sélectionnez Single instance environment (Environnement à une seule instance). Pour plus d'informations, consultez [Types d'environnement \(p. 519\)](#)
- e. Sélectionnez Suivant.



Pour plus d'informations sur les options AWS qui ne sont pas utilisées dans cet exemple, considérez les pages suivantes :

- Pour Use custom AMI (Utiliser une AMI personnalisée), veuillez consulter [Utilisation d'une Amazon Machine Image \(AMI\) personnalisée \(p. 787\)](#).
  - Si vous ne sélectionnez pas Single instance environment (Environnement à une seule instance), vous devez choisir un type d'équilibrage de charge. Pour plus d'informations, consultez [Équilibrage de charge pour votre environnement Elastic Beanstalk \(p. 562\)](#).
  - Elastic Beanstalk utilise la configuration par défaut [Amazon VPC](#) (Amazon Virtual Private Cloud) si vous n'avez pas choisi Use non-default VPC (Utiliser un VPC autre que le VPC par défaut). Pour plus d'informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#).
  - Le choix de l'option Activer les déploiements mobiles divise un déploiement en lots afin d'éviter les temps d'arrêt potentiels pendant les déploiements. Pour de plus amples informations, veuillez consulter [Déploiement d'applications dans des environnements Elastic Beanstalk \(p. 476\)](#).
  - L'option Relational Database Access (Accès de base de données relationnelle) vous permet de connecter votre environnement Elastic Beanstalk à une base de données Amazon RDS précédemment créée avec des groupes de sécurité de base de données Amazon RDS. Pour de plus amples informations, veuillez consulter [Contrôle d'accès par groupe de sécurité](#) dans le Guide de l'utilisateur Amazon RDS.
6. Sélectionnez Next (Suivant) dans la boîte de dialogue Permissions (Autorisations).
  7. Sélectionnez Next (Suivant) dans la boîte de dialogue Applications Options (Options d'applications).
  8. Passez en revue vos options de déploiement. Après avoir vérifié que vos paramètres sont corrects, sélectionnez Déployer.

Votre application Web ASP.NET Core est exportée en tant que fichier de déploiement Web. Votre fichier est ensuite chargé dans Amazon S3 et enregistré en tant que nouvelle version d'application auprès

d'Elastic Beanstalk. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. L'information Status (Statut) de votre environnement s'affiche sur l'onglet Env:<nom de l'environnement>. Une fois que le statut devient Environment is healthy (L'environnement est sain), sélectionnez l'adresse URL pour lancer l'application web.

Event Time	Event Type	Version Label	Event Details
7/21/2020 2:56:54 AM	INFO		Successfully launched environment: MyWebApplication11
7/21/2020 2:56:54 AM	INFO		Application available at mywebapplication11-dev.us-east-
7/21/2020 2:56:45 AM	INFO	v20200721020104	Added EC2 instance 'i-00c5680f13fc6f089' to Auto Scaling
7/21/2020 2:56:45 AM	INFO	v20200721020104	Environment health has been set to GREEN
7/21/2020 2:56:45 AM	INFO	v20200721020104	Adding instance 'i-00c5680f13fc6f089' to your environment
7/21/2020 2:56:18 AM	INFO		Waiting for EC2 instances to launch. This may take a few minutes.
7/21/2020 2:54:58 AM	INFO		Created EIP: 3.14.235.39
7/21/2020 2:54:42 AM	INFO		Created security group named: awseb-e-ffi5z3mn6m-stac
7/21/2020 2:54:24 AM	INFO		Using elasticbeanstalk-us-east-2-164656829171 as Amazon VPC
7/21/2020 2:54:23 AM	INFO		createEnvironment is starting.

## Résiliation d'un environnement

Vous pouvez résilier un environnement en cours d'exécution à l'aide d'AWS Toolkit for Visual Studio afin d'éviter de payer des frais pour des ressources AWS inutilisées.

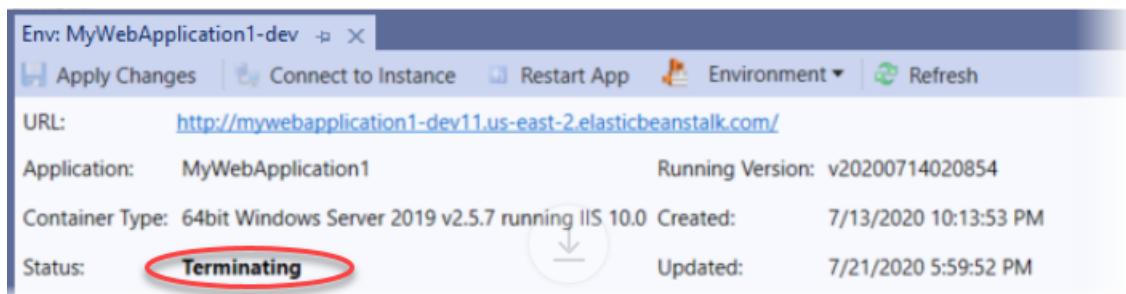
### Note

Vous pouvez toujours lancer un nouvel environnement en utilisant la même version ultérieurement.

### Pour résilier un environnement

1. Développez le nœud Elastic Beanstalk et le nœud de l'application. Dans AWS Explorer (Explorateur AWS), ouvrez le menu contextuel (clic droit) de votre environnement d'application et sélectionnez Terminate Environment (Résilier l'environnement).
2. Lorsque vous y êtes invité, sélectionnez Oui (Yes) afin de confirmer que vous souhaitez résilier l'environnement. Il faut quelques minutes à Elastic Beanstalk pour résilier les ressources AWS en cours d'exécution dans l'environnement.

L'information relative à Status (Statut) de votre environnement sous l'onglet Env:<nom de l'environnement> deviendra Terminating (Résiliation en cours), puis Terminated (Résilié).



#### Note

Lorsque vous résiliez votre environnement, le CNAME associé à l'environnement résilié devient disponible pour que tout le monde puisse l'utiliser.

## Gestion de vos environnements d'application Elastic Beanstalk

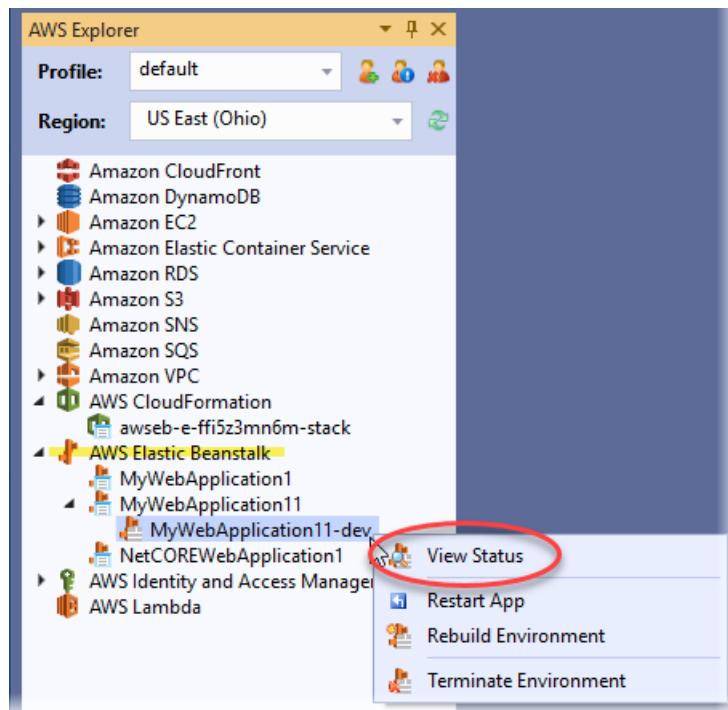
Avec AWS Toolkit for Visual Studio et la console de gestion AWS, vous pouvez modifier l'approvisionnement et la configuration des ressources AWS utilisées par les environnements de votre application. Pour plus d'informations sur la façon de gérer les environnements de votre application à l'aide de la console de gestion AWS, consultez [Gestion des environnements \(p. 428\)](#). Cette section décrit les paramètres de service spécifiques que vous pouvez modifier dans AWS Toolkit for Visual Studio dans le cadre de la configuration d'environnement de votre application.

### Modification des paramètres de configuration de l'environnement

Lorsque vous déployez votre application, Elastic Beanstalk configure plusieurs services AWS de cloud computing. Vous pouvez contrôler la façon dont ces services individuels sont configurés à l'aide d'AWS Toolkit for Visual Studio.

Pour modifier les paramètres d'environnement d'une application

1. Dans Visual Studio, dans le menu File (Fichier), choisissez AWS Explorer (Explorateur AWS).
2. Développez le nœud Elastic Beanstalk et le nœud de votre application. Ouvrez le menu contextuel (clic droit) de votre environnement d'application et sélectionnez View Status (Afficher le statut).



Vous pouvez à présent configurer des paramètres pour les éléments suivants :

- AWS X-Ray
- de bases de données
- Équilibrer de charge (s'applique uniquement aux environnements à plusieurs instances)
- Auto Scaling (s'applique uniquement aux environnements à plusieurs instances)
- Notifications
- Conteneur
- Options de configuration avancées

## Configuration d'AWS X-Ray à l'aide d'AWS Toolkit for Visual Studio

AWS X-Ray fournit des fonctions de suivi des demandes, de collecte des exceptions et de profilage. Avec le panneau AWS X-Ray, vous pouvez activer ou désactiver X-Ray pour votre application. Pour plus d'informations sur X-Ray, veuillez consulter le [Guide du développeur AWS X-Ray](#).

AWS X-Ray Console. To learn how to instrument your .NET application visit the [AWS X-Ray SDK for .NET GitHub repository](#).'"/>

AWS X-Ray is a service that collects data about requests that your application serves, and provides tools you can use to view, filter, and gain insights into that data to identify issues and opportunities for optimization. For any traced request to your application, you can see detailed information not only about the request and response, but also about calls that your application makes to downstream AWS resources, microservices, databases and HTTP web APIs.

Scorekeep 200 39.0 ms    Scorekeep 200 6.0 ms    Scorekeep 200 5.0 ms    Scorekeep 200 5.0 ms    Scorekeep 200 6.0 ms    GameModel saveGame 200 14.0 ms    Scorekeep 200 5.0 ms    Scorekeep 200 8.0 ms

PUT scorekeep elasticbean  
GetItem: scorekeep-game  
GetItem: scorekeep-session  
GetItem: scorekeep-game  
UpdateItem: scorekeep-state  
UpdateItem: scorekeep-session  
UpdateItem: scorekeep-game

Enable AWS X-Ray true

To see your application's service map and traces visit the [AWS X-Ray Console](#).

To learn how to instrument your .NET application visit the [AWS X-Ray SDK for .NET GitHub repository](#).

## Configuration des instances EC2 à l'aide d'AWS Toolkit for Visual Studio

Vous pouvez utiliser Amazon Elastic Compute Cloud (Amazon EC2) pour lancer et gérer des instances de serveur dans les centres de données d'Amazon. Vous pouvez utiliser des instances de serveur Amazon EC2 à tout moment, aussi longtemps que vous le souhaitez et pour tout motif (dans le cadre d'une utilisation légale). Les instances sont disponibles dans différentes tailles et configurations. Pour de plus amples informations, veuillez consulter [Amazon EC2](#).

Vous pouvez modifier la configuration de votre instance Amazon EC2 à l'aide de l'onglet Server (Serveur) de votre environnement d'application dans AWS Toolkit for Visual Studio.

These settings allow you to control your environment's servers and enable login.

\*EC2 Instance Type t2.micro

\*EC2 Security Group awseb-e-ffi5z3mn6m-stack-AWSEBSecurityGroup-18TE8OFU

\*Existing Key Pair myuseastkeypair

\*Monitoring Interval 5 minute

\*AMI ID ami-005469737bfd6c6e

Note: "It may take a few minutes to see changes to these options take effect in your environment."

### Types d'instances Amazon EC2

Instance type (Type d'instance) affiche les types d'instance disponibles pour votre application Elastic Beanstalk. Changez le type d'instance pour sélectionner un serveur dont les caractéristiques (y compris la taille de la mémoire et la puissance d'UC) sont les mieux adaptées à votre application. Par exemple, les applications exécutant des opérations intensives et de longue durée peuvent nécessiter plus de puissance de calcul et de mémoire.

Pour de plus amples informations sur les types d'instance Amazon EC2 disponibles pour votre application Elastic Beanstalk, veuillez consulter [Types d'instances](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

## Groupes de sécurité Amazon EC2

Vous pouvez contrôler l'accès à votre application Elastic Beanstalk par le biais d'un groupe de sécurité Amazon EC2. Un groupe de sécurité définit les règles de pare-feu de vos instances. Ces règles déterminent le trafic réseau entrant qui doit être acheminé vers votre instance. Tout autre trafic entrant est éliminé. Vous pouvez modifier les règles pour un groupe à la fois. Les nouvelles règles sont appliquées automatiquement pour toutes les instances en cours d'exécution et les instances lancées par la suite.

Vous pouvez spécifier quels groupes de sécurité Amazon EC2 contrôlent l'accès à votre application Elastic Beanstalk. Pour ce faire, entrez les noms de groupes de sécurité Amazon EC2 spécifiques (en séparant plusieurs groupes de sécurité par des virgules) dans la zone de texte Groupes de sécurité EC2. Pour ce faire, vous pouvez utiliser la Console de gestion AWS ou AWS Toolkit for Visual Studio.

Pour créer un groupe de sécurité à l'aide d'AWS Toolkit for Visual Studio

1. Dans Visual Studio, dans AWS Explorer (Explorateur AWS), développez le nœud Amazon EC2, puis sélectionnez Security Groups (Groupes de sécurité).
2. Cliquez sur Create Security Group (Créer un groupe de sécurité) et entrez un nom et une description pour votre groupe de sécurité.
3. Sélectionnez OK.

Pour de plus amples informations sur les groupes de sécurité Amazon EC2, veuillez consulter [Utilisation des groupes de sécurité](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

## Paires de clés Amazon EC2

Vous pouvez vous connecter en toute sécurité aux instances Amazon EC2 allouées pour votre application Elastic Beanstalk avec une paire de clés Amazon EC2.

### Important

Vous devez créer une paire de clés Amazon EC2 et configurer vos instances Amazon EC2 allouées par Elastic Beanstalk pour pouvoir accéder à ces instances. Vous pouvez créer votre paire de clés à l'aide de l'assistant Publish to AWS (Publier dans AWS) à l'intérieur d'AWS Toolkit for Visual Studio lorsque vous déployez votre application sur Elastic Beanstalk. Si vous souhaitez créer des paires de clés supplémentaires à l'aide de la boîte à outils, procédez comme suit. Sinon, vous pouvez configurer vos paires de clés Amazon EC2 via la [console de gestion AWS](#). Pour obtenir des instructions sur la création d'une paire de clés pour Amazon EC2, veuillez consulter le [Guide de démarrage Amazon Elastic Compute Cloud](#).

La zone de texte Existing Key Pair (Paire de clés existantes) vous permet de spécifier le nom d'une paire de clés Amazon EC2 que vous pouvez utiliser pour vous connecter en toute sécurité aux instances Amazon EC2 qui exécutent votre application Elastic Beanstalk.

Pour spécifier le nom d'une paire de clés Amazon EC2

1. Développez le nœud Amazon EC2 et sélectionnez Key Pairs (Paires de clés).
2. Sélectionnez Create Key Pair (Créer une paire de clés) et saisissez le nom de la paire de clés.
3. Sélectionnez OK.

Pour de plus amples informations sur les paires de clés Amazon EC2, veuillez consulter [Utilisation des informations d'identification Amazon EC2](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud. Pour de plus amples informations sur la connexion à des instances Amazon EC2, veuillez consulter

## Intervalle de surveillance

Par défaut, seules les métriques de base d'Amazon Cloudwatch sont activées. Elles renvoient des données toutes les cinq minutes. Vous pouvez activer des métriques CloudWatch plus détaillées en une minute

en sélectionnant 1 minute pour la Monitoring Interval (Intervalle de surveillance) dans la section Server (Serveur) de l'onglet Configuration correspondant à votre environnement dans AWS Toolkit for Eclipse.

#### Note

Des frais de service Amazon CloudWatch peuvent s'appliquer aux métriques d'intervalle d'une minute. Pour de plus amples informations, veuillez consulter [Amazon CloudWatch](#).

#### ID d'AMI personnalisé

Vous pouvez remplacer l'AMI par défaut utilisée pour vos instances Amazon EC2 par votre propre AMI personnalisée en saisissant l'identifiant de cette dernière dans la zone Custom AMI ID (ID d'AMI personnalisée) de la section Server (Serveur) de l'onglet Configuration correspondant à votre environnement dans AWS Toolkit for Eclipse.

#### Important

L'utilisation de votre propre image AMI est une tâche avancée qui doit être effectuée avec soin. Si vous avez besoin d'une AMI personnalisée, nous vous recommandons de démarrer par l'AMI Elastic Beanstalk par défaut, puis de la modifier. Pour être considérées saines par Elastic Beanstalk, les instances Amazon EC2 doivent respecter un ensemble de conditions, y compris disposer d'un gestionnaire hôte en cours d'exécution. Si ces conditions ne sont pas satisfaites, il se peut que votre environnement ne fonctionne pas correctement.

### Configuration d'Elastic Load Balancing à l'aide d'AWS Toolkit for Visual Studio

Elastic Load Balancing est un service d'Amazon Web Services qui vous aide à améliorer la disponibilité et l'évolutivité de votre application. Ce service vous permet de facilement répartir les charges d'application entre au moins deux instances Amazon EC2. Elastic Load Balancing améliore la disponibilité en fournissant une redondance supplémentaire et prend en charge la croissance du trafic pour votre application.

Avec Elastic Load Balancing, vous pouvez répartir et équilibrer automatiquement le trafic d'applications entrant entre toutes vos instances en cours d'exécution. Vous pouvez également ajouter facilement de nouvelles instances lorsque l'augmentation de la capacité de votre application est requise.

Elastic Beanstalk fournit automatiquement Elastic Load Balancing lorsque vous déployez une application. Vous pouvez modifier la configuration d'instance Amazon EC2 de l'environnement Elastic Beanstalk avec l'onglet Load Balancer (ÉquilibrEUR de charge) à l'intérieur de l'onglet de l'environnement de votre application dans AWS Toolkit for Visual Studio.

Events These settings allow you to control the behavior of your environment's load balancer.

Monitoring

Resources

AWS X-Ray

Server

**Load Balancer**

Auto Scaling

Notifications

Container

Advanced

Logs

HTTP Listener Port:

HTTPS Listener Port:

SSL Certificate ID

Application Health Check: /

Health Check Interval (seconds):

Health Check Timeout (seconds):

Healthy Check Count Threshold:

Unhealthy Check Count Threshold:

These settings allow you to control how your load balancer handles session cookies.

Enable Session Stickiness

Cookie Expiration Period (seconds):

Les sections suivantes décrivent les paramètres Elastic Load Balancing que vous pouvez configurer pour votre application.

## Ports

L'équilibrEUR de charge alloué pour gérer les demandes pour votre application Elastic Beanstalk envoie des demandes aux instances Amazon EC2 qui exécutent votre application. L'équilibrEUR de charge alloué peut écouter les demandes sur les ports HTTP et HTTPS, et les acheminer vers les instances Amazon EC2 dans votre application AWS Elastic Beanstalk. Par défaut, l'équilibrEUR de charge gère les demandes sur le port HTTP. Pour que cela fonctionne, au moins un des ports (HTTP ou HTTPS) doit être activé.



### Important

Assurez-vous que le port que vous avez spécifié n'est pas verrouillé ; sinon, vous ne pourrez pas vous connecter à votre application Elastic Beanstalk.

## Contrôle du port HTTP

Pour désactiver le port HTTP, sélectionnez OFF (désactivé) pour HTTP Listener Port (Port d'écoute HTTP). Pour activer le port HTTP, vous sélectionnez un port HTTP (par exemple, 80) dans la liste.

### Note

Pour accéder à votre environnement à l'aide d'un port autre que le port 80, par exemple le port 8080, vous pouvez ajouter un écouteur à l'équilibrEUR de charge existant et configurer le nouvel écouteur de sorte qu'il écoute sur ce port.

Par exemple, en utilisant la [AWS CLI for Classic load balancers](#) (CLI pour les équilibreurs de charge Classic Load Balancer), tapez la commande suivante en remplaçant **LOAD\_BALANCER\_NAME** par le nom de votre équilibreur de charge pour Elastic Beanstalk.

```
aws elb create-load-balancer-listeners --load-balancer-name LOAD_BALANCER_NAME
--listeners "Protocol=HTTP, LoadBalancerPort=8080, InstanceProtocol=HTTP,
InstancePort=80"
```

Par exemple, en utilisant la [AWS CLI for Application Load Balancers](#) (CLI pour les équilibreurs de charge Application Load Balancer), tapez la commande suivante en remplaçant **LOAD\_BALANCER\_ARN** par l'ARN de votre équilibreur de charge pour Elastic Beanstalk.

```
aws elbv2 create-listener --load-balancer-arn LOAD_BALANCER_ARN --protocol HTTP --
port 8080
```

Si vous souhaitez que Elastic Beanstalk surveille votre environnement, ne supprimez pas l'écouteur sur le port 80.

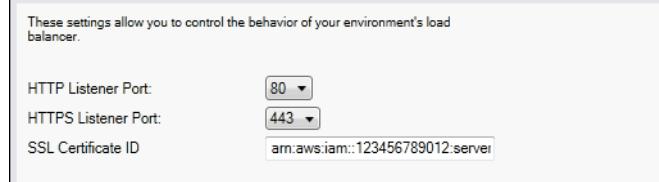
## Contrôle du port HTTPS

Elastic Load Balancing prend en charge le protocole HTTPS/TLS pour activer le chiffrement du trafic pour les connexions client à l'équilibreur de charge. Les connexions à partir de l'équilibreur de charge aux instances EC2 utilisent le chiffrement en clair. Par défaut, le port HTTPS est désactivé.

### Pour activer le port HTTPS

- Créez un nouveau certificat à l'aide d'AWS Certificate Manager (ACM) ou téléchargez un certificat et une clé dans AWS Identity and Access Management (IAM). Pour plus d'informations sur une demande de certificat ACM, consultez [Request a Certificate](#) (Demande de certificat) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur d'AWS Certificate Manager). Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez [Importing Certificates](#) (Importation de certificats) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur d'AWS Certificate Manager). Si ACM n'est pas [disponible dans votre région](#), utilisez AWS Identity and Access Management (IAM) pour télécharger un certificat tiers. Les services ACM et IAM stockeront le certificat et fourniront un Amazon Resource Name (ARN) pour le certificat SSL. Pour de plus amples informations sur la création et le chargement des certificats dans IAM, veuillez consulter [Utilisation des certificats de serveur](#) dans le Guide de l'utilisateur IAM.

- Spécifiez le port HTTPS en sélectionnant un port pour HTTPS Listener Port (Port d'écoute HTTPS).



- Pour SSL Certificate ID (ID du certificat SSL), saisissez l'ARN (Amazon Resources Name) de votre certificat SSL. Par exemple, `arn:aws:iam::123456789012:server-certificate/abc/certs/build` ou `arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`. Utilisez le certificat SSL que vous avez créé ou chargé à l'étape 1.

Pour désactiver le port HTTPS, sélectionnez OFF (désactivé) pour HTTPS Listener Port (Port d'écoute HTTPS).

## Vérifications de l'état

La définition de la vérification de l'état inclut une URL à interroger pour l'intégrité de l'instance. Par défaut, Elastic Beanstalk utilise TCP:80 pour les conteneurs non hérités et HTTP:80 pour les conteneurs hérités. Vous pouvez remplacer l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/default.aspx`) en entrant celle-ci dans la zone URL de vérification de l'état de l'application. Si vous remplacez l'URL par défaut, Elastic Beanstalk utilise HTTP pour interroger la ressource. Pour vérifier si vous utilisez un type de conteneur hérité, consultez [the section called "Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?" \(p. 507\)](#)

Vous pouvez contrôler les paramètres de vérification de l'état via la section Vérification de l'état de l'instance EC2 du panneau Équilibrage de charge.

The screenshot shows a configuration panel for 'Application Health Check'. It includes fields for 'Application Health Check' (set to '/'), 'Health Check Interval (seconds)' (set to 30), 'Health Check Timeout (seconds)' (set to 5), 'Healthy Check Count Threshold' (set to 3), and 'Unhealthy Check Count Threshold' (set to 5). A note at the top states: 'These settings allow you to configure how Elastic Beanstalk determines whether an EC2 instance is healthy or not.'

La définition de la vérification de l'état inclut une URL à interroger pour l'intégrité de l'instance. Remplacez l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/index.jsp`) en entrant celle-ci dans la zone URL de vérification de l'état de l'application.

La liste suivante décrit les paramètres de vérification de l'état que vous pouvez définir pour votre application.

- Pour Intervalle de vérification de l'état (secondes), entrez le nombre de secondes d'attente pour Elastic Load Balancing entre les vérifications de l'état pour les instances Amazon EC2 de votre application.
- Pour Délai de vérification de l'état (secondes), spécifiez le nombre de secondes d'attente d'une réponse pour Elastic Load Balancing avant de considérer que l'instance ne répond pas.
- Pour Seuil du nombre de vérifications de l'état saines et Seuil du nombre de vérifications de l'état non saines, spécifiez le nombre d'analyses d'URL consécutives réussies et non réussies avant qu'Elastic Load Balancing ne modifie l'état de l'instance. Par exemple, si vous spécifiez 5 pour Seuil du nombre de vérifications de l'état non saines, l'URL doit renvoyer un message d'erreur ou une expiration du délai cinq fois de suite avant qu'Elastic Load Balancing considère que la vérification de l'état est un échec.

## Sessions

Par défaut, un équilibrEUR de charge achemine chaque demande de façon indépendante à l'instance de serveur ayant la plus petite charge. Par comparaison, une session permanente lie la session d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la session soient envoyées à la même instance de serveur.

Elastic Beanstalk utilise des cookies HTTP générés par l'équilibrEUR de charge lorsque des sessions permanentes sont activées pour une application. L'équilibrEUR de charge utilise un cookie spécial généré par l'équilibrEUR de charge pour suivre l'instance d'application pour chaque demande. Lorsque l'équilibrEUR de charge reçoit une demande, il vérifie d'abord si ce cookie est présent dans la demande. Si elle est présente, la demande est envoyée à l'instance d'application spécifiée dans le cookie. S'il n'y a pas de cookie, l'équilibrEUR de charge choisit une instance d'application à partir de l'algorithme d'équilibrage de charge existant. Un cookie est inséré dans la réponse pour lier les demandes suivantes provenant du même utilisateur à cette instance d'application. La configuration de la stratégie définit l'expiration d'un cookie, ce qui établit la durée de validité de chaque cookie.

Vous pouvez utiliser la section Sessions de l'onglet ÉquilibrEUR de charge pour spécifier si l'équilibrEUR de charge de votre application autorise la permanence de session.

These settings allow you to control how your load balancer handles session cookies.

Enable Session Stickiness

Cookie Expiration Period (seconds):  (0 - 1000000)

Pour de plus amples informations sur Elastic Load Balancing, veuillez consulter le [Guide du développeur Elastic Load Balancing](#).

## Configuration d'Auto Scaling à l'aide d'AWS Toolkit for Visual Studio

Amazon EC2 Auto Scaling est un service web d'Amazon conçu pour lancer ou résilier automatiquement des instances Amazon EC2 en fonction de déclencheurs définis par l'utilisateur. Vous pouvez configurer des groupes Auto Scaling et y associer des déclencheurs afin de mettre à l'échelle automatiquement les ressources de calcul en fonction de métriques comme l'utilisation de la bande passante ou l'utilisation de l'UC. Amazon EC2 Auto Scaling fonctionne avec Amazon CloudWatch afin de récupérer des métriques pour les instances de serveur exécutant votre application.

Amazon EC2 Auto Scaling vous permet de récupérer un groupe d'instances Amazon EC2 et de définir différents paramètres pour que ce groupe augmente ou diminue automatiquement en nombre. Amazon EC2 Auto Scaling peut ajouter ou supprimer des instances Amazon EC2 de ce groupe pour vous aider à gérer facilement l'évolution du trafic vers votre application.

De plus, Amazon EC2 Auto Scaling surveille l'état de chaque instance Amazon EC2 qu'il lance. Si une instance est résiliée de façon inattendue, Amazon EC2 Auto Scaling détecte cette résiliation et lance une instance de remplacement. Cette fonctionnalité vous permet de maintenir automatiquement un nombre fixe et souhaité d'instances Amazon EC2.

Elastic Beanstalk met en service Amazon EC2 Auto Scaling pour votre application. Vous pouvez modifier la configuration d'instance Amazon EC2 de l'environnement Elastic Beanstalk avec l'onglet Auto Scaling à l'intérieur de l'onglet de l'environnement de votre application dans AWS Toolkit for Visual Studio.

**Events** Auto-scaling automatically launches or terminates EC2 instances based on defined metrics and thresholds called triggers. Auto-scaling will also launch a new EC2 instance in the event of a failure. These settings allow you to control auto-scaling behavior.

**Monitoring**

**Resources**

**AWS X-Ray**

**Server**

**Load Balancer**

**Auto Scaling**

**Notifications**

**Container**

**Advanced**

**Logs**

Minimum Instance Count:  (0 - 10000)

Maximum Instance Count:  (0 - 10000)

Availability Zones:

Scaling Cooldown Time (seconds):  (0 - 10000)

Trigger Measurement:

Trigger Statistic:

Unit of Measurement:

Measurement Period (minutes):  (1 - 600)

Breach Duration (minutes):  (1 - 600)

Upper Threshold:

Upper Breach Scalement Increment:

Lower Threshold:

Lower Breach Scalement Increment:

La section suivante explique comment configurer les paramètres Auto Scaling pour votre application.

### Lancement de la configuration

Vous pouvez modifier la configuration de lancement pour contrôler la façon dont votre application Elastic Beanstalk alloue des ressources Amazon EC2 Auto Scaling.

Les zones Minimum Instance Count (Nombre minimum d'instances) et Maximum Instance Count (Nombre maximum d'instances) vous permettent de spécifier les tailles maximale et minimale du groupe Auto Scaling utilisé par votre application Elastic Beanstalk.

*Auto-scaling automatically launches or terminates EC2 instances based on defined metrics and thresholds called triggers. Auto-scaling will also launch a new EC2 instance in the event of a failure. These settings allow you to control auto-scaling behavior.*

Minimum Instance Count:	<input type="text" value="1"/> (1 - 10000)
Maximum Instance Count:	<input type="text" value="4"/> (1 - 10000)
Availability Zones:	<input type="button" value="Any"/>
Scaling Cooldown Time (seconds):	<input type="text" value="360"/> (0 - 10000)

#### Note

Pour maintenir un nombre fixe d'instances Amazon EC2, attribuez la même valeur aux champs Nombre minimum d'instances et Nombre maximum d'instances.

La zone Zones de disponibilité vous permet de spécifier le nombre de zones de disponibilité dans lesquelles vous souhaitez que se trouvent vos instances Amazon EC2. Il est important de définir ce nombre si vous souhaitez créer des applications à tolérance de panne. Si une zone de disponibilité est défaillante, l'exécution de vos instances se poursuivra dans vos autres zones de disponibilité.

#### Note

Il est actuellement impossible de spécifier la zone de disponibilité dans laquelle se situera votre instance.

### Triggers

Un déclencheur est un mécanisme Amazon EC2 Auto Scaling que vous définissez pour indiquer au système quand vous souhaitez augmenter (monter en puissance) ou réduire (diminuer en puissance) le nombre d'instances. Vous pouvez configurer des déclencheurs pour qu'ils soient activés en fonction de n'importe quelle métrique publiée dans Amazon CloudWatch (l'utilisation de l'UC, par exemple) et pour déterminer si les conditions que vous avez spécifiées sont réunies. Lorsque les seuils inférieurs ou supérieurs des conditions que vous avez spécifiées pour la métrique ont été dépassés pendant la période spécifiée, le déclencheur lance un processus de longue durée que nous appelons une activité de dimensionnement.

Vous pouvez définir un déclencheur de mise à l'échelle de votre application Elastic Beanstalk à l'aide d'AWS Toolkit for Visual Studio.

Trigger Measurement:	NetworkOut
Trigger Statistic:	Average
Unit of Measurement:	Bytes
Measurement Period (minutes):	5 (1 - 600)
Breach Duration (minutes):	5 (1 - 600)
Upper Threshold:	6000000 (0 - 20000000)
Upper Breach Scalement Increment:	1
Lower Threshold:	2000000 (0 - 20000000)
Lower Breach Scalement Increment:	-1

Les déclencheurs Amazon EC2 Auto Scaling fonctionnent en surveillant une métrique Amazon CloudWatch spécifique pour une instance donnée. Les métriques incluent l'utilisation de l'UC, le trafic réseau et l'activité du disque. Utilisez le paramètre Mesure du déclencheur pour sélectionner une métrique associée à votre déclencheur.

La liste suivante décrit les paramètres de déclencheur que vous pouvez configurer à l'aide de la console de gestion AWS.

- Vous pouvez spécifier les statistiques que le déclencheur devrait utiliser. Vous pouvez sélectionner Minimum, Maximum, Sum (Somme) ou Average (Moyenne) pour Statistique du déclencheur.
- Pour Unité de mesure, spécifiez l'unité de mesure du déclencheur.
- La valeur dans la zone Measurement Period (Durée de mesure) spécifie la fréquence à laquelle Amazon CloudWatch mesure les métriques pour votre déclencheur. La valeur Breach duration correspond à la durée pendant laquelle une métrique peut aller au-delà de sa limite définie (telle que spécifiée dans Upper threshold et Lower threshold) avant l'activation du déclencheur.
- Pour Hausse du dimensionnement supérieur en cas de faille et Hausse du dimensionnement inférieur en cas de faille :, spécifiez le nombre d'instances Amazon EC2 à ajouter ou à supprimer lorsque vous effectuez une activité de dimensionnement.

Pour de plus amples informations sur Amazon EC2 Auto Scaling, veuillez consulter Amazon EC2 Auto Scaling dans la [documentation Amazon Elastic Compute Cloud](#).

## Configuration des notifications à l'aide d'AWS Toolkit for Visual Studio

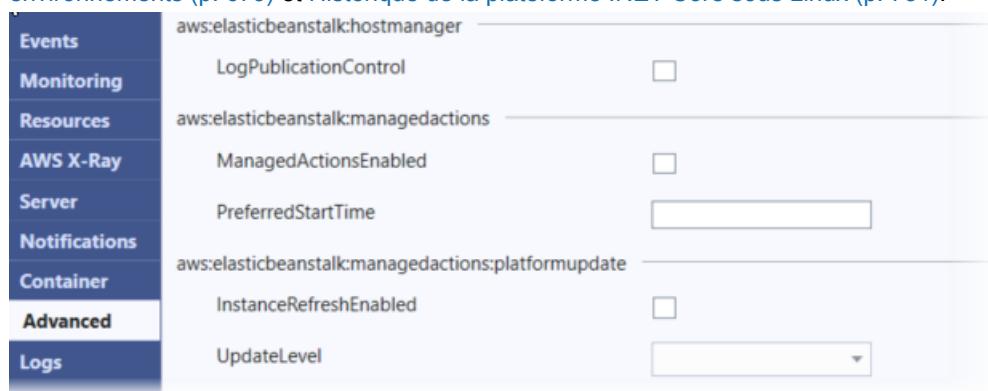
Elastic Beanstalk utilise Amazon Simple Notification Service (Amazon SNS) pour vous informer des événements importants qui concernent votre application. Pour activer les notifications Amazon SNS, saisissez votre adresse e-mail dans la zone Email Address (Adresse e-mail). Pour désactiver ces notifications, supprimez votre adresse e-mail de la zone.

The screenshot shows the AWS Toolkit for Visual Studio configuration interface. On the left, there is a vertical navigation bar with the following items: Events, Monitoring, Resources, AWS X-Ray, Server, Notifications (which is highlighted in blue), Container, Advanced, and Logs. The main area has a title "Events" and a sub-section "Monitoring". Below that, there is a text input field labeled "Email Address:" with the placeholder text "Enter an e-mail address which will be sent notifications regarding important events using the Amazon Simple Notification Service. If you wish to stop receiving notifications, simply remove your e-mail address." To the right of the input field, there is a small "X" button.

## Configuration d'options d'environnement supplémentaires à l'aide d'AWS Toolkit for Visual Studio

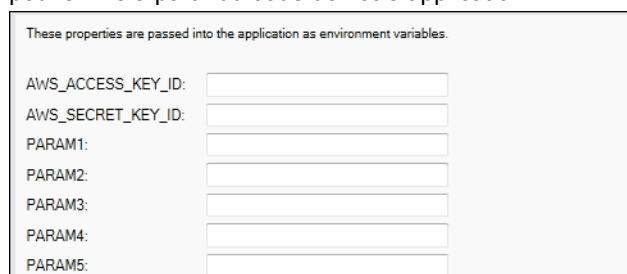
Elastic Beanstalk définit un grand nombre d'options de configuration que vous pouvez utiliser pour configurer le comportement de votre environnement et les ressources qu'il contient. Les options de configuration sont organisées en espaces de noms comme `aws:autoscaling:asg`. Chaque espace de noms définit les options pour le groupe Auto Scaling d'un environnement. Le panneau Advanced (Avancé) répertorie les espaces de noms des options de configuration dans l'ordre alphabétique. Vous pouvez le mettre à jour après la création de l'environnement.

Pour obtenir une liste complète des espaces de noms et des options, y compris les valeurs par défaut et celles prises en charge pour chacun, veuillez consulter [Options générales pour tous les environnements \(p. 679\)](#) et [Historique de la plateforme .NET Core sous Linux \(p. 731\)](#).



## Configuration de conteneurs .NET Core à l'aide d'AWS Toolkit for Visual Studio

Le panneau Container (Conteneur) vous permet de spécifier des variables d'environnement que vous pouvez lire à partir du code de votre application.



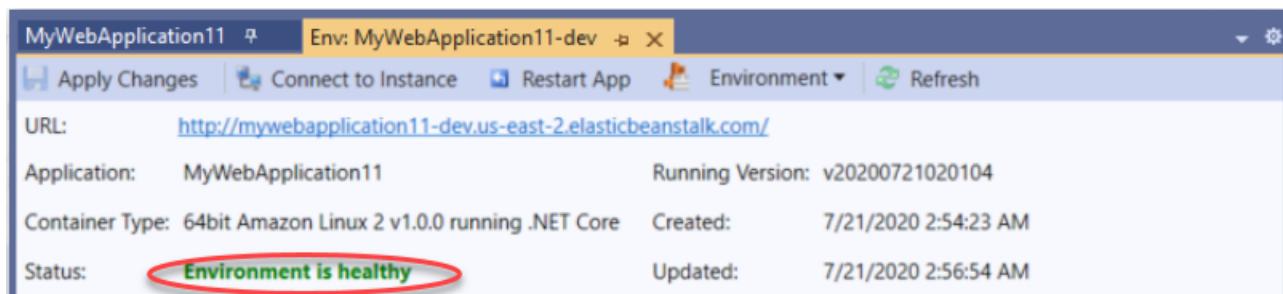
## Surveillance de l'intégrité d'une application

Il est important de savoir que votre site web de production est disponible et répond aux demandes. Elastic Beanstalk fournit des fonctionnalités pour vous aider à surveiller la réactivité de votre application. Il surveille les statistiques relatives à votre application et vous avertit lorsque les seuils sont dépassés.

Pour de plus amples informations sur la surveillance de l'état fournie par Elastic Beanstalk, veuillez consulter [Création de rapports d'intégrité de base \(p. 834\)](#).

Vous pouvez accéder aux informations opérationnelles sur votre application en utilisant soit l'AWS Toolkit for Visual Studio, soit la console de gestion AWS.

La boîte à outils affiche le statut de votre environnement et l'état de votre application dans le champ Status (Statut).



Pour surveiller l'intégrité de l'application

1. Dans AWS Toolkit for Visual Studio, dans AWS Explorer (Explorateur AWS), développez le nœud Elastic Beanstalk, puis le nœud de votre application.
2. Ouvrez le menu contextuel (clic droit) de votre environnement d'application et sélectionnez View Status (Afficher le statut).
3. Dans l'onglet de l'environnement de votre application, sélectionnez Monitoring (Surveillance).

Le panneau Monitoring (Surveillance) comprend un ensemble de graphiques illustrant l'utilisation des ressources pour votre environnement d'application spécifique.



#### Note

Par défaut, la plage de temps est définie sur la dernière heure. Pour modifier ce paramètre, dans la liste Time Range (Plage de temps), sélectionnez une plage de temps différente.

Vous pouvez utiliser AWS Toolkit for Visual Studio ou la console de gestion AWS pour afficher les événements associés à votre application.

Pour afficher des événements d'application

1. Dans AWS Toolkit for Visual Studio, sous AWS Explorer (Explorateur AWS), développez le nœud Elastic Beanstalk et le nœud de votre application.
2. Ouvrez le menu contextuel (clic droit) de votre environnement d'application et sélectionnez View Status (Afficher le statut).
3. Dans l'onglet de l'environnement de votre application, sélectionnez Events (Événements).

Events	Filter: <input type="text"/>			
	Event Time	Event Type	Version Label	Event Details
Monitoring	7/21/2020 2:56:54 AM	INFO		Successfully launched environment: MyWebApplication11
Resources	7/21/2020 2:56:54 AM	INFO		Application available at mywebapplication11-dev.us-east-
AWS X-Ray	7/21/2020 2:56:45 AM	INFO	v20200721020104	Added EC2 instance 'i-00c5680f13fc6f089' to Auto Scaling
Server	7/21/2020 2:56:45 AM	INFO	v20200721020104	Environment health has been set to GREEN
Notifications	7/21/2020 2:56:45 AM	INFO	v20200721020104	Adding instance 'i-00c5680f13fc6f089' to your environme
Container	7/21/2020 2:56:18 AM	INFO		Waiting for EC2 instances to launch. This may take a few r
Advanced	7/21/2020 2:54:58 AM	INFO		Created EIP: 3.14.235.39
Logs	7/21/2020 2:54:42 AM	INFO		Created security group named: awseb-e-ffi5z3mn6m-stac
	7/21/2020 2:54:24 AM	INFO		Using elasticbeanstalk-us-east-2-164656829171 as Amazo
	7/21/2020 2:54:23 AM	INFO		createEnvironment is starting.

## Migration depuis .NET sur les plateformes Windows Server vers la plateforme .NET Core sous Linux

Vous pouvez migrer des applications qui s'exécutent sur les plateformes .NET sous Windows Server vers les plateformes .NET Core sous Linux. Vous trouverez ci-après quelques considérations relatives à la migration depuis Windows vers les plateformes Linux.

### Considérations relatives à la migration vers la plateforme .NET Core sous Linux

Area	Modifications et informations
Configuration de l'application	Sur les plateformes Windows, vous utilisez un <a href="#">manifeste de déploiement (p. 200)</a> pour spécifier les applications qui s'exécutent dans votre environnement. Les plateformes .NET Core sous Linux utilisent un <a href="#">fichier Procfile (p. 164)</a> pour spécifier les applications qui s'exécutent sur les instances de votre environnement. Pour plus de détails sur le regroupement d'applications, veuillez consulter <a href="#">the section called "Regroupement d'applications" (p. 163)</a> .
Serveur proxy	Sur les plateformes Windows, vous utilisez IIS en tant que serveur proxy de votre application. Les plateformes .NET Core sous Linux incluent nginx en tant que proxy inverse par défaut. Vous pouvez choisir de n'utiliser aucun serveur proxy et d'utiliser Kestrel en tant que serveur Web de votre application. Pour en savoir plus, consultez la section <a href="#">the section called "Serveur proxy" (p. 165)</a> .
Routage	Sur les plateformes Windows, vous utilisez IIS dans votre code d'application et incluez un <a href="#">manifeste de déploiement (p. 200)</a> pour configurer le chemin d'accès IIS. Pour la plateforme .NET Core sous Linux, vous utilisez le <a href="#">routage ASP .NET Core</a> dans votre code d'application et mettez à jour la configuration nginx de votre environnement. Pour en savoir plus, consultez la section <a href="#">the section called "Serveur proxy" (p. 165)</a> .
Journaux	Les plateformes Linux et Windows diffusent différents journaux. Pour plus d'informations, consultez <a href="#">the section called "Méthode de configuration de CloudWatch Logs par Elastic Beanstalk" (p. 898)</a> .

# Création et déploiement d'applications .NET sur Elastic Beanstalk

AWS Elastic Beanstalk pour .NET permet de déployer, de gérer et de dimensionner plus facilement vos applications web ASP.NET qui utilisent Amazon Web Services. Elastic Beanstalk pour .NET est accessible à toute personne qui développe ou héberge une application web utilisant IIS.

Pour commencer : pour utiliser notre tutoriel, accédez directement à la page [Tutoriel : Apprenez à déployer un exemple d'application .NET à l'aide d'Elastic Beanstalk \(p. 206\)](#). Dans ce didacticiel, vous allez déployer un exemple d'application web ASP.NET dans un conteneur d'application AWS Elastic Beanstalk.

Le reste de cette section présente les instructions à suivre pour créer, tester, déployer et redéployer votre application web ASP.NET dans Elastic Beanstalk. Certains exemples illustrent l'utilisation d'AWS Toolkit for Visual Studio et la sous-section [the section called ". AWS Toolkit for Visual Studio" \(p. 224\)](#) explique comment gérer et configurer vos applications et environnements à l'aide de la boîte à outils. Pour de plus amples informations sur les conditions préalables, les instructions d'installation et l'exécution d'exemples de code, accédez à la [boîte à outils AWS for Visual Studio](#). Ce site fournit également des informations utiles sur les outils, des didacticiels et des ressources supplémentaires pour les développeurs ASP.NET.

## Note

Cette plateforme ne prend pas en charge les fonctionnalités Elastic Beanstalk suivantes :

- Environnements de travail. Pour plus d'informations, consultez [Environnements de travail Elastic Beanstalk \(p. 521\)](#).
- Journaux de groupes. Pour plus d'informations, consultez [Affichage des journaux d'instance \(p. 884\)](#).

En outre, les versions de plateforme antérieures à v2.0.0 ne prennent pas en charge les rapports d'intégrité améliorés, les mises à jour de plateformes gérées, les mises à jour immuables, les déploiements immuables et les déploiements par propagation avec un lot supplémentaire.

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

## Rubriques

- [Démarrer avec .NET sur Elastic Beanstalk \(p. 192\)](#)
- [Configuration de votre environnement de développement .NET \(p. 195\)](#)
- [Utilisation de la plateforme .NET Elastic Beanstalk \(p. 196\)](#)
- [Tutoriel : Apprenez à déployer un exemple d'application .NET à l'aide d'Elastic Beanstalk \(p. 206\)](#)
- [Tutoriel : Déploiement d'une application ASP.NET Core avec Elastic Beanstalk \(p. 212\)](#)
- [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application .NET \(p. 221\)](#)
- [. AWS Toolkit for Visual Studio \(p. 224\)](#)
- [Migration de votre application .NET sur site vers Elastic Beanstalk \(p. 250\)](#)
- [Resources \(p. 251\)](#)

## Démarrer avec .NET sur Elastic Beanstalk

Pour commencer à utiliser des applications .NET sur AWS Elastic Beanstalk, vous avez simplement besoin de télécharger la [solution groupée de fichiers source \(p. 415\)](#) d'une application en tant que première

version de l'application et de la déployer dans un environnement. Lorsque vous créez un environnement, Elastic Beanstalk alloue toutes les ressources AWS nécessaires pour exécuter une application web hautement évolutive.

## Lancement d'un environnement avec un exemple d'application .NET

Elastic Beanstalk fournit des exemples d'applications à page unique pour chaque plateforme et des exemples plus complexes qui illustrent l'utilisation de ressources AWS supplémentaires. Celles-ci incluent Amazon RDS et les fonctions et API propres au langage ou à la plateforme.

### Samples

Nom	Configurations prises en charge	Type d'environnement	Source	Description
.NET WS 2019 R2		Serveur web	<a href="#">dotnet-asp-v1.zip</a>	
Default	WS 2019 R2 Server Core			Application web ASP.NET avec une page unique configurée pour s'afficher à la racine du site web.
	WS 2016 R2			
	WS 2016 R2 Server Core			
	WS 2012 R2			
	WS 2012 R2 Server Core			
	WS 2012			
ASP.NET 2012 R2		Serveur web	<a href="#">dotnet-aspmvc5-v1.zip</a>	L'application web ASP.NET avec une architecture modèle-vue-contrôle classique.
MVC5				

Téléchargez l'un des exemples d'application et déployez-le dans Elastic Beanstalk en procédant comme suit :

Pour lancer un environnement avec un exemple d'application (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le volet de navigation, choisissez Applications, puis le nom d'une application existante dans la liste ou [créez-en un \(p. 406\)](#).
3. Sur la page de présentation de l'application, choisissez Create a new environment (Créer un nouvel environnement).

Environment name	Health	Date created	Last modified	URL	Platform
GettingStartedApp-env	<span>OK</span>	2020-01-28 12:06:50 UTC-0800	2020-01-30 15:02:35 UTC-0800	GettingStartedApp-env.bn7dx222kw.us-east-2.elasticbeanstalk.com	Tomcat running Linux
GettingStartedApp-Worker	<span>OK</span>	2020-01-28 16:34:29 UTC-0800	2020-01-28 16:38:20 UTC-0800	GettingStartedApp-Worker.bn7dx222kw.us-east-2.elasticbeanstalk.com	IIS 10.0 Windows

4. Ensuite, pour le niveau d'environnement, choisissez l'environnement de serveur web ou le [niveau d'environnement de travail \(p. 14\)](#). Vous ne pouvez pas modifier le niveau d'un environnement après sa création.

Note

La plateforme [.NET sur Windows Server \(p. 192\)](#) ne prend pas en charge le niveau d'environnement worker.

**Select environment tier**

AWS Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web server environments run a website, web application, or web API that serves HTTP requests. Worker environments run a background processing task that listens for messages on an Amazon SQS queue. Worker applications post those messages to your application by using HTTP.

**Web server environment**  
Run a website, web application, or web API that serves HTTP requests.  
[Learn more](#)

**Worker environment**  
Run a worker application that processes long-running workloads on demand or performs tasks on a schedule.  
[Learn more](#)

5. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

Note

Elastic Beanstalk prend en charge plusieurs [versions \(p. 31\)](#) pour la plupart des plateformes répertoriées. Par défaut, la console sélectionne la version recommandée pour la plateforme et la branche de plateforme que vous choisissez. Si votre application nécessite

une version différente, vous pouvez la sélectionner ici, ou choisir Configurer plus d'options, selon les instructions de l'étape 7. Pour plus d'informations sur les versions de plateforme prises en charge, consultez [the section called “Plateformes prises en charge” \(p. 31\)](#).

6. Pour l'option Code de l'application, choisissez Exemple d'application.
7. Pour personnaliser davantage votre environnement, choisissez Configurer plus d'options. Les options suivantes peuvent être définies uniquement lors de la création de l'environnement :
  - Nom de l'environnement
  - Nom de domaine
  - Version de plateforme
  - VPC
  - Palier

Vous pouvez modifier les paramètres suivants après la création de l'environnement, mais ils requièrent la mise en œuvre de nouvelles instances ou d'autres ressources et leur application peut prendre du temps :

- Type d'instance, volume racine, paire de clés et rôle AWS Identity and Access Management (IAM)
- Base de données interne Amazon RDS
- Equilibreur de charge

Pour de plus amples informations sur tous les paramètres disponibles, veuillez consulter [Assistant de création d'un environnement \(p. 442\)](#).

8. Choisissez Create environment.

## Étapes suivantes

Une fois que vous disposez d'un environnement exécutant une application, vous pouvez [déployer une nouvelle version \(p. 476\)](#) de l'application ou une application totalement différente à tout moment. Le déploiement d'une nouvelle version d'application est rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2.

Une fois que vous avez déployé un ou deux exemples d'application et que vous êtes prêt à développer des applications localement, vous pouvez suivre les instructions de la [section suivante \(p. 195\)](#) afin de configurer un environnement de développement .NET.

## Configuration de votre environnement de développement .NET

Configurez un environnement de développement .NET pour tester votre application localement avant de la déployer dans AWS Elastic Beanstalk. Cette rubrique décrit les étapes de configuration de l'environnement de développement et des liens vers les pages d'installation pour des outils utiles.

Pour accéder aux outils et aux étapes de configuration courants qui s'appliquent à toutes les langues, veuillez consulter [Configuration de votre machine de développement pour une utilisation avec Elastic Beanstalk \(p. 1024\)](#).

### Sections

- [Installation d'un IDE \(p. 161\)](#)
- [Installation du AWS Toolkit for Visual Studio \(p. 161\)](#)

Si vous avez besoin gérer les ressources AWS à partir de votre application, installez AWS SDK pour .NET. Par exemple, vous pouvez utiliser Amazon S3 pour stocker et récupérer des données.

Avec AWS SDK pour .NET, vous pouvez démarrer en quelques minutes avec un seul package téléchargeable avec des modèles de projet Visual Studio, la bibliothèque AWS .NET, des exemples de code C# et de la documentation. Des exemples pratiques sont fournis dans C# sur la façon d'utiliser les bibliothèques pour créer des applications. Nous vous proposons des didacticiels vidéo en ligne et des documents de référence pour vous aider à utiliser les bibliothèques et les exemples de code.

Consultez la [page d'accueil AWS SDK pour .NET](#) pour plus d'informations et des instructions d'installation.

## Installation d'un IDE

Les environnements de développement intégré (IDE) offrent un large éventail de fonctionnalités qui facilitent le développement d'applications. Si vous n'avez pas utilisé un IDE pour le développement .NET, essayez Visual Studio Community pour démarrer.

Visitez la page [Visual Studio Community](#) pour télécharger et installer Visual Studio Community.

## Installation du AWS Toolkit for Visual Studio

[AWS Toolkit for Visual Studio \(p. 224\)](#) est un plug-in open source pour l'IDE Visual Studio. Il permet aux développeurs de développer, de déboguer et de déployer plus facilement des applications .NET utilisant AWS. Consultez la [page d'accueil Toolkit for Visual Studio](#) pour obtenir des instructions d'installation.

## Utilisation de la plateforme .NET Elastic Beanstalk

AWS Elastic Beanstalk prend en charge un certain nombre de plateformes pour différentes versions de l'infrastructure de programmation .NET et de Windows Server. Consultez [.NET sur Windows Server avec IIS](#) dans le document Plateformes AWS Elastic Beanstalk pour obtenir une liste complète.

Elastic Beanstalk fournit des [options de configuration \(p. 658\)](#) que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute sur des instances EC2 dans votre environnement Elastic Beanstalk. Vous pouvez configurer des variables d'environnement nécessaires pour votre application, activer la rotation des journaux sur Amazon S3 et définir les paramètres de .NET framework.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour [modifier la configuration d'un environnement en cours d'exécution \(p. 671\)](#). Pour éviter de perdre la configuration de votre environnement en le résilient, vous pouvez utiliser des [configurations enregistrées \(p. 779\)](#) pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des [fichiers de configuration \(p. 737\)](#). Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour de plus amples informations sur la priorité et les autres méthodes de modification des paramètres, veuillez consulter [Options de configuration \(p. 658\)](#).

## Configuration de votre environnement .NET dans la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour activer la rotation des journaux sur Amazon S3, configurer des variables que votre application peut lire depuis l'environnement et modifier les paramètres de .NET Framework.

Pour configurer votre environnement .NET dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

### Options du conteneur

- Cibler l'exécution .NET – Réglez sur 2.0 pour exécuter CLR v2.
- Activer les applications 32 bits – Réglez sur True pour exécuter des applications 32 bits.

### Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

### Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Ces paramètres sont passés en tant que paires clé-valeur à l'application. Utilisez `System.EnvironmentVariable` pour les lire. Des clés identiques peuvent exister dans `web.config` et en tant que propriétés de l'environnement. Utilisez l'espace de noms `System.Configuration` pour lire les valeurs de `web.config`.

```
NameValueCollection appConfig = ConfigurationManager.AppSettings;  
string endpoint = appConfig["API_ENDPOINT"];
```

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

### Espace de noms aws:elasticbeanstalk:container:dotnet:apppool

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de

configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

La plateforme .NET définit les options dans l'espace de noms

`aws:elasticbeanstalk:container:dotnet:apppool` que vous pouvez utiliser pour configurer l'environnement d'exécution .NET.

L'exemple de fichier de configuration suivant affiche des paramètres pour chacune des options disponibles dans cet espace de noms :

Example `.ebextensions/dotnet-settings.config`

```
option_settings:  
  aws:elasticbeanstalk:container:dotnet:apppool:  
    Target Runtime: 2.0  
    Enable 32-bit Applications: True
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Migration entre les principales versions de la plateforme Windows Server pour Elastic Beanstalk

AWS Elastic Beanstalk a eu plusieurs versions majeures de sa plateforme Windows Server. Cette page couvre les améliorations principales de chaque version majeure et les éléments que vous devez prendre en compte avant de migrer vers une version ultérieure.

La plateforme Windows Server est actuellement à la version 2 (v2). Si votre application utilise une version de plateforme Windows Server antérieure à v2, nous vous recommandons de migrer vers v2.

### Nouveautés des versions majeures de la plateforme Windows Server

#### Plateforme Windows Server V2

La version 2 (v2) de la plateforme Windows Server pour Elastic Beanstalk a été [publiée en février 2019](#). La V2 aligne le comportement de la plateforme Windows Server sur celui des plateformes Linux Elastic Beanstalk pour plusieurs aspects importants. Le V2 est entièrement rétrocompatible avec la V1, ce qui facilite la migration à partir de la V1.

La plateforme Windows Server prend désormais en charge les fonctionnalités suivantes :

- Gestion des versions : chaque version reçoit un nouveau numéro et vous pouvez consulter les versions précédentes (qui sont toujours disponibles) lors de la création et de la gestion des environnements.
- État amélioré – Pour plus de détails, veuillez consulter [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#).
- Déploiements immuables et propagés avec un lot supplémentaire – Pour de plus amples informations sur les stratégies de déploiement, veuillez consulter [Déploiement d'applications dans des environnements Elastic Beanstalk \(p. 476\)](#).
- Mises à jour immuables – Pour de plus amples informations sur les types de mises à jour, veuillez consulter [Configuration changes \(p. 488\)](#).
- Mises à jour gérées de la plateforme – Pour de plus amples informations, veuillez consulter [Mises à jour gérées de la plateforme \(p. 501\)](#).

## Note

Les nouvelles fonctionnalités de déploiement et de mise à jour dépendent des rapports améliorés sur l'état de santé. Activez la fonctionnalité correspondante pour les utiliser. Pour plus d'informations, consultez [Activation des rapports améliorés sur l'état Elastic Beanstalk \(p. 845\)](#).

### Plateforme Windows Server V1

La version 1.0.0 (v1) de la plateforme Windows Server pour Elastic Beanstalk a été publiée en octobre 2015. Cette version modifie l'ordre dans lequel Elastic Beanstalk traite les commandes dans les [fichiers de configuration \(p. 737\)](#) lors de la création et des mises à jour d'un environnement.

Les versions antérieures de la plateforme ne comportent pas de numéro dans le nom de pile de solutions :

- Windows Server 64 bits 2012 R2 exécutant IIS 8.5
- Windows Server Core 64 bits 2012 R2 exécutant IIS 8.5
- Windows Server 64 bits 2012 exécutant IIS 8
- Windows Server 64 bits 2008 R2 exécutant IIS 7.5

Dans les versions antérieures, l'ordre de traitement des fichiers de configuration n'est pas cohérent. Pendant la création de l'environnement, les commandes `Container Commands` sont exécutées une fois que la source de l'application est déployée dans IIS. Lors d'un déploiement dans un environnement en cours d'exécution, les commandes de conteneur sont exécutées avant le déploiement de la nouvelle version. Lors d'un ajustement à la hausse, les fichiers de configuration ne sont pas traités du tout.

En outre, IIS démarre avant l'exécution des commandes de conteneur. Ce comportement a conduit certains clients à mettre en place des solutions de contournement dans les commandes de conteneur. Elles consistent à mettre en veille le serveur IIS avant l'exécution des commandes et à le redémarrer une fois l'exécution terminée.

La version 1 corrige le problème d'incohérence et aligne le comportement de la plateforme Windows Server sur celui des plateformes Linux Elastic Beanstalk. Sur la plateforme v1, Elastic Beanstalk exécute toujours les commandes de conteneur avant de démarrer le serveur IIS.

Les piles de solutions de la plateforme v1 comportent la mention `v1` après la version de Windows Server :

- Windows Server 64 bits 2012 R2 v1.1.0 exécutant IIS 8.5
- Windows Server Core 64 bits 2012 R2 v1.1.0 exécutant IIS 8.5
- Windows Server 64 bits 2012 v1.1.0 exécutant IIS 8
- Windows Server 64 bits 2008 R2 v1.1.0 exécutant IIS 7.5

En outre, la plateforme v1 extrait le contenu du bundle de fichiers source de votre application dans `C:\staging\` avant d'exécuter les commandes de conteneur. Une fois que l'exécution des commandes de conteneur est terminée, le contenu de ce dossier est compressé en fichier `.zip` et déployé dans IIS. Ce processus vous permet de modifier le contenu du bundle source de votre application à l'aide des commandes ou d'un script avant le déploiement.

### Migration à partir de versions majeures antérieures de la plateforme Windows Server

Consultez cette section pour prendre connaissance des considérations relatives à la migration avant de mettre à jour votre environnement. Pour mettre à jour votre plateforme d'environnement vers une version plus récente, consultez [Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk \(p. 496\)](#).

## De V1 vers V2

La plateforme Windows Server v2 ne prend pas en charge .NET Core 1.x et 2.0. Si vous migrez votre application à partir de Windows Server v1 vers v2 et que votre application utilise l'une de ces versions .NET Core, mettez à jour votre application vers une version .NET Core prise en charge par la v2. Pour obtenir la liste des versions prises en charge, consultez [.NET sur Windows Server avec IIS](#) dans les Plateformes AWS Elastic Beanstalk.

Si votre application utilise une Amazon Machine Image (AMI) personnalisée, créez une AMI personnalisée basée sur une AMI de plate-forme Windows Server v2. Pour en savoir plus, consultez la section [Utilisation d'une Amazon Machine Image \(AMI\) personnalisée \(p. 787\)](#).

### Note

Les fonctionnalités de déploiement et de mise à jour qui sont nouvelles dans Windows Server v2 dépendent des rapports améliorés sur l'état de santé. Lorsque vous migrez un environnement vers v2, les rapports améliorés sur l'état de santé sont désactivés. Activez-les pour utiliser ces fonctionnalités. Pour plus d'informations, consultez [Activation des rapports améliorés sur l'état Elastic Beanstalk \(p. 845\)](#).

## À partir de versions antérieures à la V1

Outre les considérations concernant la migration à partir de la v1, si vous migrez votre application à partir d'une pile de solutions Windows Server antérieure à la v1 et que vous utilisez actuellement des commandes de conteneur, supprimez toutes les commandes que vous avez ajoutées pour contourner les incohérences de traitement lors de la migration vers une version plus récente. Depuis la v1, l'exécution complète des commandes de conteneur est garantie avant l'application source qui est déployée et avant le démarrage d'IIS. Vous pouvez ainsi apporter des modifications à la source dans c:\staging et modifier sans problème les fichiers de configuration IIS au cours de cette étape.

Par exemple, vous pouvez utiliser la AWS CLI pour télécharger un fichier DLL dans la source de votre application à partir d'Amazon S3 :

```
.ebextensions\copy-dll.config

container_commands:
  copy-dll:
    command: aws s3 cp s3://DOC-EXAMPLE-BUCKET/dlls/large-dll.dll .\lib\
```

Pour de plus amples informations sur l'utilisation des fichiers de configuration, veuillez consulter [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

## Exécution de plusieurs applications et d'applications ASP.NET Core avec un manifeste de déploiement

Vous pouvez utiliser un manifeste de déploiement pour indiquer à Elastic Beanstalk comment déployer votre application. Avec cette méthode, vous n'avez pas besoin d'utiliser MSDeploy pour générer une solution groupée source pour une seule application ASP.NET qui s'exécute sur le chemin d'accès racine de votre site Web. À la place, vous pouvez utiliser un fichier manifeste afin d'exécuter plusieurs applications sur différents chemins. Vous pouvez également demander à Elastic Beanstalk de déployer et d'exécuter l'application avec ASP.NET Core. Vous pouvez également utiliser un manifeste de déploiement pour configurer un groupe d'applications dans lequel exécuter vos applications.

Les manifestes de déploiement ajoutent la prise en charge des [applications .NET Core \(p. 201\)](#) à Elastic Beanstalk. Vous pouvez déployer une application .NET Framework sans manifeste de déploiement. Toutefois, les applications .NET Core nécessitent un manifeste de déploiement pour s'exécuter sur Elastic Beanstalk. Lors de l'utilisation d'un manifeste de déploiement, vous créez une archive de site pour chaque

application, puis regroupez les archives de site dans un second fichier ZIP contenant le manifeste de déploiement.

Les manifestes de déploiement ajoutent également la possibilité d'[exécuter plusieurs applications sur des chemins d'accès différents \(p. 202\)](#). Un manifeste de déploiement définit un ensemble de cibles de déploiement, dotées chacune d'une archive de site et d'un chemin d'accès sur lequel elle doit être exécutée par IIS. Par exemple, vous pouvez exécuter une API web sur le chemin d'accès /api pour répondre aux demandes asynchrones, et une application web sur le chemin d'accès racine qui utilise l'API.

Vous pouvez également utiliser un manifeste de déploiement pour [exécuter plusieurs applications à l'aide de groupes d'applications dans IIS ou Kestrel \(p. 203\)](#). Vous pouvez configurer un pool d'applications pour redémarrer périodiquement vos applications, exécuter des applications 32 bits ou utiliser une version spécifique du runtime .NET Framework.

Pour effectuer une personnalisation complète, vous pouvez [écrire vos propres scripts de déploiement \(p. 205\)](#) dans Windows PowerShell et indiquer à Elastic Beanstalk les scripts à exécuter pour installer, désinstaller et redémarrer votre application.

Les manifestes de déploiement et les fonctionnalités associées requièrent une plateforme Windows Server [version 1.2.0 ou ultérieure \(p. 198\)](#).

#### Sections

- [Applications .NET core \(p. 201\)](#)
- [Exécution de plusieurs applications \(p. 202\)](#)
- [Configuration de groupes d'applications \(p. 203\)](#)
- [Définition de déploiements personnalisés \(p. 205\)](#)

## Applications .NET core

Vous pouvez utiliser un manifeste de déploiement pour exécuter des applications .NET Core sur Elastic Beanstalk. .NET Core est une version multiplateforme de .NET qui inclut un outil de ligne de commande (dotnet). Vous pouvez l'utiliser pour générer une application, l'exécuter localement et la préparer en vue de sa publication.

#### Note

Consultez [Tutorial : Déploiement d'une application ASP.NET Core avec Elastic Beanstalk \(p. 212\)](#) pour accéder à un tutoriel et un exemple d'application qui utilisent un manifeste de déploiement pour exécuter une application .NET Core sur Elastic Beanstalk.

Pour exécuter une application .NET Core sur Elastic Beanstalk, vous pouvez exécuter `dotnet publish` et placer la sortie dans une archive ZIP, en excluant les répertoires contenant des fichiers. Placez l'archive de site dans un bundle de fichiers source avec un manifeste de déploiement, avec une cible de déploiement de type `aspNetCoreWeb`.

Le manifeste de déploiement suivant exécute une application .NET Core à partir d'une archive de site nommée `dotnet-core-app.zip` sur le chemin d'accès racine.

Example aws-windows-deployment-manifest.json - .NET core

```
{  
  "manifestVersion": 1,  
  "deployments": {  
    "aspNetCoreWeb": [  
      {  
        "name": "my-dotnet-core-app",  
        "parameters": {  
          "archive": "dotnet-core-app.zip",  
        }  
      }  
    ]  
  }  
}
```

```
        "iisPath": "/"  
    }  
}  
]  
}
```

Regroupez le manifeste et l'archive de site dans un fichier ZIP pour créer un bundle de fichiers source.

Example dotnet-core-bundle.zip

```
.  
|-- aws-windows-deployment-manifest.json  
`-- dotnet-core-app.zip
```

L'archive de site contient le code d'application compilé, les dépendances et le fichier web.config.

Example dotnet-core-app.zip

```
.  
|-- Microsoft.AspNetCore.Hosting.Abstractions.dll  
|-- Microsoft.AspNetCore.Hosting.Server.Abstractions.dll  
|-- Microsoft.AspNetCore.Hosting.dll  
|-- Microsoft.AspNetCore.Http.Abstractions.dll  
|-- Microsoft.AspNetCore.Http.Extensions.dll  
|-- Microsoft.AspNetCore.Http.Features.dll  
|-- Microsoft.AspNetCore.Http.dll  
|-- Microsoft.AspNetCore.HttpOverrides.dll  
|-- Microsoft.AspNetCore.Server.IISIntegration.dll  
|-- Microsoft.AspNetCore.Server.Kestrel.dll  
|-- Microsoft.AspNetCore.WebUtilities.dll  
|-- Microsoft.Extensions.Configuration.Abstractions.dll  
|-- Microsoft.Extensions.Configuration.EnvironmentVariables.dll  
|-- Microsoft.Extensions.Configuration.dll  
|-- Microsoft.Extensions.DependencyInjection.Abstractions.dll  
|-- Microsoft.Extensions.DependencyInjection.dll  
|-- Microsoft.Extensions.FileProviders.Abstractions.dll  
|-- Microsoft.Extensions.FileProviders.Physical.dll  
|-- Microsoft.Extensions.FileSystemGlobbing.dll  
|-- Microsoft.Extensions.Logging.Abstractions.dll  
|-- Microsoft.Extensions.Logging.dll  
|-- Microsoft.Extensions.ObjectPool.dll  
|-- Microsoft.Extensions.Options.dll  
|-- Microsoft.Extensions.PlatformAbstractions.dll  
|-- Microsoft.Extensions.Primitives.dll  
|-- Microsoft.Net.Http.Headers.dll  
|-- System.Diagnostics.Contracts.dll  
|-- System.Net.WebSockets.dll  
|-- System.Text.Encodings.Web.dll  
|-- dotnet-core-app.deps.json  
|-- dotnet-core-app.dll  
|-- dotnet-core-app.pdb  
|-- dotnet-core-app.runtimeconfig.json  
`-- web.config
```

Consultez [le didacticiel \(p. 212\)](#) pour obtenir un exemple complet.

## Exécution de plusieurs applications

Vous pouvez exécuter plusieurs applications à l'aide d'un manifeste de déploiement en définissant plusieurs cibles de déploiement.

Le manifeste de déploiement suivant configure deux applications .NET Core. L'application `WebAPITest` implémente quelques API Web et sert des demandes asynchrones au chemin `/api`. L'application `ASPNetTest` est une application Web qui sert les demandes à la racine du chemin d'accès.

Example `aws-windows-deployment-manifest.json` - multiple apps

```
{  
  "manifestVersion": 1,  
  "deployments": [  
    "aspNetCoreWeb": [  
      {  
        "name": "WebAPITest",  
        "parameters": {  
          "AppBundle": "webapi.zip",  
          "iisPath": "/api"  
        }  
      },  
      {  
        "name": "ASPNetTest",  
        "parameters": {  
          "AppBundle": "aspnet.zip",  
          "iisPath": "/"  
        }  
      }  
    ]  
  }  
}
```

Un exemple d'application avec plusieurs applications est disponible ici :

- bundle source à déployer - [dotnet-multiapp-sample-bundle-v2.zip](#)
- Code source - [dotnet-multiapp-sample-source-v2.zip](#)

## Configuration de groupes d'applications

Vous pouvez prendre en charge plusieurs applications dans votre environnement Windows. Vous avez le choix entre deux approches :

- Vous pouvez utiliser le modèle d'hébergement hors processus avec le serveur Web Kestrel. Avec ce modèle, vous configurez plusieurs applications pour qu'elles s'exécutent dans un groupe d'applications.
- Vous pouvez utiliser le modèle d'hébergement en cours, avec lequel vous utilisez plusieurs groupes d'applications pour exécuter plusieurs applications avec une seule application dans chaque groupe. Si vous utilisez le serveur IIS et que vous avez besoin d'exécuter plusieurs applications, vous devez utiliser cette approche.

Pour configurer Kestrel de façon à exécuter plusieurs applications dans un groupe d'applications, ajoutez `hostingModel="OutOfProcess"` dans le fichier `web.config`. Considérez les exemples suivants.

Example `web.config` - pour le modèle d'hébergement hors processus Kestrel

```
<configuration>  
<location path=". " inheritInChildApplications="false">  
<system.webServer>  
<handlers>  
<add  
  name="aspNetCore"  
  path="*" verb="*"  
  modules="AspNetCoreModuleV2"
```

```
    resourceType="Unspecified" />
</handlers>
<aspNetCore
    processPath="dotnet"
    arguments=".\\CoreWebApp-5-0.dll"
    stdoutLogEnabled="false"
    stdoutLogFile=".\\logs\\stdout"
    hostingModel="OutOfProcess" />
</system.webServer>
</location>
</configuration>
```

#### Example aws-windows-deployment-manifest.json - plusieurs applications

```
{
"manifestVersion": 1,
"deployments": {"msDeploy": [
    {"name": "Web-app1",
        "parameters": {"archive": "site1.zip",
                      "iisPath": "/"}
    },
    {"name": "Web-app2",
        "parameters": {"archive": "site2.zip",
                      "iisPath": "/app2"}
    }
]
}
```

IIS ne prend pas en charge plusieurs applications dans un groupe d'applications car il utilise le modèle d'hébergement en cours. Par conséquent, vous devez configurer plusieurs applications en affectant chaque application à un groupe d'applications. En d'autres termes, n'affectez qu'une seule application à un groupe d'applications.

Vous pouvez configurer IIS pour qu'il utilise différents groupes d'applications dans le fichier `aws-windows-deployment-manifest.json`. Effectuez les mises à jour suivantes lorsque vous vous référez au fichier exemple suivant :

- Ajoutez une section `iisConfig` qui comprend une sous-section appelée `appPools`.
- Dans le bloc `appPools`, répertoriez les groupes d'applications.
- Dans la section `deployments`, définissez une section `parameters` pour chaque application.
- Pour chaque application, la section `parameters` spécifie une archive, un chemin d'accès pour l'exécuter et un `appPool` dans lequel s'exécute.

Le manifeste de déploiement suivant configure deux groupes d'applications qui redémarrent leur application toutes les 10 minutes. En outre, ils attachent leurs applications à une application Web .NET Framework qui s'exécute au chemin spécifié.

#### Example aws-windows-deployment-manifest.json - une application par groupe d'applications

```
{
"manifestVersion": 1,
"iisConfig": {"appPools": [
    {"name": "MyFirstPool",
        "recycling": {"regularTimeInterval": 10}
    }
],
```

```
{ "name": "MySecondPool",
  "recycling": {"regularTimeInterval": 10}
}
],
},
"deployments": {"msDeploy": [
  {"name": "Web-app1",
    "parameters": {
      "archive": "site1.zip",
      "iisPath": "/",
      "appPool": "MyFirstPool"
    }
  },
  {"name": "Web-app2",
    "parameters": {
      "archive": "site2.zip",
      "iisPath": "/app2",
      "appPool": "MySecondPool"
    }
  }
]
}
```

## Définition de déploiements personnalisés

Pour encore davantage de contrôle, vous pouvez personnaliser entièrement un déploiement d'applications en définissant un déploiement personnalisé.

Le manifeste de déploiement suivant indique à Elastic Beanstalk d'exécuter un script `install` nommé `siteInstall.ps1`. Ce script installe le site Web pendant le lancement et les déploiements d'instance. En outre, le manifeste de déploiement indique également à Elastic Beanstalk d'exécuter un script `uninstall` avant d'installer une nouvelle version pendant un déploiement et un script `restart` pour redémarrer l'application lorsque vous sélectionnez [Restart App Server \(p. 432\)](#) (Redémarrer le server d'application) dans la console de gestion AWS.

Example aws-windows-deployment-manifest.json - custom deployment

```
{
  "manifestVersion": 1,
  "deployments": {
    "custom": [
      {
        "name": "Custom site",
        "scripts": {
          "install": {
            "file": "siteInstall.ps1"
          },
          "restart": {
            "file": "siteRestart.ps1"
          },
          "uninstall": {
            "file": "siteUninstall.ps1"
          }
        }
      }
    ]
  }
}
```

Incluez les artefacts requis pour exécuter l'application dans le bundle de fichiers source avec le manifeste et les scripts.

### Example Custom-site-bundle.zip

```
.  
|--- aws-windows-deployment-manifest.json  
|--- siteInstall.ps1  
|--- siteRestart.ps1  
|--- siteUninstall.ps1  
`--- site-contents.zip
```

## Tutoriel : Apprenez à déployer un exemple d'application .NET à l'aide d'Elastic Beanstalk

Dans ce didacticiel, vous allez apprendre à déployer un exemple d'application .NET sur AWS Elastic Beanstalk à l'aide du AWS Toolkit for Visual Studio.

#### Note

Ce didacticiel utilise un exemple d'application web ASP.NET que vous pouvez télécharger [ici](#). Il utilise également [Toolkit for Visual Studio](#) et a été testé à l'aide de Visual Studio Professional 2012.

## Création de l'environnement

Tout d'abord, utilisez l'assistant Create New Application (Créer une nouvelle application) dans la console Elastic Beanstalk pour créer l'environnement de l'application. Pour Platform (Plateforme), choisissez .NET.

#### Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](http://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créer une application.

Lorsque l'environnement est opérationnel, ajoutez une instance de base de données Amazon RDS que l'application utilisera pour stocker les données. Pour Moteur de base de données, choisissez sqlserver-ex.

#### Pour ajouter une instance DB à votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.

4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Choisissez Apply.

## Publiez votre application sur Elastic Beanstalk

Utilisez AWS Toolkit for Visual Studio pour publier votre application sur Elastic Beanstalk.

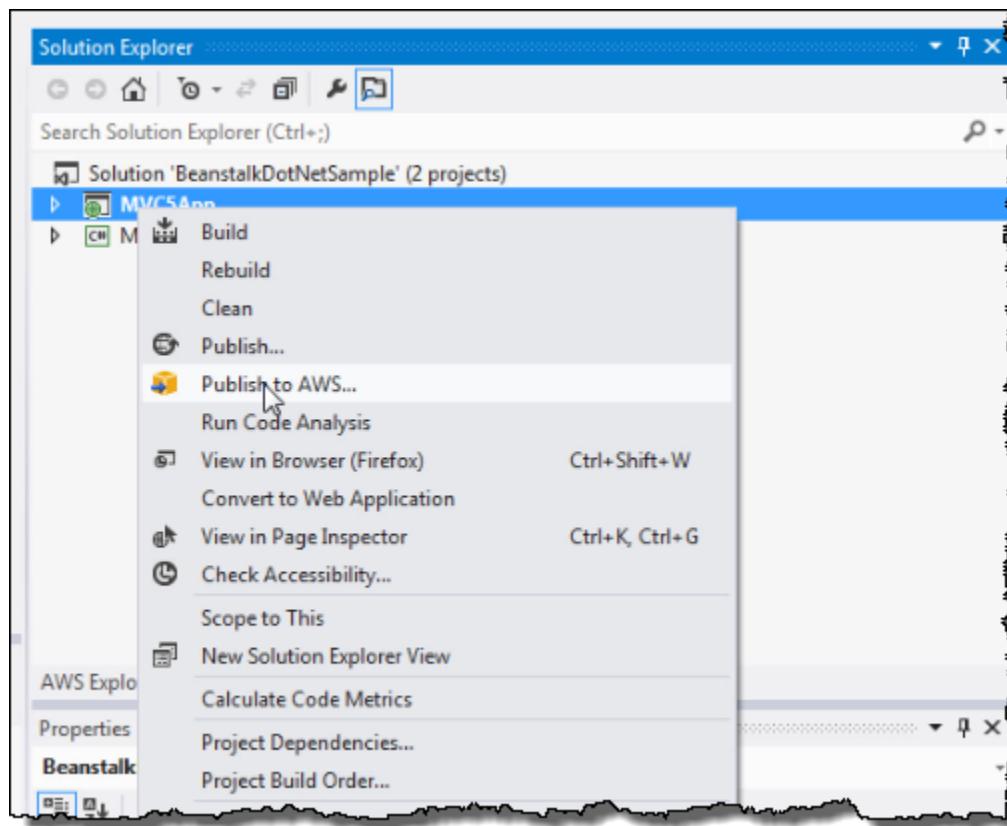
Pour publier votre application sur Elastic Beanstalk

1. Vérifiez que votre environnement s'est bien lancé en vérifiant le statut Health (Santé) dans la console Elastic Beanstalk. Il devrait être OK (vert).
2. Dans Visual Studio, ouvrez BeanstalkDotNetSample.sln.

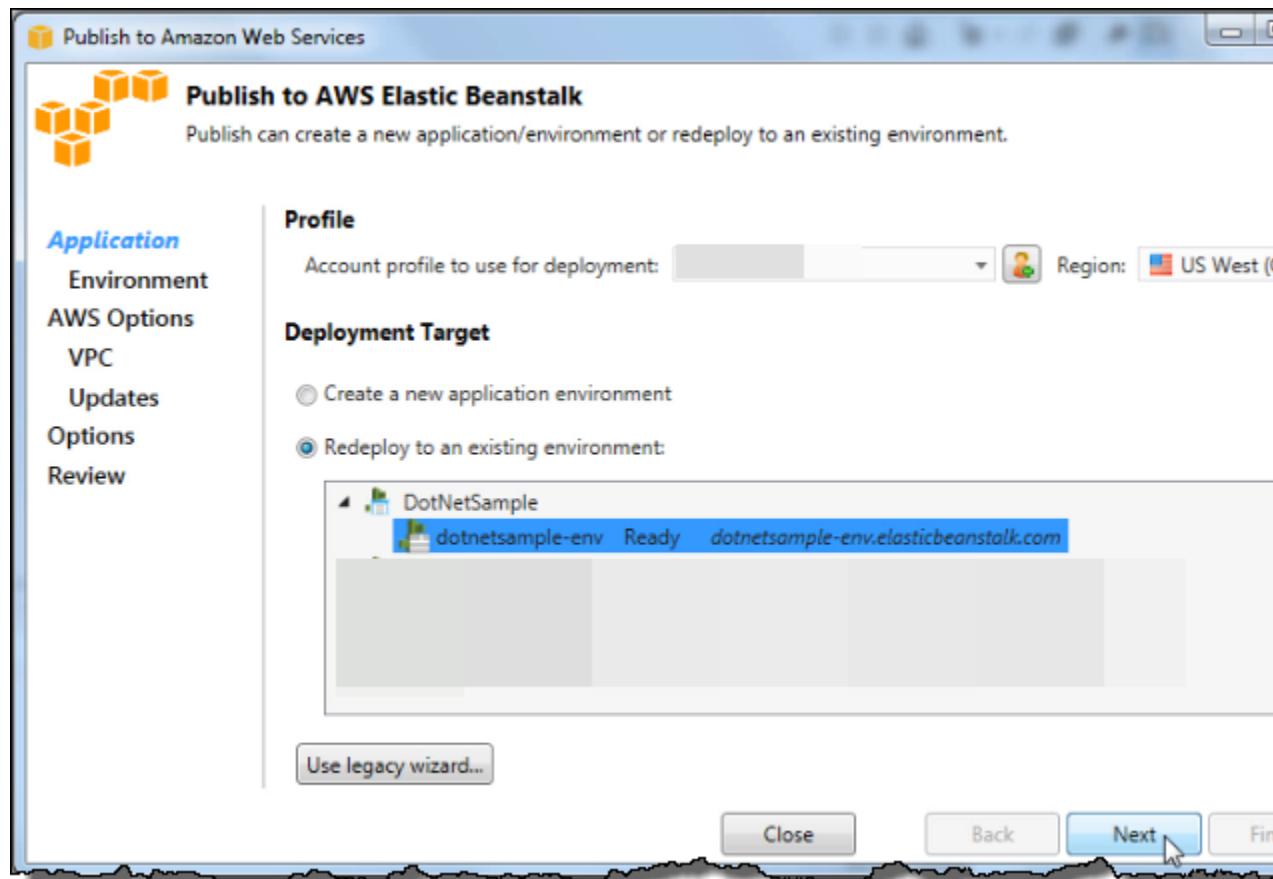
### Note

Si vous ne l'avez pas déjà fait, vous pouvez obtenir l'échantillon [ici](#).

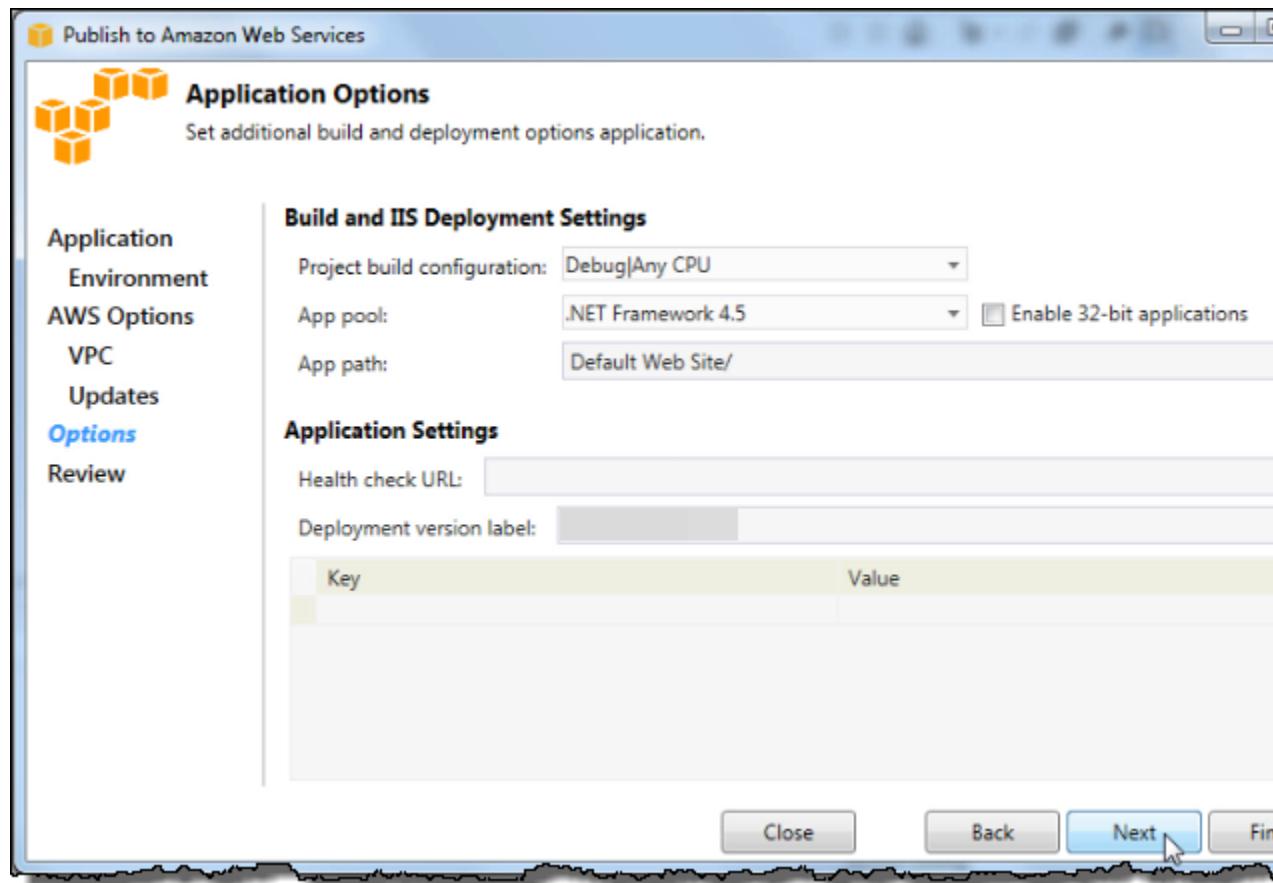
3. Dans le menu Affichage, choisissez Explorateur de solutions.
4. Développez Solution 'BeanstalkDotNetSample' (2 projets).
5. Ouvrez le menu contextuel (clic droit) pour MVC5App, puis choisissez Publish to AWS (Publier dans AWS).



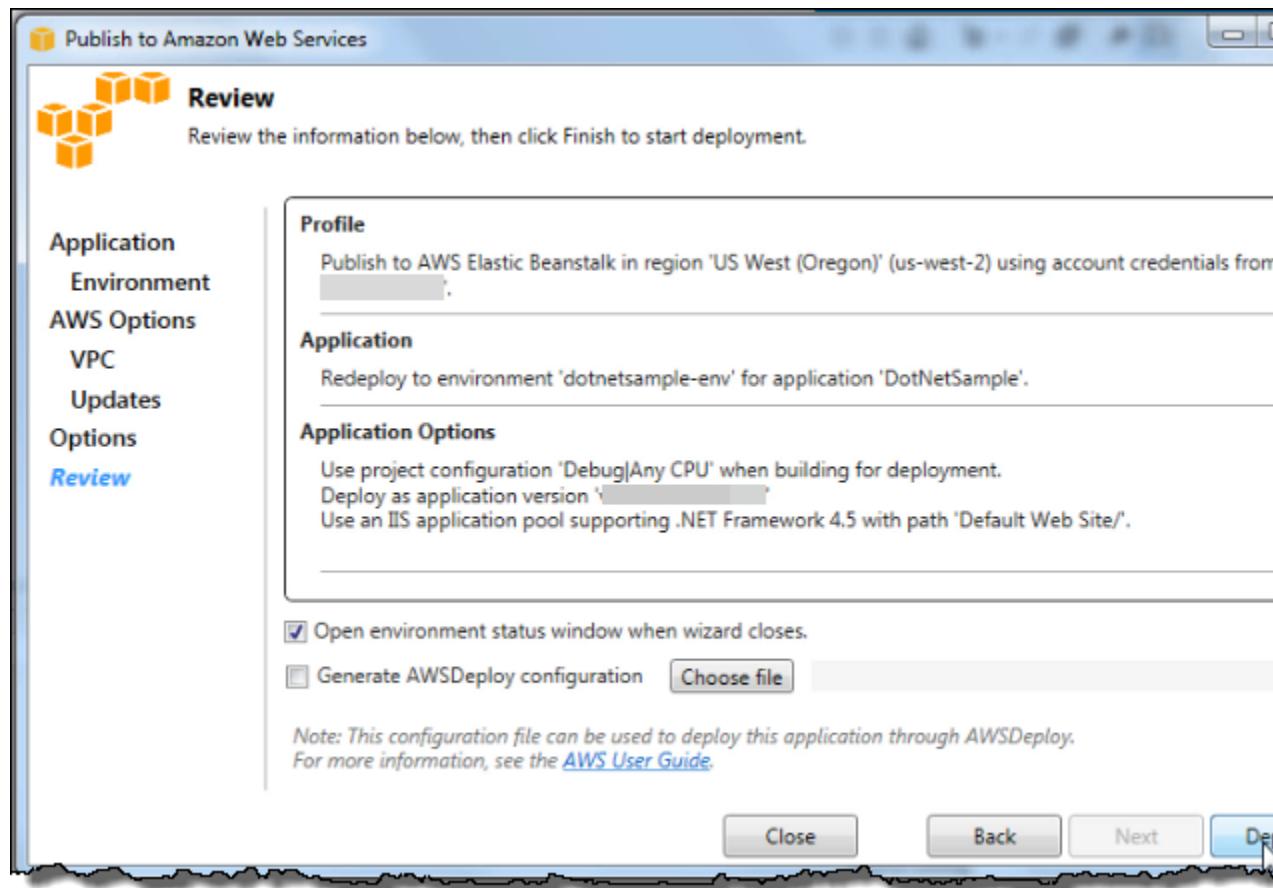
6. Sur la page Publish to AWS Elastic Beanstalk (Publier dans AWS Elastic Beanstalk), pour Deployment Target (Cible de déploiement), sélectionnez l'environnement que vous venez de créer, puis sélectionnez Next (Suivant).



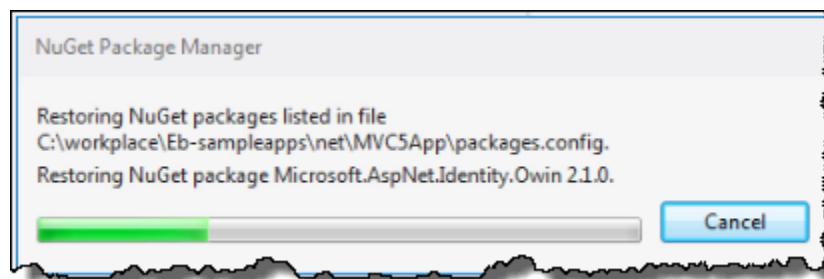
7. Sur la page Application Options, acceptez toutes les valeurs par défaut, puis choisissez Next.



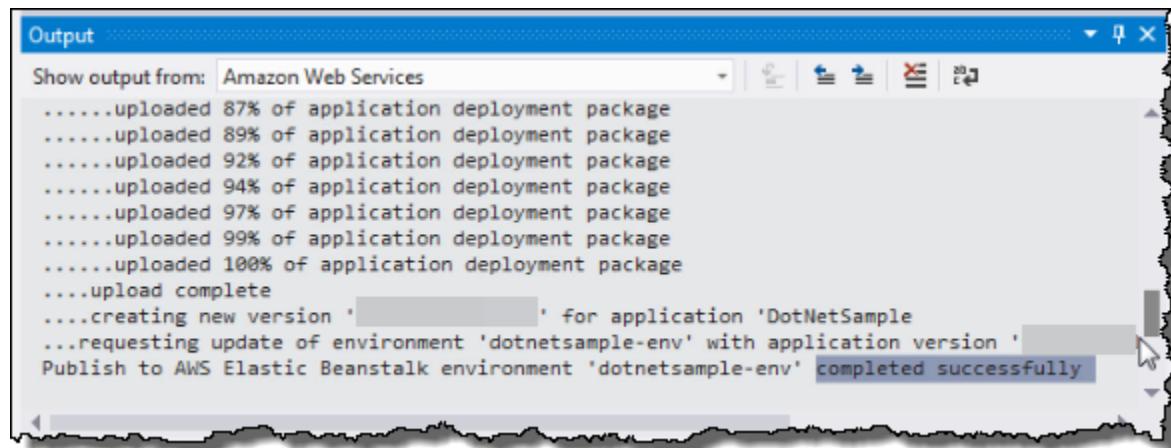
8. Sur la page Révision, choisissez Déployer.



9. Si vous souhaitez surveiller l'état de déploiement, utilisez le NuGet Package Manager dans Visual Studio.



Lorsque l'application a été déployée avec succès, la zone Sortie affiche opération effectuée correctement.



10. Revenir à la console Elastic Beanstalk. Dans le volet de navigation, choisissez Go to environment (Aller à l'environnement).

Votre application ASP.NET s'affiche dans un nouvel onglet.

The screenshot shows a web browser window for an AWS Elastic Beanstalk application. The header has links for "AWS Elastic Beanstalk", "Home", "About", and "Contact". The main content area displays the following text:

# My ASP.NET Application

Congratulations!, Your AWS Elastic Beanstalk ASP.NET application is now running on your own environment in the AWS Cloud.

In-Depth Introduction

AWS re:Invent APP201 Going Zero to Sixty with AWS Elastic

## Nettoyage de vos ressources AWS

Après que votre application s'est déployée avec succès, obtenez davantage d'informations sur Elastic Beanstalk en [regardant la vidéo](#) dans l'application.

Si vous avez terminé d'utiliser Elastic Beanstalk pour l'instant, vous pouvez arrêter votre environnement .NET.

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate Environment (Arrêter l'environnement).

Elastic Beanstalk nettoie toutes les ressources AWS associées à votre environnement, y compris les instances EC2, l'instance de base de données, l'équilibrage de charge, les groupes de sécurité, les alarmes CloudWatch, etc.

Pour de plus amples informations, veuillez consulter [Création et déploiement d'applications .NET sur Elastic Beanstalk \(p. 192\)](#), le [blog de développement AWS .NET](#), ou le [blog de gestion d'application AWS](#).

## Tutoriel : Déploiement d'une application ASP.NET Core avec Elastic Beanstalk

Ce didacticiel vous guide tout au long du processus de création d'une nouvelle application ASP.NET Core et de son déploiement dans AWS Elastic Beanstalk.

Tout d'abord, vous allez utiliser l'outil de ligne de commande `dotnet` du kit SDK .NET Core pour générer une application de ligne de commande .NET Core de base, installer les dépendances, compiler le code et exécuter les applications localement. Ensuite, vous allez créer la classe `Program.cs` par défaut et ajouter une classe `Startup.cs` et des fichiers de configuration ASP.NET pour créer une application qui répond aux demandes HTTP avec ASP.NET et IIS.

Enfin, Elastic Beanstalk utilise un [manifeste de déploiement \(p. 200\)](#) pour configurer les déploiements pour des applications .NET Core, des applications personnalisées ainsi que plusieurs applications .NET Core ou MSBuild se trouvant sur un même serveur. Pour déployer une application .NET Core dans un environnement Windows Server, vous ajoutez une archive de site au bundle de fichiers source d'une application avec un manifeste de déploiement. La commande `dotnet publish` génère des classes compilées et des dépendances que vous pouvez regrouper avec un fichier `web.config` afin de créer une archive de site. Le manifeste de déploiement indique à Elastic Beanstalk le chemin d'accès sur lequel le site doit être exécuté. Il permet également de configurer des groupes d'applications et d'exécuter plusieurs applications sur des chemins différents.

Le code source de l'application est disponible ici : [dotnet-core-tutorial-source.zip](#)

Le bundle de fichiers source pouvant être déployé est disponible ici : [dotnet-core-tutorial-bundle.zip](#)

### Sections

- [Prerequisites \(p. 213\)](#)
- [Génération d'un projet .NET core \(p. 214\)](#)

- [Lancer un environnement Elastic Beanstalk \(p. 214\)](#)
- [Mise à jour du code source \(p. 215\)](#)
- [Déploiement de votre application \(p. 219\)](#)
- [Cleanup \(p. 220\)](#)
- [Étapes suivantes \(p. 221\)](#)

## Prerequisites

Ce didacticiel utilise le kit SDK .NET Core pour générer une application .NET Core de base, l'exécuter localement et créer un package pouvant être déployé.

### Requirements

- .NET Core (x64) 1.0.1, 2.0.0 ou ultérieur

### Pour installer le kit SDK .NET core

1. Téléchargez le programme d'installation à partir de [microsoft.com/net/core](https://microsoft.com/net/core). Choisissez Windows. Choisissez Download .NET SDK (Télécharger le kit SDK .NET).
2. Exécutez le programme d'installation et suivez les instructions.

Ce tutoriel utilise un utilitaire ZIP de ligne de commande pour créer un bundle de fichiers source que vous pouvez déployer sur Elastic Beanstalk. Pour utiliser la commande `zip` sous Windows, vous pouvez installer `UnxUtils`, une collection simple d'utilitaires de ligne de commande très utiles tels que `zip` et `ls`. Sinon, vous pouvez [utiliser l'Explorateur Windows \(p. 416\)](#) ou tout autre utilitaire ZIP pour créer des archives de bundle source.

### Pour installer UnxUtils

1. Téléchargement [UnxUtils](#).
2. Extrayez l'archive dans un répertoire local. Par exemple, `C:\Program Files (x86)`.
3. Ajoutez le chemin d'accès aux fichiers binaires à votre variable utilisateur PATH sous Windows. Par exemple, `C:\Program Files (x86)\UnxUtils\usr\local\wbin`.
  - a. Appuyez sur la touche Windows et entrez **environment variables**.
  - b. Choisissez Modifier les variables d'environnement pour votre compte.
  - c. Choisissez PATH, puis Modifier.
  - d. Ajoutez des chemins d'accès dans le champ Valeur de la variable, en les séparant par des points virgules. Par exemple: `C:\item1\path;C:\item2\path`
  - e. Choisissez OK deux fois pour appliquer les nouveaux paramètres.
  - f. Fermez toutes les fenêtres d'invite de commande en cours d'exécution, puis rouvrez une fenêtre d'invite de commande.
4. Ouvrez une nouvelle fenêtre d'invite de commande et exécutez la commande `zip` pour vérifier qu'elle fonctionne.

```
> zip -h
Copyright (C) 1990-1999 Info-ZIP
Type 'zip "-L"' for software license.
...
```

## Génération d'un projet .NET core

Utilisez l'outil de ligne de commande dotnet pour générer un nouveau projet d'application web C# .NET Core. et l'exécuter localement. L'application .NET Core par défaut est un utilitaire de ligne de commande qui imprime Hello World! puis se ferme.

Pour générer un nouveau projet .NET core

1. Ouvrez une nouvelle fenêtre d'invite de commande et accédez à votre dossier utilisateur.

```
> cd %USERPROFILE%
```

2. Utilisez la commande dotnet new pour générer un nouveau projet .NET Core.

```
C:\Users\username> dotnet new console -o dotnet-core-tutorial
Content generation time: 65.0152 ms
The template "Console Application" created successfully.
C:\Users\username> cd dotnet-core-tutorial
```

3. Utilisez la commande dotnet restore pour installer les dépendances.

```
C:\Users\username\dotnet-core-tutorial> dotnet restore
Restoring packages for C:\Users\username\dotnet-core-tutorial\dotnet-core-
tutorial.csproj...
Generating MSBuild file C:\Users\username\dotnet-core-tutorial\obj\dotnet-core-
tutorial.csproj.nuget.g.props.
Generating MSBuild file C:\Users\username\dotnet-core-tutorial\obj\dotnet-core-
tutorial.csproj.nuget.g.targets.
Writing lock file to disk. Path: C:\Users\username\dotnet-core-tutorial\obj
\project.assets.json
Restore completed in 1.25 sec for C:\Users\username\dotnet-core-tutorial\dotnet-core-
tutorial.csproj.

NuGet Config files used:
  C:\Users\username\AppData\Roaming\NuGet\NuGet.Config
  C:\Program Files (x86)\NuGet\Config\Microsoft.VisualStudio.Offline.config
Feeds used:
  https://api.nuget.org/v3/index.json
  C:\Program Files (x86)\Microsoft SDKs\NuGetPackages\
```

4. Utilisez la commande dotnet run pour créer et exécuter l'application localement.

```
C:\Users\username\dotnet-core-tutorial> dotnet run
Hello World!
```

## Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour lancer un environnement Elastic Beanstalk. Pour cet exemple, vous utiliserez une plateforme .NET. Une fois que vous avez lancé et configuré votre environnement, vous pouvez déployer du nouveau code source à tout moment.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créer une application.

La création de l'environnement prend environ 10 minutes. Pendant ce temps, vous pouvez mettre à jour votre code source.

## Mise à jour du code source

Modifiez l'application par défaut dans une application web qui utilise ASP.NET et IIS.

- ASP.NET est l'infrastructure de site web pour .NET.
- IIS est le serveur web qui exécute l'application sur les instances Amazon EC2 dans votre environnement Elastic Beanstalk.

Les exemples de code source à suivre sont disponibles ici : [dotnet-core-tutorial-source.zip](#)

### Note

La procédure suivante illustre comment convertir le code du projet en une application web. Pour simplifier le processus, vous pouvez générer le projet sous la forme d'une application web dès le départ. Dans la section précédente [Génération d'un projet .NET core \(p. 214\)](#), modifiez la commande de l'étape `dotnet new` avec la commande suivante.

```
C:\Users\username> dotnet new web -o dotnet-core-tutorial
```

Pour ajouter la prise en charge d'ASP.NET et IIS à votre code

1. Copiez `Program.cs` dans le répertoire de votre application pour l'exécuter en tant que générateur d'hôte web.

Example c:\users\username\dotnet-core-tutorial\Program.cs

```
using System;
using Microsoft.AspNetCore.Hosting;
using System.IO;

namespace aspnetcoreapp
{
    public class Program
    {
        public static void Main(string[] args)
        {
            var host = new WebHostBuilder()
                .UseKestrel()
                .UseContentRoot(Directory.GetCurrentDirectory())
                .UseIISIntegration()
                .UseStartup<Startup>()
                .Build();

            host.Run();
        }
    }
}
```

2. Ajoutez `Startup.cs` pour exécuter un site web ASP.NET.

Example c:\users\username\dotnet-core-tutorial\Startup.cs

```
using System;
using Microsoft.AspNetCore.Builder;
using Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Http;

namespace aspnetcoreapp
{
    public class Startup
    {
        public void Configure(IApplicationBuilder app)
        {
            app.Run(context =>
            {
                return context.Response.WriteAsync("Hello from ASP.NET Core!");
            });
        }
    }
}
```

3. Ajoutez le fichier web.config pour configurer le serveur IIS.

Example c:\users\username\dotnet-core-tutorial\web.config

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <system.webServer>
        <handlers>
            <add name="aspNetCore" path="*" verb="*" modules="AspNetCoreModule"
resourceType="Unspecified" />
        </handlers>
        <aspNetCore processPath="dotnet" arguments=".\\dotnet-core-
tutorial.dll" stdoutLogEnabled="false" stdoutLogFile=".\\logs\\stdout"
forwardWindowsAuthToken="false" />
    </system.webServer>
</configuration>
```

4. Ajoutez dotnet-core-tutorial.csproj, qui inclut l'intergiciel IIS et le fichier web.config à partir de la sortie de dotnet publish.

#### Note

L'exemple suivant a été développé à l'aide de .NET Core Runtime 2.2.1. Vous devrez peut-être modifier le TargetFramework ou les valeurs d'attribut Version dans les éléments PackageReference pour correspondre à la version de .NET Core Runtime que vous utilisez dans vos projets personnalisés.

Example c:\users\username\dotnet-core-tutorial\dotnet-core-tutorial.csproj

```
<Project Sdk="Microsoft.NET.Sdk">

    <PropertyGroup>
        <OutputType>Exe</OutputType>
        <TargetFramework>netcoreapp2.2</TargetFramework>
    </PropertyGroup>

    <ItemGroup>
        <PackageReference Include="Microsoft.AspNetCore.Server.Kestrel"
Version="2.2.0" />
    </ItemGroup>
```

```
<ItemGroup>
  <PackageReference Include="Microsoft.AspNetCore.Server.IISIntegration"
Version="2.2.0" />
</ItemGroup>

<ItemGroup>
  <None Include="web.config" CopyToPublishDirectory="Always" />
</ItemGroup>

</Project>
```

Ensuite, installez les nouvelles dépendances et exécutez le site web ASP.NET localement.

Pour exécuter le site web localement

1. Utilisez la commande `dotnet restore` pour installer les dépendances.
2. Utilisez la commande `dotnet run` pour créer et exécuter l'application localement.
3. Ouvrez [localhost:5000](http://localhost:5000) pour afficher le site.

Pour exécuter l'application sur un serveur web, vous devez regrouper le code source compilé avec un fichier de configuration `web.config` et les dépendances d'exécution. L'outil `dotnet` fournit une commande `publish` qui rassemble ces fichiers dans un répertoire basé sur la configuration dans `dotnet-core-tutorial.csproj`.

Pour créer votre site web

- Utilisez la commande `dotnet publish` pour générer la sortie du code compilé et des dépendances dans un dossier nommé `site`.

```
C:\users\username\dotnet-core-tutorial> dotnet publish -o site
```

Pour déployer l'application sur Elastic Beanstalk, regroupez l'archive de site avec un [manifeste de déploiement \(p. 200\)](#). Cela indique à Elastic Beanstalk comment l'exécuter.

Pour créer un bundle de fichiers source

1. Ajoutez les fichiers du dossier du site à un fichier ZIP.

Note

Si vous utilisez un utilitaire ZIP différent, veillez à ajouter tous les fichiers au dossier racine de l'archive ZIP résultante. Ceci est requis pour un déploiement réussi de l'application dans votre environnement Elastic Beanstalk.

```
C:\users\username\dotnet-core-tutorial> cd site
C:\users\username\dotnet-core-tutorial\site> zip ../site.zip *
  adding: dotnet-core-tutorial.deps.json (164 bytes security) (deflated 84%)
  adding: dotnet-core-tutorial.dll (164 bytes security) (deflated 59%)
  adding: dotnet-core-tutorial.pdb (164 bytes security) (deflated 28%)
  adding: dotnet-core-tutorial.runtimeconfig.json (164 bytes security) (deflated 26%)
  adding: Microsoft.AspNetCore.Authentication.Abstractions.dll (164 bytes security)
(deflated 49%)
  adding: Microsoft.AspNetCore.Authentication.Core.dll (164 bytes security) (deflated
57%)
  adding: Microsoft.AspNetCore.Connections.Abstractions.dll (164 bytes security)
(deflated 51%)
```

```
adding: Microsoft.AspNetCore.Hosting.Abstractions.dll (164 bytes security) (deflated 49%)
adding: Microsoft.AspNetCore.Hosting.dll (164 bytes security) (deflated 60%)
adding: Microsoft.AspNetCore.Hosting.Server.Abstractions.dll (164 bytes security) (deflated 44%)
adding: Microsoft.AspNetCore.Http.Abstractions.dll (164 bytes security) (deflated 54%)
adding: Microsoft.AspNetCore.Http.dll (164 bytes security) (deflated 55%)
adding: Microsoft.AspNetCore.Http.Extensions.dll (164 bytes security) (deflated 50%)
adding: Microsoft.AspNetCore.Http.Features.dll (164 bytes security) (deflated 50%)
adding: Microsoft.AspNetCore.HttpOverrides.dll (164 bytes security) (deflated 49%)
adding: Microsoft.AspNetCore.Server.IISIntegration.dll (164 bytes security) (deflated 46%)
adding: Microsoft.AspNetCore.Server.Kestrel.Core.dll (164 bytes security) (deflated 63%)
adding: Microsoft.AspNetCore.Server.Kestrel.dll (164 bytes security) (deflated 46%)
adding: Microsoft.AspNetCore.Server.Kestrel.Https.dll (164 bytes security) (deflated 44%)
adding: Microsoft.AspNetCore.Server.Kestrel.Transport.Abstractions.dll (164 bytes security) (deflated 56%)
adding: Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.dll (164 bytes security) (deflated 51%)
adding: Microsoft.AspNetCore.WebUtilities.dll (164 bytes security) (deflated 55%)
adding: Microsoft.Extensions.Configuration.Abstractions.dll (164 bytes security) (deflated 48%)
adding: Microsoft.Extensions.Configuration.Binder.dll (164 bytes security) (deflated 47%)
adding: Microsoft.Extensions.Configuration.dll (164 bytes security) (deflated 46%)
adding: Microsoft.Extensions.Configuration.EnvironmentVariables.dll (164 bytes security) (deflated 46%)
adding: Microsoft.Extensions.Configuration.FileExtensions.dll (164 bytes security) (deflated 47%)
adding: Microsoft.Extensions.DependencyInjection.Abstractions.dll (164 bytes security) (deflated 54%)
adding: Microsoft.Extensions.DependencyInjection.dll (164 bytes security) (deflated 53%)
adding: Microsoft.Extensions.FileProviders.Abstractions.dll (164 bytes security) (deflated 46%)
adding: Microsoft.Extensions.FileProviders.Physical.dll (164 bytes security) (deflated 47%)
adding: Microsoft.Extensions.FileSystemGlobbing.dll (164 bytes security) (deflated 49%)
adding: Microsoft.Extensions.Hosting.Abstractions.dll (164 bytes security) (deflated 47%)
adding: Microsoft.Extensions.Logging.Abstractions.dll (164 bytes security) (deflated 54%)
adding: Microsoft.Extensions.Logging.dll (164 bytes security) (deflated 48%)
adding: Microsoft.Extensions.ObjectPool.dll (164 bytes security) (deflated 45%)
adding: Microsoft.Extensions.Options.dll (164 bytes security) (deflated 53%)
adding: Microsoft.Extensions.Primitives.dll (164 bytes security) (deflated 50%)
adding: Microsoft.Net.Http.Headers.dll (164 bytes security) (deflated 53%)
adding: System.IO.Pipelines.dll (164 bytes security) (deflated 50%)
adding: System.Runtime.CompilerServices.Unsafe.dll (164 bytes security) (deflated 43%)
adding: System.Text.Encodings.Web.dll (164 bytes security) (deflated 57%)
adding: web.config (164 bytes security) (deflated 39%)
C:\users\username\dotnet-core-tutorial\site> cd ..\
```

2. Ajoutez un manifeste de déploiement qui pointe vers l'archive de site.

Example c:\users\username\dotnet-core-tutorial\aws-windows-deployment-manifest.json

```
{  
  "manifestVersion": 1,  
  "deployments": {
```

```
"aspNetCoreWeb": [
{
  "name": "test-dotnet-core",
  "parameters": {
    "AppBundle": "site.zip",
    "iisPath": "/",
    "iisWebSite": "Default Web Site"
  }
}
]
```

3. Utilisez la commande `zip` pour créer un bundle de fichiers source nommé `dotnet-core-tutorial.zip`.

```
C:\users\username\dotnet-core-tutorial> zip dotnet-core-tutorial.zip site.zip aws-
windows-deployment-manifest.json
adding: site.zip (164 bytes security) (stored 0%)
adding: aws-windows-deployment-manifest.json (164 bytes security) (deflated 50%)
```

## Déploiement de votre application

Déployez le bundle de fichiers source dans l'environnement Elastic Beanstalk que vous avez créé.

Vous pouvez télécharger le bundle de fichiers source ici : [dotnet-core-tutorial-bundle.zip](#)

Pour déployer un groupe source

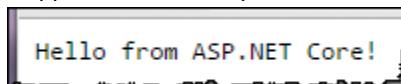
1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

L'application écrit simplement le message `Hello from ASP.NET Core!` à la réponse et le renvoie.



Le lancement d'un environnement crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques,

ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

#### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.region.elasticbeanstalk.com**.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour plus d'informations, consultez [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances](#)

Amazon EC2 (p. 538), les instances de base de données (p. 617), les équilibreurs de charge (p. 562), les groupes de sécurité et les alarmes (p. 562).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

## Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez probablement gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

Si vous utilisez Visual Studio pour développer votre application, vous pouvez également utiliser AWS Toolkit for Visual Studio pour déployer des modifications, gérer vos environnements Elastic Beanstalk et gérer d'autres ressources AWS. Pour plus d'informations, consultez [AWS Toolkit for Visual Studio \(p. 224\)](#).

Pour le développement et les tests, vous pouvez utiliser la fonctionnalité Elastic Beanstalk permettant d'ajouter directement une instance de base de données gérée à votre environnement. Pour savoir comment configurer une base de données dans votre environnement, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, [configurez un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activez HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Ajout d'une instance de base de données Amazon RDS à votre environnement d'application .NET

Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être associée à votre environnement et gérée par Elastic Beanstalk, ou créée et gérée en externe.

Si vous utilisez Amazon RDS pour la première fois, [ajoutez une instance de base de données \(p. 222\)](#) à un environnement de test avec la console Elastic Beanstalk et assurez-vous que votre application peut s'y connecter.

Pour vous connecter à une base de données, [ajoutez le pilote \(p. 223\)](#) à votre application, chargez la classe de pilote dans votre code et [créez une chaîne de connexion \(p. 223\)](#) avec les propriétés

d'environnement fournies par Elastic Beanstalk. La configuration et le code de connexion varient selon le moteur de base de données et l'infrastructure que vous utilisez.

#### Sections

- [Ajout d'une instance de base de données à votre environnement \(p. 222\)](#)
- [Téléchargement d'un pilote \(p. 223\)](#)
- [Connexion à une base de données \(p. 223\)](#)

## Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Choisissez Apply.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour de plus amples informations sur la configuration d'une instance de base de données interne, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

## Téléchargement d'un pilote

Téléchargez et installez le package `EntityFramework` et un pilote de base de données pour votre environnement de développement avec NuGet.

Fournisseurs de base de données Common Entity Framework pour .NET

- SQL Server – `Microsoft.EntityFrameworkCore.SqlServer`
- MySQL – `Pomelo.EntityFrameworkCore.MySql`
- PostgreSQL – `Npgsql.EntityFrameworkCore.PostgreSQL`

## Connexion à une base de données

Elastic Beanstalk fournit des informations de connexion pour les instances de base de données attachées dans les propriétés de l'environnement. Utilisez `ConfigurationManager.AppSettings` pour lire les propriétés et configurer une connexion de base de données.

Example `Helpers.cs` - Méthode de chaîne de connexion

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Linq;
using System.Web;

namespace MVC5App.Models
{
    public class Helpers
    {
        public static string GetRDSConnectionString()
        {
            var appConfig = ConfigurationManager.AppSettings;

            string dbname = appConfig["RDS_DB_NAME"];

            if (string.IsNullOrEmpty(dbname)) return null;

            string username = appConfig["RDS_USERNAME"];
            string password = appConfig["RDS_PASSWORD"];
            string hostname = appConfig["RDS_HOSTNAME"];
            string port = appConfig["RDS_PORT"];

            return "Data Source=" + hostname + ";Initial Catalog=" + dbname + ";User ID=" +
                username + ";Password=" + password + ";";
        }
    }
}
```

Utilisez la chaîne de connexion pour initialiser votre contexte de base de données.

Example `DbContext.cs`

```
using System.Data.Entity;
using System.Security.Claims;
using System.Threading.Tasks;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;
```

```
namespace MVC5App.Models
{
    public class RDSContext : DbContext
    {
        public RDSContext()
            : base(GetRDSConnectionString())
        {
        }

        public static RDSContext Create()
        {
            return new RDSContext();
        }
    }
}
```

## . AWS Toolkit for Visual Studio

Visual Studio fournit des modèles pour différents langages de programmation et types d'applications. Vous pouvez commencer avec n'importe lequel de ces modèles. AWS Toolkit for Visual Studio fournit également trois modèles de projets qui amorcent le développement de votre application : AWS Console Project, AWS Web Project et AWS Empty Project. Pour cet exemple, vous allez créer une application web ASP.NET.

Pour créer un projet d'application web ASP.NET

1. Dans Visual Studio, dans le menu Fichier, cliquez sur Nouveau puis cliquez sur Projet.
2. Dans la boîte de dialogue Nouveau projet, cliquez sur Modèles installés, cliquez sur Visual C#, puis cliquez sur Web. Cliquez sur Application Web ASP.NET vide, tapez un nom de projet puis cliquez sur OK.

Pour exécuter un projet

Effectuez l'une des actions suivantes :

1. Appuyez sur F5.
2. Sélectionnez Démarrer le débogage dans le menu Déboguer.

## Test local

Visual Studio vous facilite le test de votre application localement. Pour tester ou exécuter des applications web ASP.NET, vous avez besoin d'un serveur web. Visual Studio fournit plusieurs options, telles qu'Internet Information Services (IIS), IIS Express ou le serveur de développement intégré de Visual Studio. Pour en savoir plus sur chacune de ces options et décider laquelle vous convient le mieux, consultez [Web Servers in Visual Studio for ASP.NET Web Projects](#).

## Créer un environnement Elastic Beanstalk

Après avoir testé votre application, vous êtes prêt à la déployer dans Elastic Beanstalk.

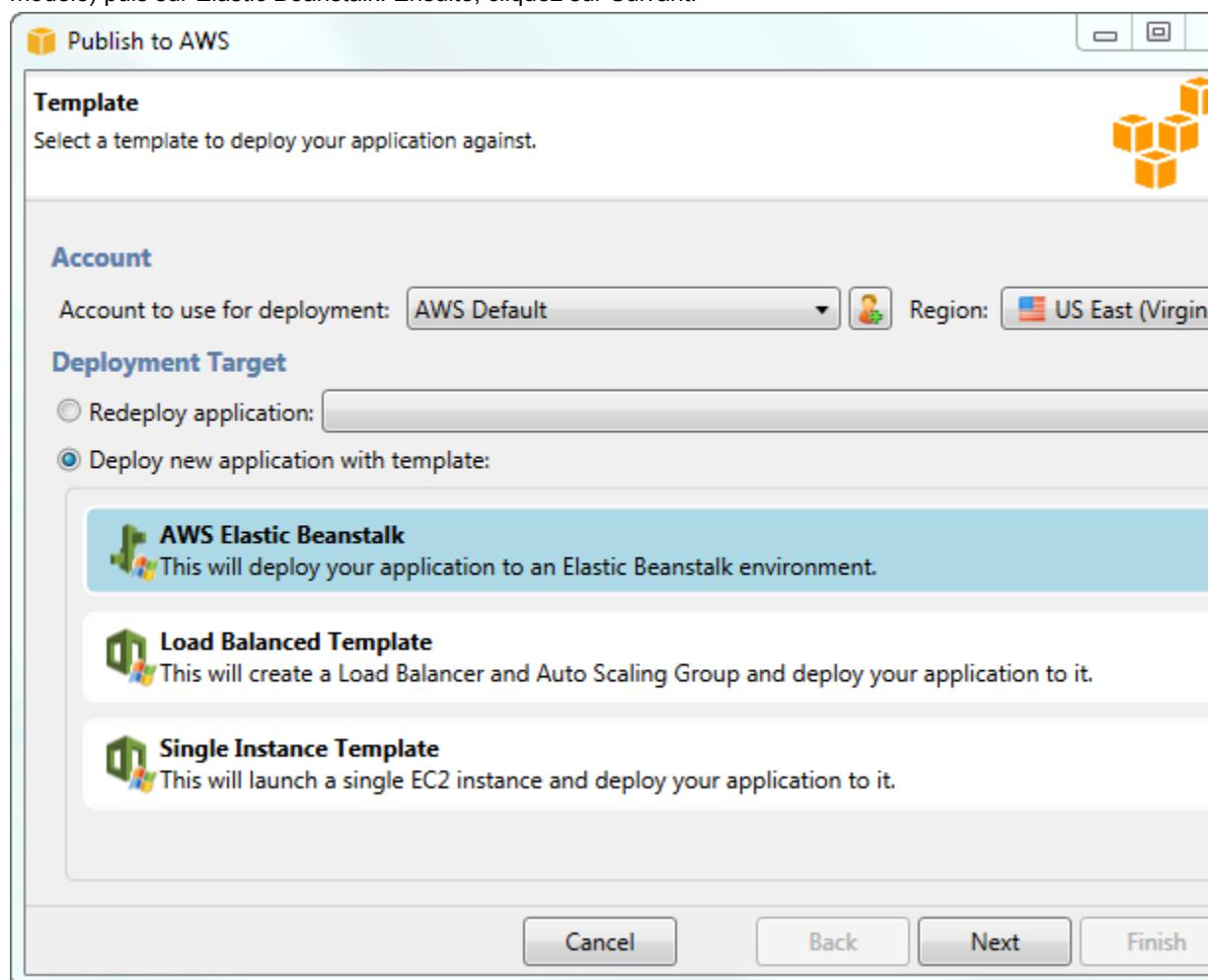
### Note

Un [fichier de configuration \(p. 737\)](#) doit faire partie du projet pour être inclus dans l'archive. Sinon, au lieu d'inclure les fichiers de configuration dans le projet, vous pouvez utiliser Visual Studio pour déployer tous les fichiers dans le dossier de projet. Dans l'Explorateur de solutions, cliquez avec le bouton droit sur le nom du projet, puis cliquez sur Propriétés. Cliquez sur l'onglet Package/Publication Web. Dans la section Items to deploy (Éléments à déployer),

sélectionnez All Files in the Project Folder (Tous les fichiers dans le dossier du projet) dans la liste déroulante.

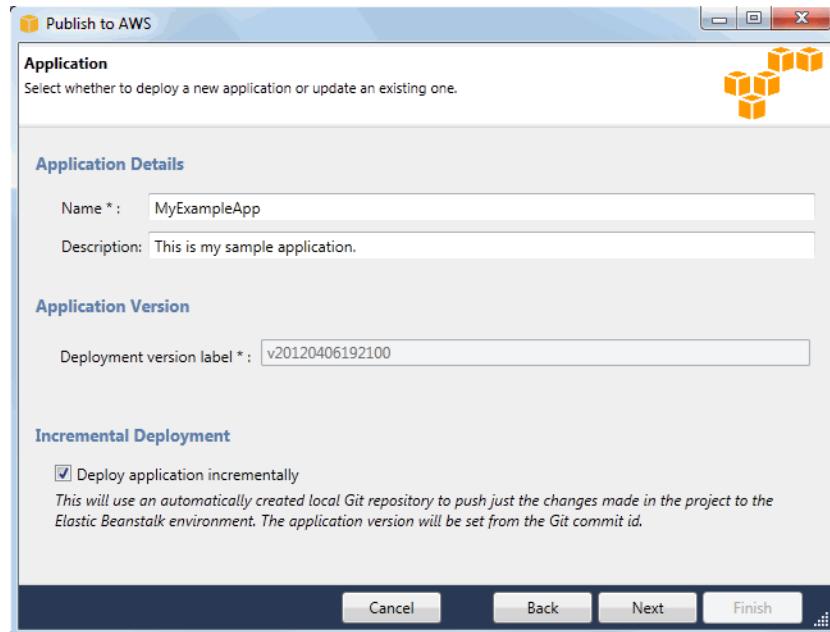
Pour déployer votre application sur Elastic Beanstalk avec AWS Toolkit for Visual Studio.

1. Dans Solution Explorer (Explorateur de solutions), cliquez avec le bouton droit de la souris sur votre application, puis sélectionnez Publish to AWS (Publier dans AWS).
2. Dans l'assistant Publish to AWS (Publier dans AWS), saisissez les informations de votre compte.
  - a. Pour AWS account to use for deployment (compte AWS à utiliser pour le déploiement), sélectionnez votre compte ou Other (Autre) pour saisir de nouvelles informations de compte.
  - b. Pour Region (Région), sélectionnez la région où vous souhaitez déployer votre application. Pour de plus amples informations sur les régions AWS disponibles, veuillez consulter [AWS Elastic Beanstalk Endpoints and Quotas](#) (Points de terminaison et quotas AWS Elastic Beanstalk) dans la référence générale AWS. Si vous sélectionnez une région qui n'est pas prise en charge par Elastic Beanstalk, l'option de déploiement vers Elastic Beanstalk devient indisponible.
  - c. Cliquez sur Deploy new application with template (Déployer la nouvelle application avec le modèle) puis sur Elastic Beanstalk. Ensuite, cliquez sur Suivant.



3. Sur la page Application, entrez les détails de votre application.
  - a. Pour Name (Nom), tapez le nom de l'application.
  - b. Pour Description, entrez une description de l'application. Cette étape est facultative.

- c. L'étiquette de version de l'application s'affiche automatiquement dans l'étiquette de version Deployment.
- d. Sélectionnez Deploy application incrementally (Déployer l'application de manière incrémentielle) pour déployer uniquement les fichiers modifiés. Un déploiement incrémentiel est plus rapide, car vous mettez à jour uniquement les fichiers qui ont été modifiés au lieu de tous les fichiers. Si vous sélectionnez cette option, une version de l'application est définie à partir de l'ID de validation Git. Si vous choisissez de ne pas déployer votre application de manière incrémentielle, vous pouvez alors mettre à jour l'étiquette de version dans la zone Deployment version label (Étiquette de version Déploiement).



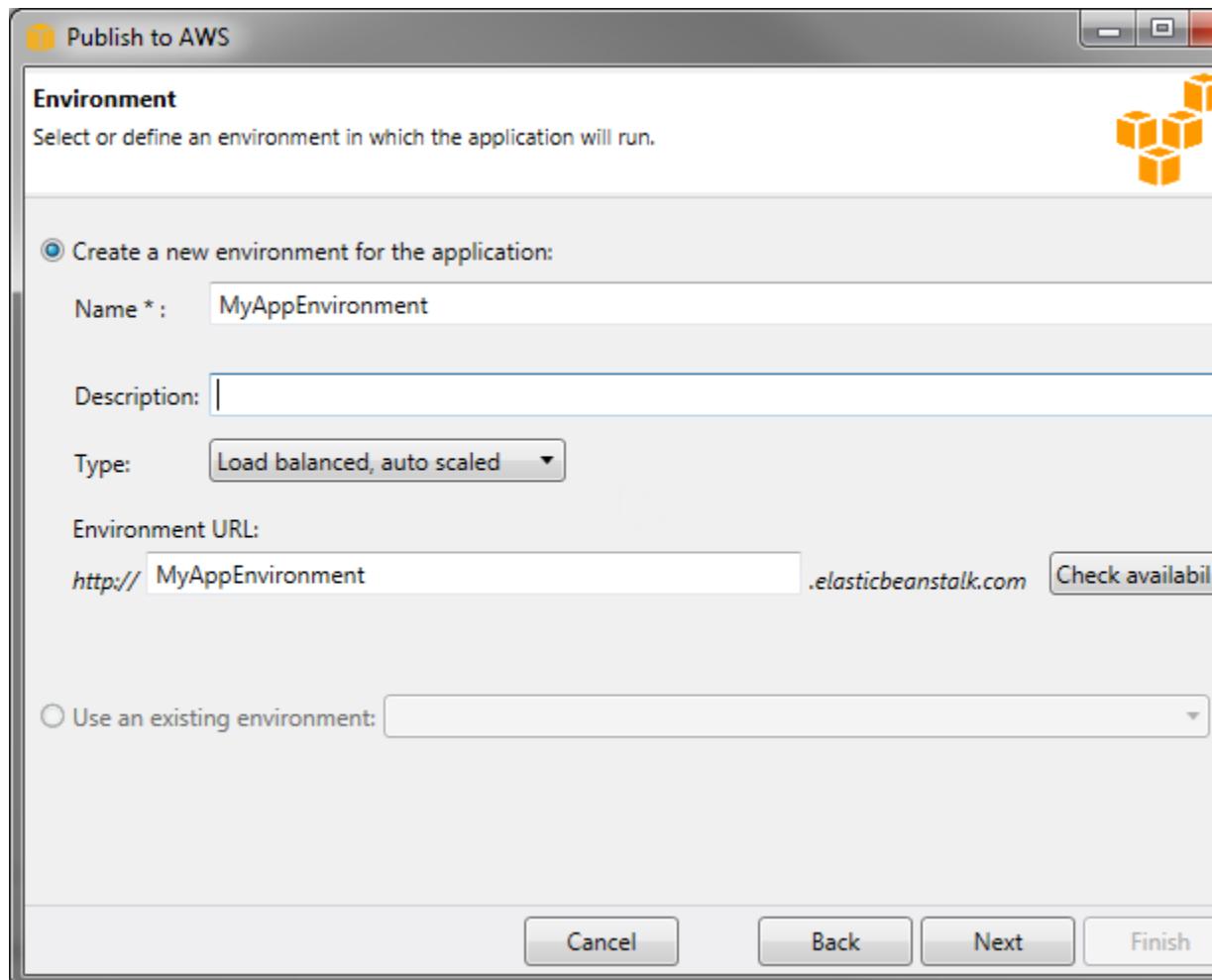
- e. Cliquez sur Suivant.
- 4. Sur la page Environment (Environnement), décrivez les détails de votre environnement.
  - a. Sélectionnez Create a new environment for this application (Créer un environnement pour cette application).
  - b. Pour Name (Nom), tapez un nom unique pour votre environnement.
  - c. Pour Description, qualifiez votre environnement. Cette étape est facultative.
  - d. Sélectionnez le Type d'environnement que vous voulez.

Vous avez le choix entre deux types d'environnement : Load balanced, auto scaled (Équilibrage de charge, scalabilité automatique) ou Single instance (Instance unique). Pour plus d'informations, consultez [Types d'environnement \(p. 519\)](#).

#### Note

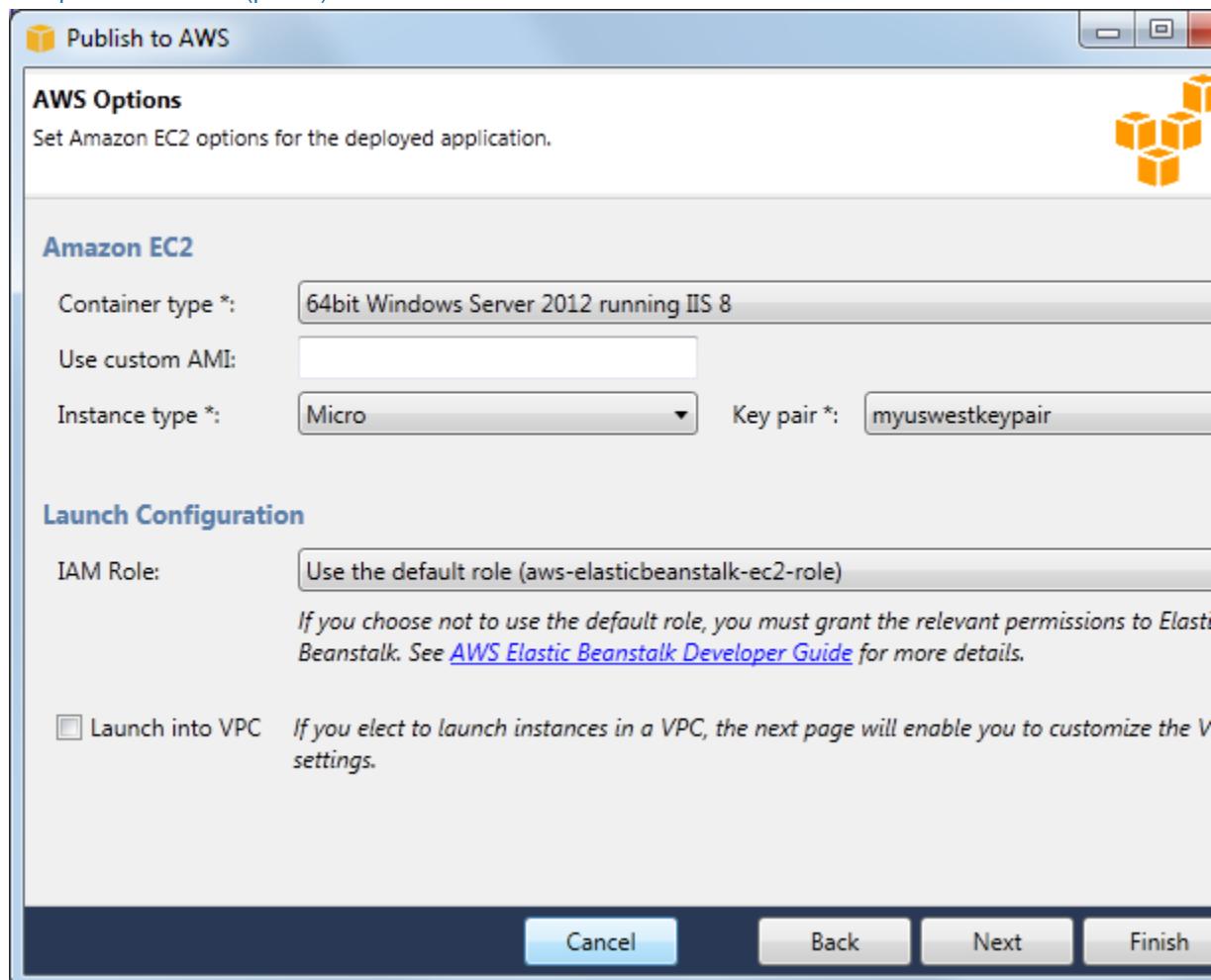
Pour les environnements instance unique, les paramètres URL d'équilibrage de charge, de mise à l'échelle automatique et de vérification de l'état ne s'appliquent pas.

- e. L'URL d'environnement s'affiche automatiquement dans Environment URL (URL de l'environnement) une fois que vous déplacez votre curseur dans cette zone.
- f. Cliquez sur Check availability (Vérifier la disponibilité) pour vous assurer que l'URL d'environnement est disponible.

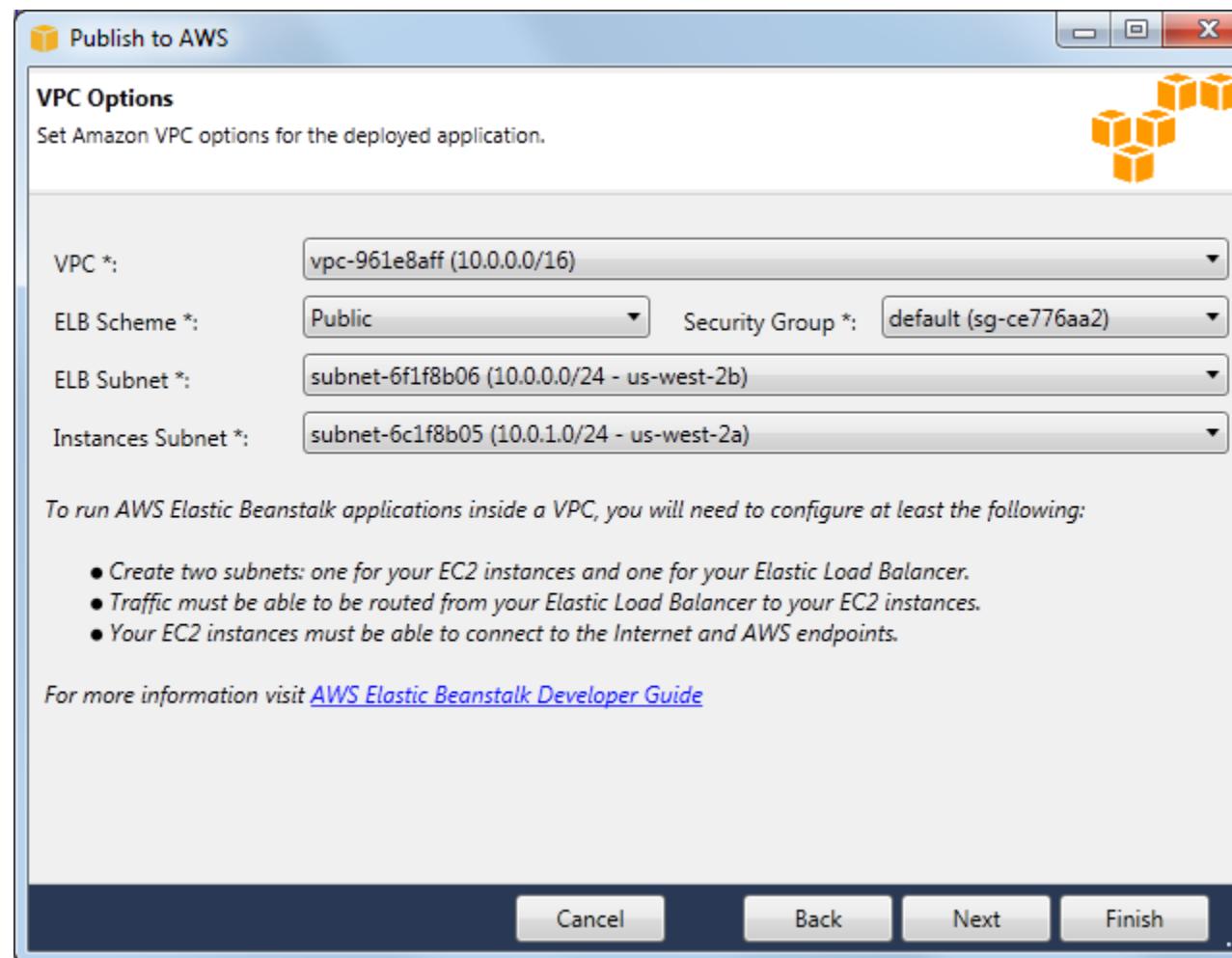


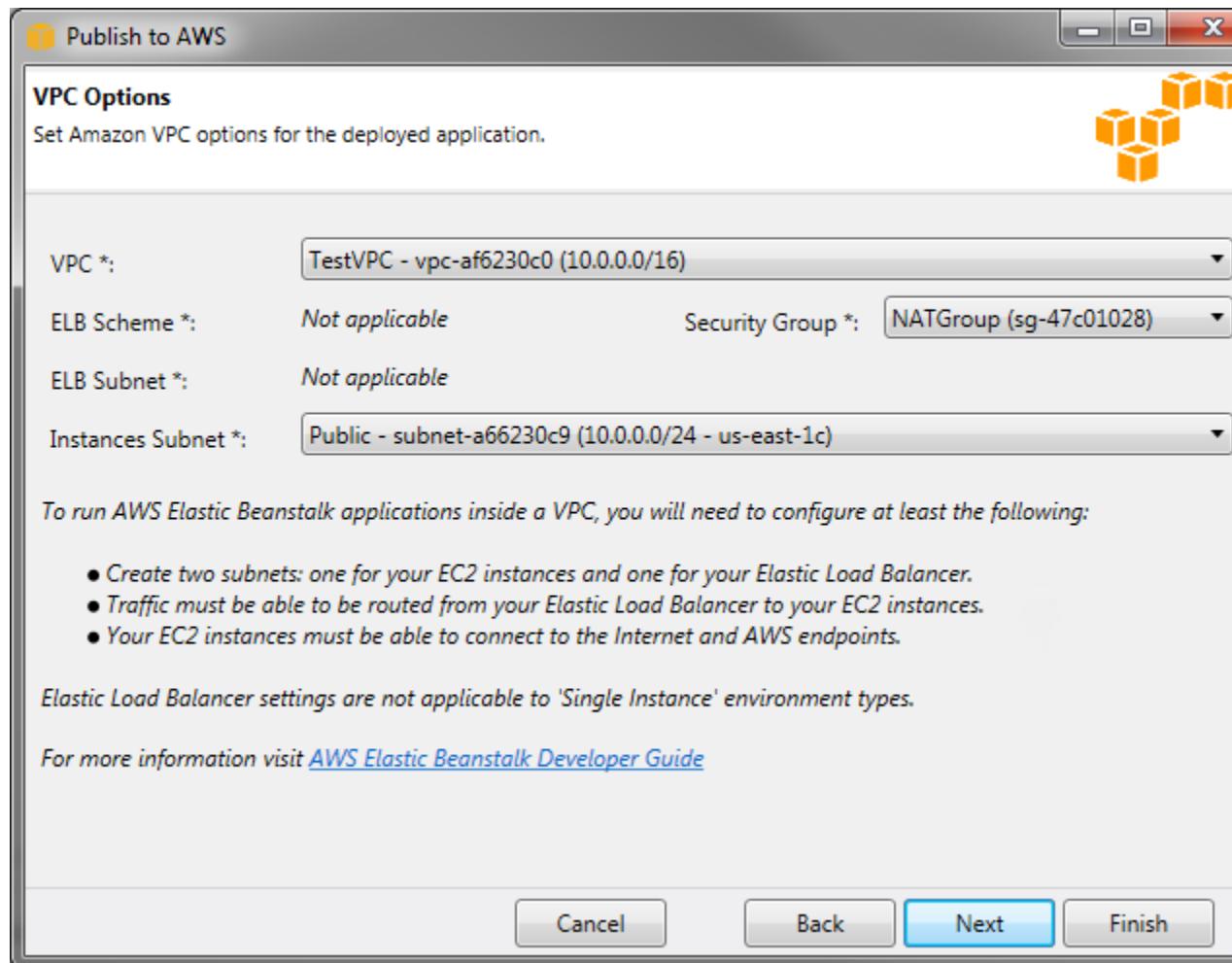
- g. Cliquez sur Suivant.
5. Sur la page AWS Options (Options AWS), configurez des informations de sécurité et des options supplémentaires pour votre déploiement.
- Pour Container Type (Type de conteneur), sélectionnez 64bit Windows Server 2012 running IIS 8 (Serveur Windows 2012 64 bits exécutant IIS 8) ou 64bit Windows Server 2008 running IIS 7.5 (Serveur Windows 2008 64 bits exécutant IIS 7.5).
  - Pour Instance Type (Type d'instance), sélectionnez Micro.
  - Pour Key pair (Paire de clés), sélectionnez Create new key pair (Créer une paire de clés). Tapez un nom pour la nouvelle paire de clés. Dans cet exemple, nous utiliserons **myuswestkeypair**, puis nous cliquerons sur OK. Une paire de clés permet un accès bureau à distance à vos instances Amazon EC2. Pour de plus amples informations sur les paires de clés Amazon EC2, veuillez consulter [Utilisation des informations d'identification](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.
  - Sélectionner un profil d'instance.
- Si vous n'avez pas de profil d'instance, sélectionnez Create a default instance profile (Créer un profil d'instance par défaut). Pour de plus amples informations sur l'utilisation des profils d'instance avec Elastic Beanstalk, veuillez consulter [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#).
- e. Si vous avez un VPC personnalisé que vous souhaitez utiliser avec votre environnement, cliquez sur Launch into VPC (Lancer dans le VPC). Vous pouvez configurer les informations de VPC sur

la page suivante. Pour de plus amples informations sur Amazon VPC, veuillez consulter [Amazon Virtual Private Cloud \(Amazon VPC\)](#). Pour afficher la liste des types de conteneurs non hérités pris en charge, consultez la section called "Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?" (p. 507)



- f. Cliquez sur Suivant.
6. Si vous avez choisi de lancer votre environnement à l'intérieur d'un VPC, la page VPC Options (Options du VPC) s'affiche. Sinon, la page Additional Options (Options supplémentaires) s'affiche. Ici, vous allez configurer vos options de VPC.





- Sélectionnez l'ID de VPC du VPC dans lequel vous souhaitez lancer votre environnement.
- Pour un environnement à charge équilibrée et évolutif, sélectionnez privé pour ELB Scheme si vous ne souhaitez pas que votre Elastic Load Balancer soit disponible sur Internet.

Pour un environnement instance unique, cette option n'est pas applicable, car l'environnement n'a pas d'équilibrer de charge. Pour plus d'informations, consultez [Types d'environnement \(p. 519\)](#).

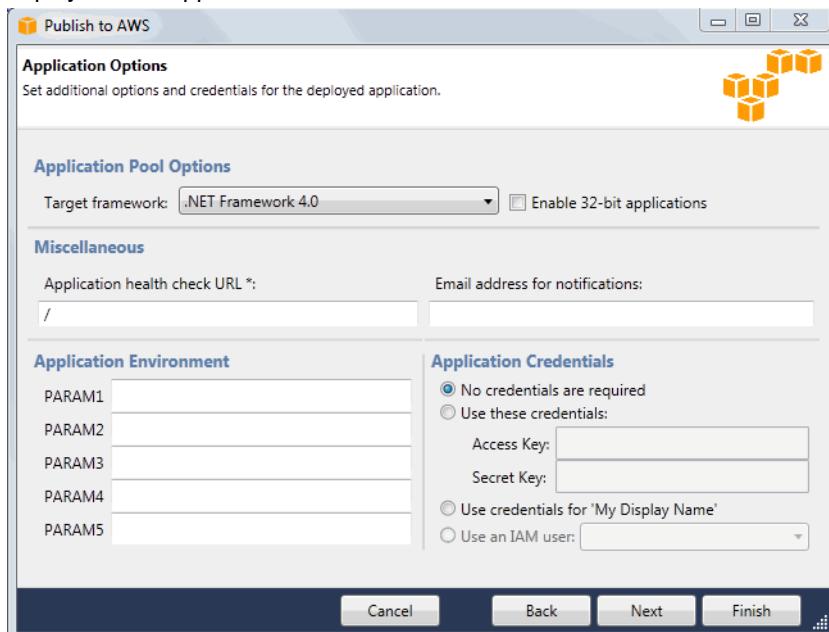
- Pour un environnement à charge équilibrée et évolutif, sélectionnez les sous-réseaux pour l'Elastic Load Balancer et les instances EC2. Si vous avez créé des sous-réseaux privés et publics, assurez-vous que l'Elastic Load Balancer et les instances EC2 sont associés au sous-réseau approprié. Par défaut, Amazon VPC crée un sous-réseau public par défaut à l'aide de 10.0.0.0/24 et d'un sous-réseau privé à l'aide de 10.0.1.0/24. Vous pouvez afficher vos sous-réseaux existants dans la console Amazon VPC à l'adresse <https://console.aws.amazon.com/vpc/>.

Pour un environnement instance unique, votre VPC a besoin uniquement d'un sous-réseau public de l'instance. Sélectionner un sous-réseau pour l'équilibrer de charge n'est pas applicable, car l'environnement n'a pas d'équilibrer de charge. Pour plus d'informations, consultez [Types d'environnement \(p. 519\)](#).

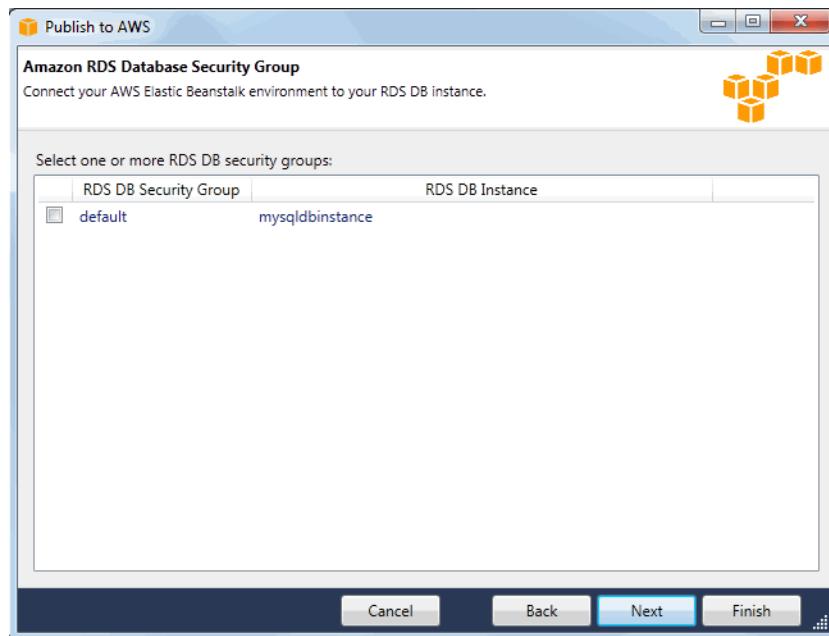
- Pour un environnement à charge équilibrée et évolutif, sélectionnez le groupe de sécurité que vous avez créé pour vos instances, le cas échéant.

Pour un environnement instance unique, vous n'avez pas besoin d'un périphérique NAT. Sélectionnez le groupe de sécurité par défaut. Elastic Beanstalk attribue à l'instance une adresse IP Elastic qui lui permet d'accéder à Internet.

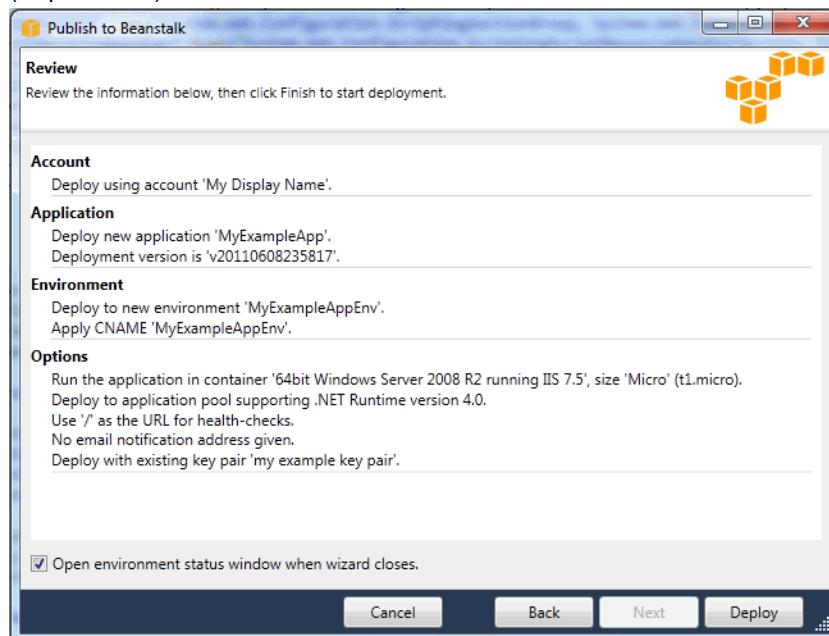
- e. Cliquez sur Suivant.
7. Sur la page Application Options (Options de l'application), configurez les options de votre application.
  - a. Pour l'infrastructure cible, sélectionnez .NET Framework 4.0.
  - b. Elastic Load Balancing utilise une vérification de l'état pour déterminer si les instances Amazon EC2 exécutant votre application sont saines. La vérification de l'état détermine l'état d'intégrité d'une instance en détectant une URL spécifiée à un intervalle défini. Vous pouvez remplacer l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, /myapp/index.aspx) en entrant celle-ci dans la zone URL de vérification de l'état de l'application. Pour plus d'informations sur les vérifications de l'état de l'application, consultez [Vérification de l'état \(p. 571\)](#).
  - c. Tapez une adresse e-mail si vous souhaitez recevoir des notifications Amazon Simple Notification Service (Amazon SNS) des événements importants qui affectent votre application.
  - d. La section Application Environment (Environnement de l'application) vous permet de spécifier des variables d'environnement sur les instances Amazon EC2 exécutant votre application. Ce paramètre permet une plus grande portabilité en éliminant la nécessité de recompiler votre code source pendant que vous vous déplacez entre les environnements.
  - e. Sélectionnez l'option d'informations d'identification des applications que vous voulez utiliser pour déployer votre application.



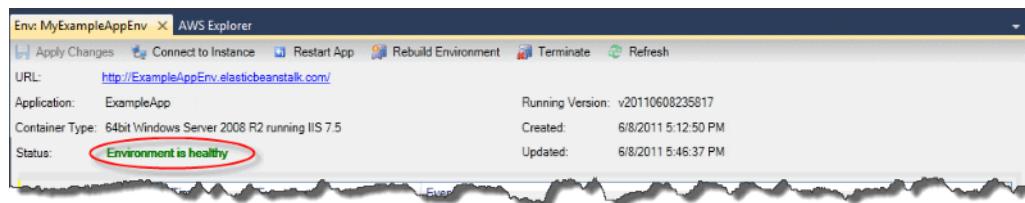
- f. Cliquez sur Suivant.
8. Si vous avez déjà mis en place une base de données Amazon RDS, la page Amazon RDS DB Security Group (Groupe de sécurité DB Amazon RDS) s'affiche. Si vous souhaitez connecter votre environnement Elastic Beanstalk à votre Instance DB Amazon RDS, sélectionnez alors un ou plusieurs groupes de sécurité. Sinon, allez à l'étape suivante. Lorsque vous êtes prêt, cliquez sur Next (Suivant).



9. Passez en revue vos options de déploiement. Si tout est tel que vous le souhaitez, cliquez sur Deploy (Déploiement).



Votre projet ASP.NET sera exporté comme un fichier de déploiement web, téléchargé sur Amazon S3 et inscrit en tant que nouvelle version de l'application avec Elastic Beanstalk. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. Sous l'onglet env:<nom de l'environnement>, vous verrez l'état de votre environnement.



## Résiliation d'un environnement

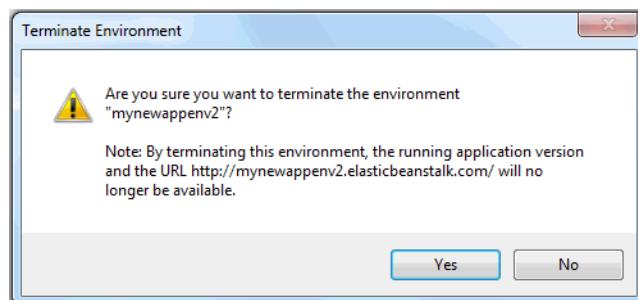
Pour éviter de payer des frais pour des ressources AWS inutilisées, vous pouvez résilier un environnement en cours d'exécution à l'aide d'AWS Toolkit for Visual Studio.

### Note

Vous pouvez toujours lancer un nouvel environnement en utilisant la même version ultérieurement.

### Pour résilier un environnement

1. Développez le nœud Elastic Beanstalk et le nœud d'application dans AWS Explorer (Explorateur AWS). Cliquez avec le bouton droit la souris sur votre environnement d'application et sélectionnez Terminate Environment (Résilier l'environnement).
2. Lorsque vous y êtes invité, cliquez sur Yes (Oui) afin de confirmer que vous souhaitez résilier l'environnement. Il faudra quelques minutes à Elastic Beanstalk pour résilier les ressources AWS en cours d'exécution dans l'environnement.



### Note

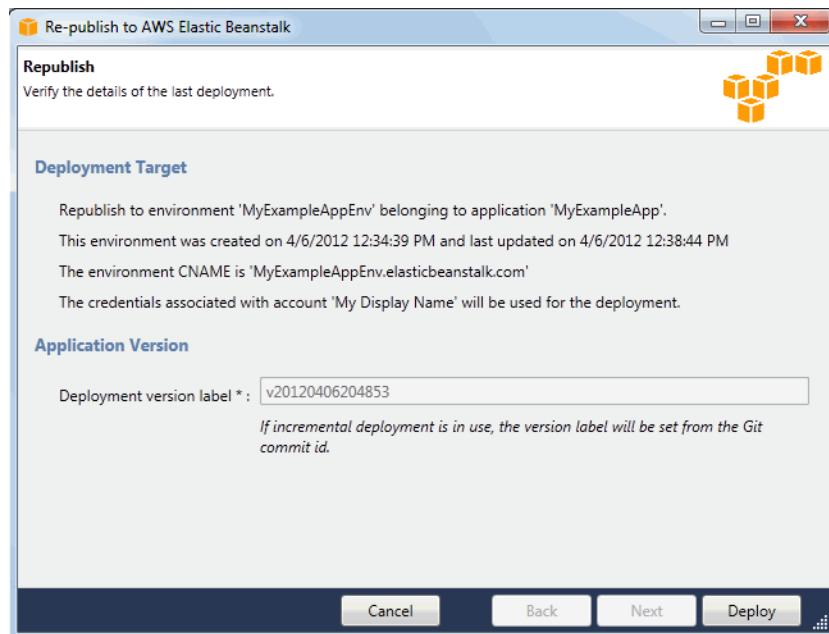
Lorsque vous résiliez votre environnement, le CNAME associé à l'environnement résilié devient disponible pour que tout le monde puisse l'utiliser.

## Déploiement dans votre environnement

Maintenant que vous avez testé votre application, il est facile de modifier et de redéployer de votre application et de consulter les résultats en quelques instants.

### Pour modifier et redéployer votre application web ASP.NET

1. Dans l'Explorateur de solutions, effectuez un clic droit sur votre application, puis cliquez sur Republish to Environment <nom de votre environnement> (Republier sur l'environnement <**nom de votre environnement**>). L'assistant Re-publish to AWS Elastic Beanstalk (Republication sur AWS Elastic Beanstalk) s'ouvre.



- Passez en revue les détails de votre déploiement et cliquez sur Deploy (Déploiement).

#### Note

Si vous souhaitez modifier certains de vos paramètres, vous pouvez cliquer sur Cancel (Annuler) et utiliser l'assistant Publish to AWS (Publier dans AWS) à la place. Pour obtenir des instructions, consultez [Créer un environnement Elastic Beanstalk \(p. 224\)](#).

Votre projet web ASP.NET mis à jour sera exporté comme un fichier Web Deploy avec la nouvelle étiquette de version, téléchargé sur Amazon S3 et inscrit en tant que nouvelle version de l'application avec Elastic Beanstalk. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement existant jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. Sous l'onglet env:<**nom de l'environnement**>, vous verrez l'état de votre environnement.

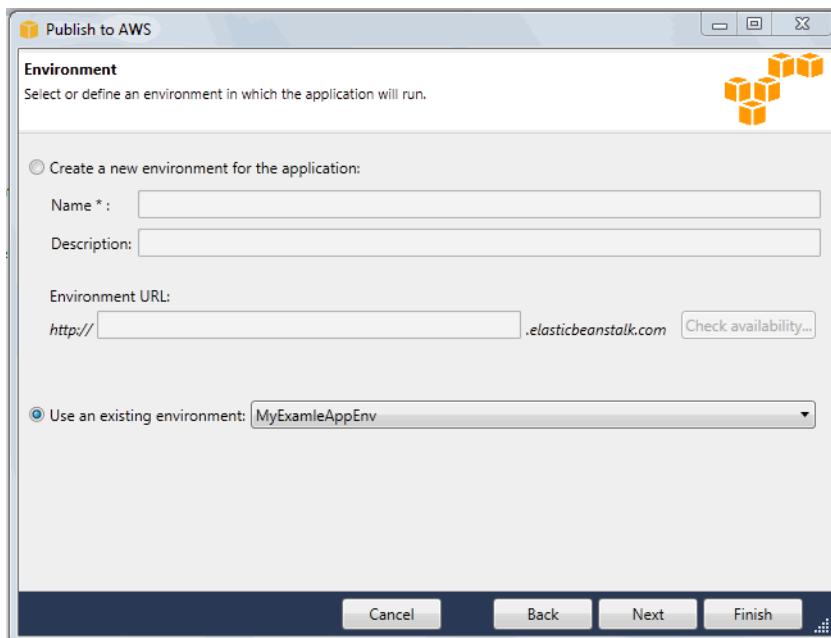
Vous pouvez également déployer une application existante dans un environnement existant si, par exemple, vous devez restaurer une version précédente de l'application.

Pour déployer une version de l'application dans un environnement existant

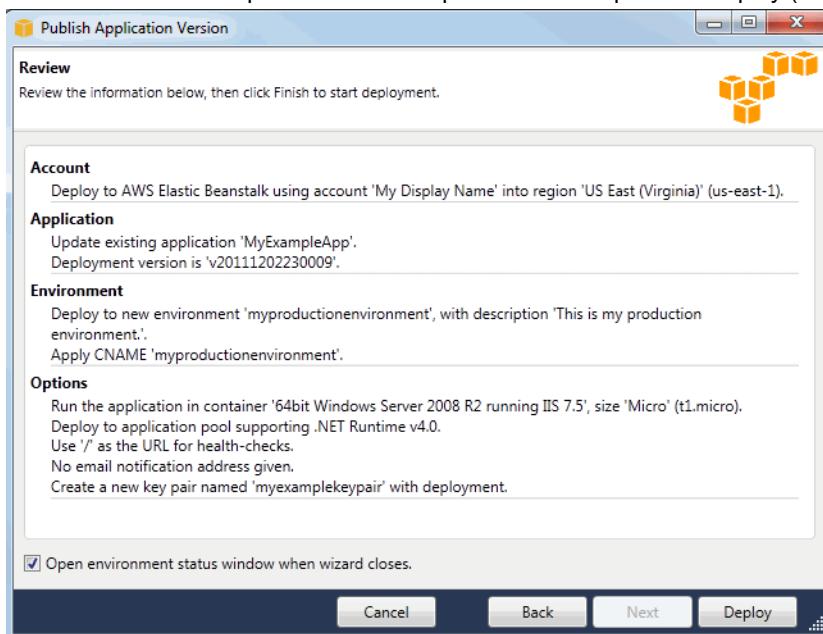
- Cliquez avec le bouton droit de la souris sur votre application Elastic Beanstalk en développant le nœud Elastic Beanstalk dans AWS Explorer (Explorateur AWS). Sélectionnez View Status (Afficher l'état).
- Dans l'onglet App: <**nom de l'application**>, cliquez sur Versions.

Events	Publish Version	Delete Version			
Versions	Version Label	Description	Created On	S3 Bucket	S3 Key
	v20111202232102		12/2/2011 3:42:59 PM	elasticbeanstalk-us-east-1-akiajcpa2ajx5z2igdq	MyExampleApp/AVS
	v2011120223009	This is my sample application.	12/2/2011 3:18:19 PM	elasticbeanstalk-us-east-1-akiajcpa2ajx5z2igdq	MyExampleApp/AVS

- Cliquez sur la version de l'application que vous souhaitez déployer et cliquez sur Publish Version (Publier la version).
- Dans l'assistant Publish Application Version (Publier la version de l'application), cliquez sur Next (Suivant).



5. Passez en revue les options de votre déploiement et cliquez sur Deploy (Déploiement).



Votre projet ASP.NET sera exporté comme un fichier Web Deploy et chargé sur Amazon S3. La fonctionnalité de déploiement Elastic Beanstalk surveille votre environnement jusqu'à ce qu'il devienne disponible avec le code nouvellement déployé. Sous l'onglet env:<nom de l'environnement>, vous verrez l'état de votre environnement.

## Gestion de vos environnements d'application Elastic Beanstalk

Avec AWS Toolkit for Visual Studio et la console de gestion AWS, vous pouvez modifier l'approvisionnement et la configuration des ressources AWS utilisées par les environnements de votre application. Pour plus d'informations sur la façon de gérer les environnements de votre application à l'aide

de la console de gestion AWS, consultez [Gestion des environnements \(p. 428\)](#). Cette section décrit les paramètres de service spécifiques que vous pouvez modifier dans AWS Toolkit for Visual Studio dans le cadre de la configuration d'environnement de votre application.

## Modification des paramètres de configuration de l'environnement

Lorsque vous déployez votre application, Elastic Beanstalk configure un certain nombre de services AWS de cloud computing. Vous pouvez contrôler la façon dont ces services individuels sont configurés à l'aide d'AWS Toolkit for Visual Studio.

Pour modifier les paramètres d'environnement d'une application

- Développez le nœud Elastic Beanstalk et le nœud de votre application. Ensuite, cliquez avec le bouton droit de la souris sur votre environnement Elastic Beanstalk dans AWS Explorer (Explorateur AWS). Sélectionnez View Status (Afficher l'état).

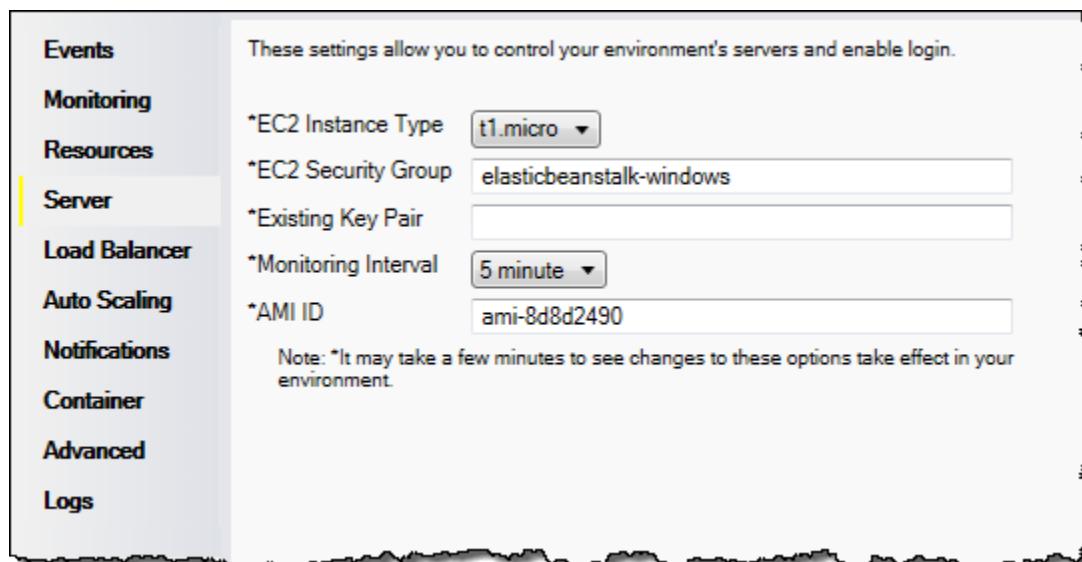
Vous pouvez à présent configurer des paramètres pour les éléments suivants :

- de bases de données
- Equilibrage de charge
- Auto Scaling
- Notifications
- Propriétés de l'environnement

## Configuration des instances de serveur EC2 à l'aide d'AWS Toolkit for Visual Studio

Amazon Elastic Compute Cloud (Amazon EC2) est un service web que vous utilisez pour lancer et gérer des instances de serveur dans les centres de données d'Amazon. Vous pouvez utiliser des instances de serveur Amazon EC2 à tout moment, aussi longtemps que vous le souhaitez et pour tout motif (dans le cadre d'une utilisation légale). Les instances sont disponibles dans différentes tailles et configurations. Pour de plus amples informations, veuillez consulter [Amazon EC2](#).

Vous pouvez modifier la configuration d'instance Amazon EC2 de l'environnement Elastic Beanstalk avec l'onglet Server (Serveur) à l'intérieur de votre onglet d'environnement de l'application dans AWS Toolkit for Visual Studio.



## Types d'instances Amazon EC2

Instance type (Type d'instance) affiche les types d'instance disponibles pour votre application Elastic Beanstalk. Changez le type d'instance pour sélectionner un serveur dont les caractéristiques (y compris la taille de la mémoire et la puissance d'UC) sont les mieux adaptées à votre application. Par exemple, les applications exécutant des opérations intensives et de longue durée peuvent nécessiter plus de puissance de calcul et de mémoire.

Pour de plus amples informations sur les types d'instance Amazon EC2 disponibles pour votre application Elastic Beanstalk, veuillez consulter [Types d'instances](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

## Groupes de sécurité Amazon EC2

Vous pouvez contrôler l'accès à votre application Elastic Beanstalk par le biais d'un groupe de sécurité Amazon EC2. Un groupe de sécurité définit les règles de pare-feu de vos instances. Ces règles déterminent le trafic réseau d'entrée (c'est à dire, entrant) doit être acheminé vers votre instance. Tout autre trafic d'entrée sera ignoré. Vous pouvez modifier les règles pour un groupe à la fois. Les nouvelles règles sont appliquées automatiquement pour toutes les instances en cours d'exécution et les instances lancées par la suite.

Vous pouvez définir vos groupes de sécurité Amazon EC2 à l'aide de la console de gestion AWS ou en utilisant l'AWS Toolkit for Visual Studio. Pour spécifier les groupes de sécurité Amazon EC2 qui contrôlent l'accès à votre application Elastic Beanstalk, saisissez les noms d'un ou de plusieurs groupes de sécurité Amazon EC2 (séparés par des virgules) dans la zone de texte EC2 Security Groups (Groupes de sécurité EC2).

### Note

Assurez-vous que le port 80 (HTTP) est accessible à partir de 0.0.0.0/0 comme source de la plage CIDR si vous souhaitez activer les vérifications de l'état pour votre application. Pour plus d'informations sur les vérifications de l'état, consultez [Vérifications de l'état \(p. 240\)](#).

Pour créer un groupe de sécurité à l'aide d'AWS Toolkit for Visual Studio

1. Dans Visual Studio, dans AWS Explorer (Explorateur AWS), développez le nœud Amazon EC2 puis double-cliquez sur Security Groups (Groupes de sécurité).
2. Cliquez sur Create Security Group (Créer un groupe de sécurité) et entrez un nom et une description pour votre groupe de sécurité.
3. Cliquez sur OK.

Pour de plus amples informations sur les groupes de sécurité Amazon EC2, veuillez consulter [Utilisation des groupes de sécurité](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

## Paires de clés Amazon EC2

Vous pouvez vous connecter en toute sécurité aux instances Amazon EC2 allouées pour votre application Elastic Beanstalk avec une paire de clés Amazon EC2.

### Important

Avant de pouvoir accéder à vos instances Amazon EC2 allouées par Elastic Beanstalk, vous devez créer une paire de clés Amazon EC2 et configurer vos instances Amazon EC2 allouées par Elastic Beanstalk pour utiliser la paire de clés Amazon EC2. Vous pouvez créer votre paire de clés à l'aide de l'assistant Publish to AWS (Publier dans AWS) à l'intérieur d'AWS Toolkit for Visual Studio lorsque vous déployez votre application sur Elastic Beanstalk. Si vous souhaitez créer des paires de clés supplémentaires à l'aide de Toolkit, procédez comme suit. Sinon, vous pouvez configurer vos paires de clés Amazon EC2 via la [console de gestion AWS](#). Pour obtenir

des instructions sur la création d'une paire de clés pour Amazon EC2, veuillez consulter le [Guide de démarrage Amazon Elastic Compute Cloud](#).

La zone de texte Existing Key Pair (Paire de clés existante) vous permet de spécifier le nom d'une paire de clés Amazon EC2 que vous pouvez utiliser pour vous connecter en toute sécurité aux instances Amazon EC2 exécutant votre application Elastic Beanstalk.

Pour spécifier le nom d'une paire de clés Amazon EC2

1. Développez le nœud Amazon EC2 et double-cliquez sur Key Pairs (Paires de clés).
2. Cliquez sur Create Key Pair (Créer une paire de clés) et saisissez le nom de la paire de clés.
3. Cliquez sur OK.

Pour de plus amples informations sur les paires de clés Amazon EC2, veuillez consulter [Utilisation des informations d'identification Amazon EC2](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.

Pour plus d'informations sur la connexion à des instances Amazon EC2, consultez [Affichage de la liste des instances de serveur et connexion à ces instances \(p. 246\)](#).

#### Intervalle de surveillance

Par défaut, seules les métriques de base d'Amazon Cloudwatch sont activées. Elles renvoient des données toutes les cinq minutes. Vous pouvez activer des métriques CloudWatch plus détaillées en une minute en sélectionnant 1 minute pour la Monitoring Interval (Intervalle de surveillance) dans la section Server (Serveur) de l'onglet Configuration correspondant à votre environnement dans AWS Toolkit for Eclipse.

#### Note

Des frais de service Amazon CloudWatch peuvent s'appliquer aux métriques d'intervalle d'une minute. Pour de plus amples informations, veuillez consulter [Amazon CloudWatch](#).

#### ID d'AMI personnalisé

Vous pouvez remplacer l'AMI par défaut utilisée pour vos instances Amazon EC2 par votre propre AMI personnalisée en saisissant l'identifiant de cette dernière dans la zone Custom AMI ID (ID d'AMI personnalisée) de la section Server (Serveur) de l'onglet Configuration correspondant à votre environnement dans AWS Toolkit for Eclipse.

#### Important

L'utilisation de votre propre image AMI est une tâche avancée qui doit être effectuée avec soin. Si vous avez besoin d'une AMI personnalisée, nous vous recommandons de démarrer par l'AMI Elastic Beanstalk par défaut, puis de la modifier. Pour être considérées saines par Elastic Beanstalk, les instances Amazon EC2 doivent respecter un ensemble de conditions, y compris disposer d'un gestionnaire hôte en cours d'exécution. Si ces conditions ne sont pas satisfaites, il se peut que votre environnement ne fonctionne pas correctement.

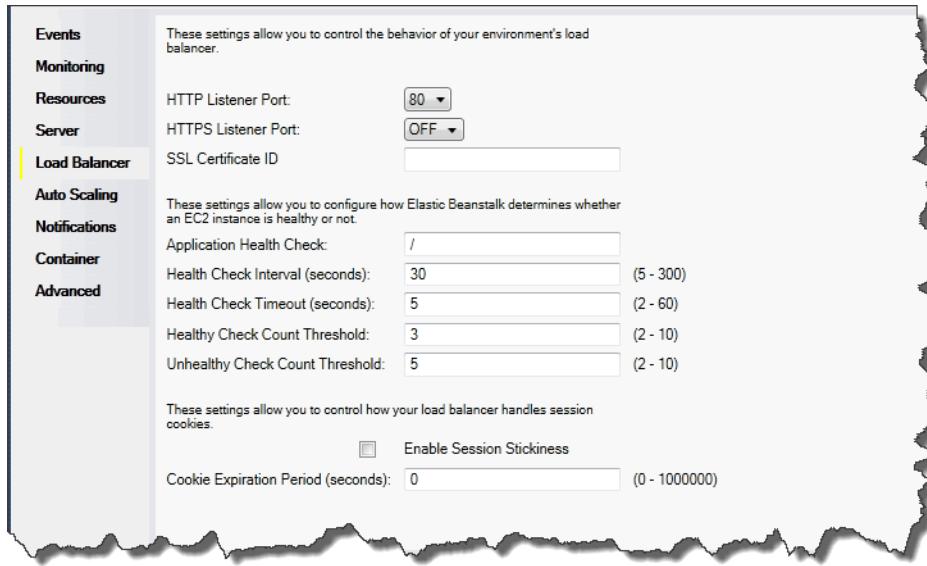
## Configuration d'Elastic Load Balancing à l'aide d'AWS Toolkit for Visual Studio

Elastic Load Balancing est un service d'Amazon Web Services qui vous aide à améliorer la disponibilité et l'évolutivité de votre application. Ce service vous permet de facilement répartir les charges d'application entre au moins deux instances Amazon EC2. Elastic Load Balancing active la disponibilité par la redondance et prend en charge l'augmentation du trafic pour votre application.

Elastic Load Balancing permet de répartir et d'équilibrer automatiquement le trafic entrant de votre application sur toutes les instances que vous exécutez. Il vous permet également d'ajouter aisément de nouvelles instances lorsque vous avez besoin d'augmenter la capacité de votre application.

Elastic Beanstalk fournit automatiquement Elastic Load Balancing lorsque vous déployez une application. Vous pouvez modifier la configuration d'instance Amazon EC2 de l'environnement Elastic Beanstalk

avec l'onglet Load Balancer (Équilibré de charge) à l'intérieur de l'onglet de l'environnement de votre application dans AWS Toolkit for Visual Studio.



Les sections suivantes décrivent les paramètres Elastic Load Balancing que vous pouvez configurer pour votre application.

## Ports

L'équilibré de charge alloué pour gérer les demandes pour votre application Elastic Beanstalk envoie des demandes aux instances Amazon EC2 qui exécutent votre application. L'équilibré de charge alloué peut écouter les demandes sur les ports HTTP et HTTPS, et les acheminer vers les instances Amazon EC2 dans votre application AWS Elastic Beanstalk. Par défaut, l'équilibré de charge gère les demandes sur le port HTTP. Au moins un des ports (HTTP ou HTTPS) doit être activé.



### Important

Assurez-vous que le port spécifié n'est pas verrouillé. S'il l'est, les utilisateurs ne pourront pas se connecter à votre application Elastic Beanstalk.

## Contrôle du port HTTP

Pour désactiver le port HTTP, sélectionnez OFF (désactivé) pour HTTP Listener Port (Port d'écoute HTTP). Pour activer le port HTTP, vous sélectionnez un port HTTP (par exemple, 80) dans la liste.

### Note

Pour accéder à votre environnement à l'aide d'un port autre que le port 80, par exemple le port 8080, vous pouvez ajouter un écouteur à l'équilibré de charge existant et configurer le nouvel écouteur de sorte qu'il écoute sur ce port.

Par exemple, en utilisant la [AWS CLI for Classic load balancers](#) (CLI pour les équilibréurs de charge Classic Load Balancer), tapez la commande suivante en remplaçant **LOAD\_BALANCER\_NAME** par le nom de votre équilibré de charge pour Elastic Beanstalk.

```
aws elb create-load-balancer-listeners --load-balancer-name LOAD_BALANCER_NAME
--listeners "Protocol=HTTP, LoadBalancerPort=8080, InstanceProtocol=HTTP,
InstancePort=80"
```

Par exemple, en utilisant la [AWS CLI for Application Load Balancers](#) (CLI pour les équilibreurs de charge Application Load Balancer), tapez la commande suivante en remplaçant **LOAD\_BALANCER\_ARN** par l'ARN de votre équilibrEUR de charge pour Elastic Beanstalk.

```
aws elbv2 create-listener --load-balancer-arn LOAD_BALANCER_ARN --protocol HTTP --
port 8080
```

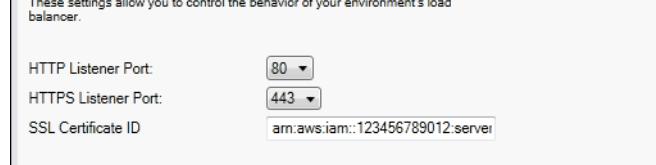
Si vous souhaitez que Elastic Beanstalk surveille votre environnement, ne supprimez pas l'écouteur sur le port 80.

## Contrôle du port HTTPS

Elastic Load Balancing prend en charge le protocole HTTPS/TLS pour activer le chiffrement du trafic pour les connexions client à l'équilibrEUR de charge. Les connexions à partir de l'équilibrEUR de charge aux instances EC2 utilisent le chiffrement en clair. Par défaut, le port HTTPS est désactivé.

### Pour activer le port HTTPS

- Créez un nouveau certificat à l'aide d'AWS Certificate Manager (ACM) ou téléchargez un certificat et une clé dans AWS Identity and Access Management (IAM). Pour plus d'informations sur une demande de certificat ACM, consultez [Request a Certificate](#) (Demande de certificat) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur d'AWS Certificate Manager). Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez [Importing Certificates](#) (Importation de certificats) dans le AWS Certificate Manager User Guide (Guide de l'utilisateur d'AWS Certificate Manager). Si ACM n'est pas [disponible dans votre région](#), utilisez AWS Identity and Access Management (IAM) pour télécharger un certificat tiers. Les services ACM et IAM stockeront le certificat et fourniront un Amazon Resource Name (ARN) pour le certificat SSL. Pour de plus amples informations sur la création et le chargement des certificats dans IAM, veuillez consulter [Utilisation des certificats de serveur](#) dans le Guide de l'utilisateur IAM.
- Spécifiez le port HTTPS en sélectionnant un port pour HTTPS Listener Port (Port d'écoute HTTPS).



- Pour SSL Certificate ID (ID du certificat SSL), saisissez l'ARN (Amazon Resources Name) de votre certificat SSL. Par exemple, `arn:aws:iam::123456789012:server-certificate/abc/certs/build` ou `arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`. Utilisez le certificat SSL que vous avez créé ou chargé à l'étape 1.

Pour désactiver le port HTTPS, sélectionnez OFF (désactivé) pour HTTPS Listener Port (Port d'écoute HTTPS).

## Vérifications de l'état

La définition de la vérification de l'état inclut une URL à interroger pour l'intégrité de l'instance. Par défaut, Elastic Beanstalk utilise TCP:80 pour les conteneurs non hérités et HTTP:80 pour les conteneurs hérités. Vous pouvez remplacer l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, `/myapp/default.aspx`) en entrant celle-ci dans la zone URL de vérification de

l'état de l'application. Si vous remplacez l'URL par défaut, Elastic Beanstalk utilise HTTP pour interroger la ressource. Pour vérifier si vous utilisez un type de conteneur hérité, consultez [the section called “Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?” \(p. 507\)](#)

Vous pouvez contrôler les paramètres de vérification de l'état via la section Vérification de l'état de l'instance EC2 du panneau Équilibrage de charge.

These settings allow you to configure how Elastic Beanstalk determines whether an EC2 instance is healthy or not.

Application Health Check:	/	
Health Check Interval (seconds):	30	(5 - 300)
Health Check Timeout (seconds):	5	(2 - 60)
Healthy Check Count Threshold:	3	(2 - 10)
Unhealthy Check Count Threshold:	5	(2 - 10)

La définition de la vérification de l'état inclut une URL à interroger pour l'intégrité de l'instance. Remplacez l'URL par défaut par une URL qui correspond à une ressource existante dans votre application (par exemple, /myapp/index.jsp) en entrant celle-ci dans la zone URL de vérification de l'état de l'application.

La liste suivante décrit les paramètres de vérification de l'état que vous pouvez définir pour votre application.

- Pour Intervalle de vérification de l'état (secondes), entrez le nombre de secondes d'attente pour Elastic Load Balancing entre les vérifications de l'état pour les instances Amazon EC2 de votre application.
- Pour Délai de vérification de l'état (secondes), spécifiez le nombre de secondes d'attente d'une réponse pour Elastic Load Balancing avant de considérer que l'instance ne répond pas.
- Pour Seuil du nombre de vérifications de l'état saines et Seuil du nombre de vérifications de l'état non saines, spécifiez le nombre d'analyses d'URL consécutives réussies et non réussies avant qu'Elastic Load Balancing ne modifie l'état de l'instance. Par exemple, si vous spécifiez 5 pour Seuil du nombre de vérifications de l'état non saines, l'URL doit renvoyer un message d'erreur ou une expiration du délai cinq fois de suite avant qu'Elastic Load Balancing considère que la vérification de l'état est un échec.

## Sessions

Par défaut, un équilibrEUR de charge achemine chaque demande de façon indépendante à l'instance de serveur ayant la plus petite charge. Par comparaison, une session permanente lie la session d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la session soient envoyées à la même instance de serveur.

Elastic Beanstalk utilise des cookies HTTP générés par l'équilibrEUR de charge lorsque des sessions permanentes sont activées pour une application. L'équilibrEUR de charge utilise un cookie spécial généré par l'équilibrEUR de charge pour suivre l'instance d'application pour chaque demande. Lorsque l'équilibrEUR de charge reçoit une demande, il vérifie d'abord si ce cookie est présent dans la demande. Si tel est le cas, la demande est envoyée à l'instance d'application spécifiée dans le cookie. S'il n'y a pas de cookie, l'équilibrEUR de charge choisit une instance d'application à partir de l'algorithme d'équilibrage de charge existant. Un cookie est inséré dans la réponse pour lier les demandes suivantes provenant du même utilisateur à cette instance d'application. La configuration de la stratégie définit l'expiration d'un cookie, ce qui établit la durée de validité de chaque cookie.

Vous pouvez utiliser la section Sessions dans l'onglet ÉquilibrEUR de charge afin d'indiquer si l'équilibrEUR de charge pour votre application autorise ou non la permanence de session.

These settings allow you to control how your load balancer handles session cookies.

<input checked="" type="checkbox"/> Enable Session Stickiness
Cookie Expiration Period (seconds): 0 (0 - 1000000)

Pour de plus amples informations Elastic Load Balancing, veuillez consulter le [Guide du développeur Elastic Load Balancing](#).

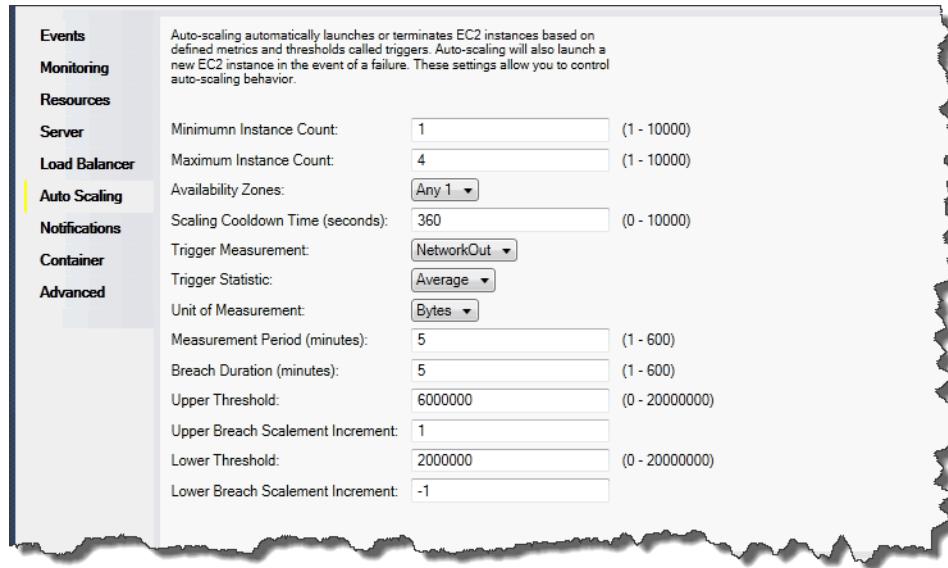
## Configuration d'Auto Scaling à l'aide d'AWS Toolkit for Visual Studio

Amazon EC2 Auto Scaling est un service web Amazon conçu pour lancer ou résilier automatiquement les instances Amazon EC2 en fonction de déclencheurs définis par l'utilisateur. Les utilisateurs peuvent configurer des groupes Auto Scaling et y associer des déclencheurs afin de mettre à l'échelle automatiquement les ressources de calcul selon des métriques comme l'utilisation de la bande passante ou l'utilisation de l'UC. Amazon EC2 Auto Scaling fonctionne avec Amazon CloudWatch afin de récupérer des métriques pour les instances de serveur exécutant votre application.

Amazon EC2 Auto Scaling vous permet de récupérer un groupe d'instances Amazon EC2 et de définir différents paramètres pour que ce groupe augmente ou diminue automatiquement en nombre. Amazon EC2 Auto Scaling peut ajouter ou supprimer des instances Amazon EC2 de ce groupe pour vous aider à gérer facilement l'évolution du trafic vers votre application.

De plus, Amazon EC2 Auto Scaling surveille l'état de chaque instance Amazon EC2 qu'il lance. Si une instance est résiliée de façon inattendue, Amazon EC2 Auto Scaling détecte cette résiliation et lance une instance de remplacement. Cette fonctionnalité vous permet de maintenir automatiquement un nombre fixe et souhaité d'instances Amazon EC2.

Elastic Beanstalk met en service Amazon EC2 Auto Scaling pour votre application. Vous pouvez modifier la configuration d'instance Amazon EC2 de l'environnement Elastic Beanstalk avec l'onglet Auto Scaling à l'intérieur de l'onglet de l'environnement de votre application dans AWS Toolkit for Visual Studio.



La section suivante explique comment configurer les paramètres Auto Scaling pour votre application.

### Lancement de la configuration

Vous pouvez modifier la configuration de lancement pour contrôler la façon dont votre application Elastic Beanstalk alloue des ressources Amazon EC2 Auto Scaling.

Les zones Minimum Instance Count (Nombre minimum d'instances) et Maximum Instance Count (Nombre maximum d'instances) vous permettent de spécifier les tailles maximale et minimale du groupe Auto Scaling utilisé par votre application Elastic Beanstalk.

Auto-scaling automatically launches or terminates EC2 instances based on defined metrics and thresholds called triggers. Auto-scaling will also launch a new EC2 instance in the event of a failure. These settings allow you to control auto-scaling behavior.

Minimum Instance Count:	<input type="text" value="1"/> (1 - 10000)
Maximum Instance Count:	<input type="text" value="4"/> (1 - 10000)
Availability Zones:	<input type="text" value="Any"/>
Scaling Cooldown Time (seconds):	<input type="text" value="360"/> (0 - 10000)

#### Note

Pour maintenir un nombre fixe d'instances Amazon EC2, attribuez la même valeur aux champs Nombre minimum d'instances et Nombre maximum d'instances.

La zone Zones de disponibilité vous permet de spécifier le nombre de zones de disponibilité dans lesquelles vous souhaitez que se trouvent vos instances Amazon EC2. Il est important de définir ce nombre si vous souhaitez créer des applications à tolérance de panne. Si une zone de disponibilité est défaillante, l'exécution de vos instances se poursuivra dans vos autres zones de disponibilité.

#### Note

Il est actuellement impossible de spécifier la zone de disponibilité dans laquelle se situera votre instance.

#### Triggers

Un déclencheur est un mécanisme Amazon EC2 Auto Scaling que vous définissez pour indiquer au système quand vous souhaitez augmenter (monter en puissance) et quand vous souhaitez diminuer (diminuer en puissance) le nombre d'instances. Vous pouvez configurer des déclencheurs pour qu'ils soient activés en fonction de n'importe quelle métrique publiée dans Amazon CloudWatch, telle que l'utilisation de l'UC, et pour déterminer si les conditions que vous avez spécifiées sont réunies. Lorsque les seuils inférieurs ou supérieurs des conditions que vous avez spécifiées pour la métrique ont été dépassés pendant la période spécifiée, le déclencheur lance un processus de longue durée que nous appelons une activité de dimensionnement.

Vous pouvez définir un déclencheur de mise à l'échelle de votre application Elastic Beanstalk à l'aide d'AWS Toolkit for Visual Studio.

Trigger Measurement:	<input type="text" value="NetworkOut"/>
Trigger Statistic:	<input type="text" value="Average"/>
Unit of Measurement:	<input type="text" value="Bytes"/>
Measurement Period (minutes):	<input type="text" value="5"/> (1 - 600)
Breach Duration (minutes):	<input type="text" value="5"/> (1 - 600)
Upper Threshold:	<input type="text" value="6000000"/> (0 - 20000000)
Upper Breach Scalement Increment:	<input type="text" value="1"/>
Lower Threshold:	<input type="text" value="2000000"/> (0 - 20000000)
Lower Breach Scalement Increment:	<input type="text" value="-1"/>

Les déclencheurs Amazon EC2 Auto Scaling fonctionnent en consultant une métrique Amazon CloudWatch spécifique pour une instance. Les déclencheurs incluent l'utilisation de l'UC, le trafic réseau et l'activité du disque. Utilisez le paramètre Mesure du déclencheur pour sélectionner une métrique associée à votre déclencheur.

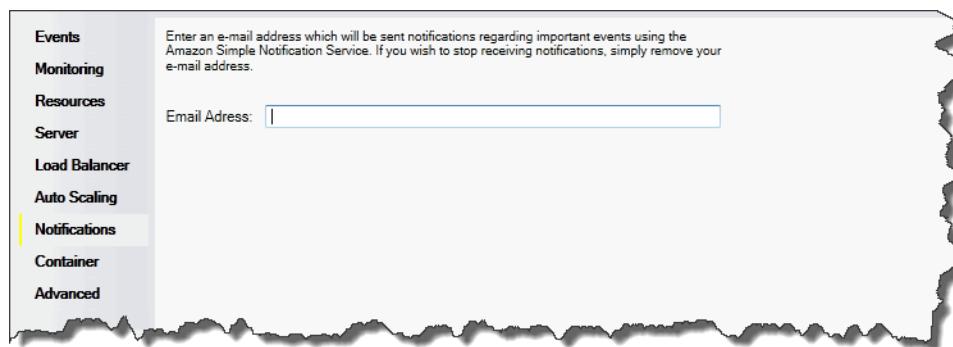
La liste suivante décrit les paramètres de déclencheur que vous pouvez configurer à l'aide de la console de gestion AWS.

- Vous pouvez spécifier les statistiques que le déclencheur devrait utiliser. Vous pouvez sélectionner Minimum, Maximum, Sum (Somme) ou Average (Moyenne) pour Statistique du déclencheur.
- Pour Unité de mesure, spécifiez l'unité de mesure du déclencheur.
- La valeur dans la zone Measurement Period (Durée de mesure) spécifie la fréquence à laquelle Amazon CloudWatch mesure les métriques pour votre déclencheur. La valeur Durée de la faille correspond à la durée pendant laquelle une métrique peut se situer au-delà de sa limite définie (telle que spécifiée dans Seuil supérieur et Seuil inférieur) avant l'activation du déclencheur.
- Pour Hausse du dimensionnement supérieur en cas de faille et Hausse du dimensionnement inférieur en cas de faille :, spécifiez le nombre d'instances Amazon EC2 à ajouter ou à supprimer lorsque vous effectuez une activité de dimensionnement.

Pour de plus amples informations sur Amazon EC2 Auto Scaling, veuillez consulter Amazon EC2 Auto Scaling dans la [documentation Amazon Elastic Compute Cloud](#).

## Configuration des notifications à l'aide d'AWS Toolkit for Visual Studio

Elastic Beanstalk utilise Amazon Simple Notification Service (Amazon SNS) pour vous informer des événements importants qui concernent votre application. Pour activer les notifications Amazon SNS, il suffit d'entrer votre adresse e-mail dans la zone Adresse e-mail. Pour désactiver ces notifications, supprimez votre adresse e-mail de la zone.



## Configuration de conteneurs .NET à l'aide d'AWS Toolkit for Visual Studio

Le panneau Container/.NET Options (Options du conteneur/.NET) vous permet d'ajuster le comportement de vos instances Amazon EC2 et d'activer ou de désactiver la rotation des journaux Amazon S3. Vous pouvez utiliser AWS Toolkit for Visual Studio pour configurer vos informations de conteneur.

### Note

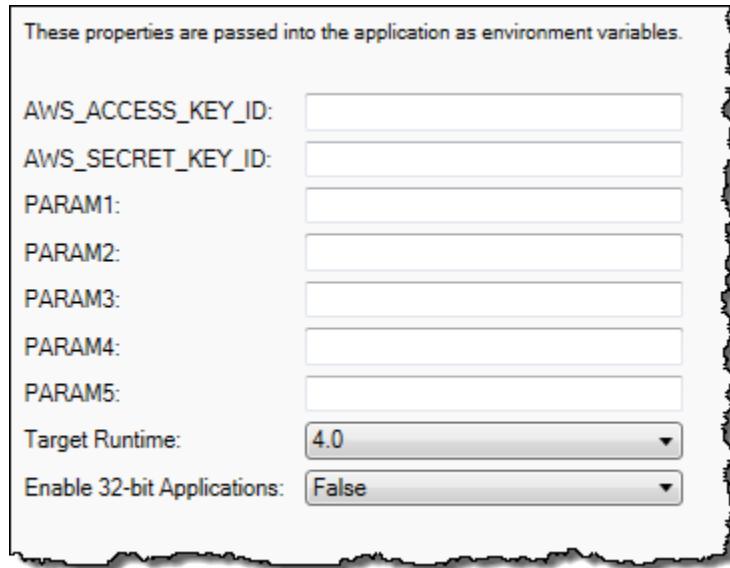
Pour modifier vos paramètres de configuration sans aucune interruption, échangez le CNAME pour vos environnements. Pour plus d'informations, consultez [Déploiements bleu/vert avec Elastic Beanstalk \(p. 486\)](#).

Si vous le souhaitez, vous pouvez étendre le nombre de paramètres. Pour plus d'informations sur l'extension des paramètres, consultez [Paramètres d'option \(p. 738\)](#).

Pour accéder au panneau d'options Container/.NET de votre application Elastic Beanstalk

1. Dans AWS Toolkit for Visual Studio, développez le nœud Elastic Beanstalk et le nœud de votre application.

2. Dans AWS Explorer (Explorateur AWS), double-cliquez sur votre environnement Elastic Beanstalk.
3. En bas du volet Présentation, cliquez sur l'onglet Configuration.
4. Sous Container (Conteneur), vous pouvez configurer les options de conteneur.

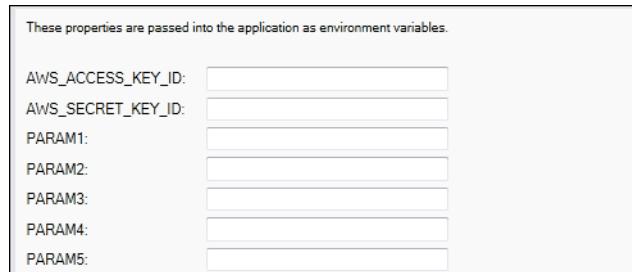


### Options du conteneur .NET

Vous pouvez choisir la version de .NET Framework pour votre application. Choisissez 2.0 ou 4.0 pour Runtime cible. Sélectionnez Activer les applications 32 bits si vous souhaitez activer des applications 32 bits.

### Paramètres de l'application

La section Paramètres de l'application vous permet de spécifier des variables d'environnement que vous pouvez lire depuis votre code d'application.



## Gestion de comptes

Si vous voulez configurer différents comptes AWS pour exécuter différentes tâches, telles que les tests, la gestion intermédiaire et la production, vous pouvez ajouter, modifier et supprimer des comptes à l'aide d'AWS Toolkit for Visual Studio.

### Pour gérer plusieurs comptes

1. Dans Visual Studio, dans le menu View (Afficher), cliquez sur AWS Explorer (Explorateur AWS).
2. À côté de la liste Account (Compte), cliquez sur le bouton Add Account (Ajouter un compte).



La boîte de dialogue Add Account (Ajouter un compte) apparaît.



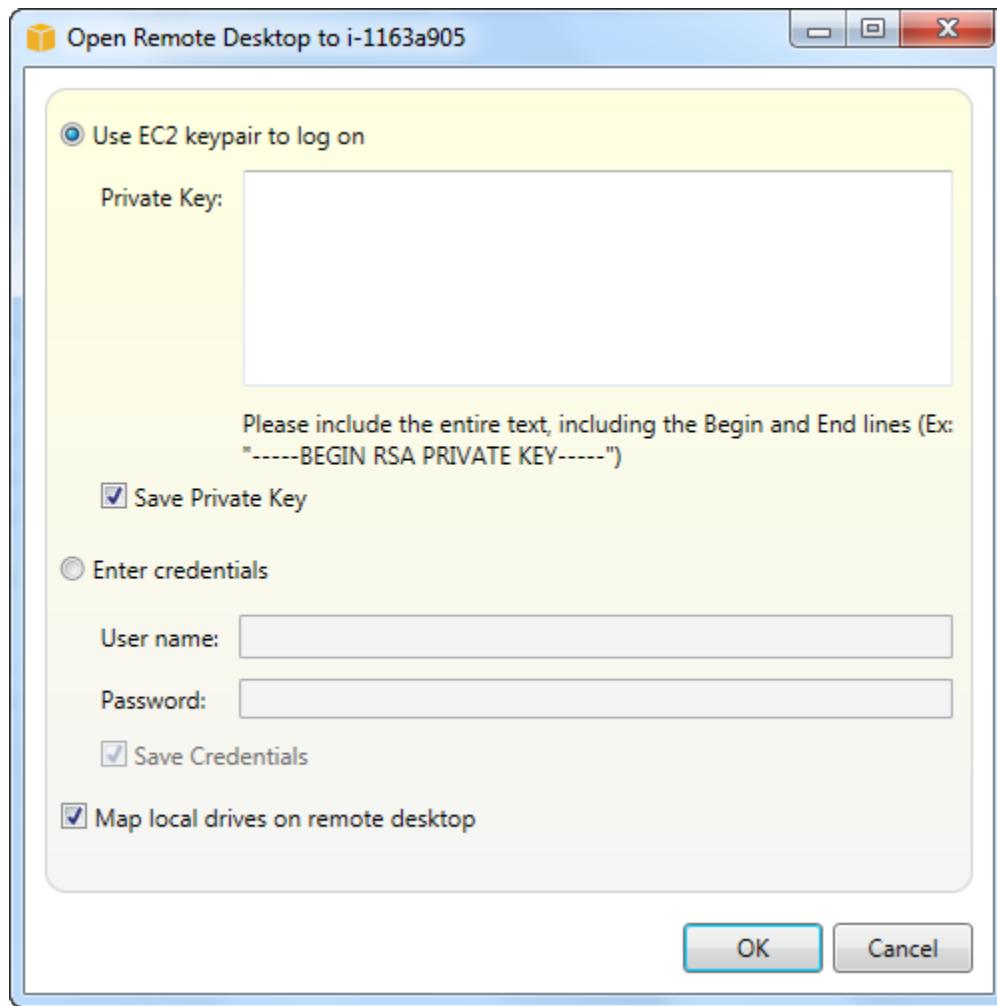
3. Renseignez les informations requises.
4. Vos informations de compte s'affichent désormais sous l'onglet AWS Explorer (Explorateur AWS). Lorsque vous publiez sur Elastic Beanstalk, vous pouvez sélectionner le compte que vous souhaitez utiliser.

## Affichage de la liste des instances de serveur et connexion à ces instances

Vous pouvez afficher une liste d'instances Amazon EC2 exécutant votre environnement d'applications Elastic Beanstalk via AWS Toolkit for Visual Studio ou à partir de la console de gestion AWS. Vous pouvez vous connecter à ces instances à l'aide d'une connexion Bureau à distance. Pour plus d'informations sur la façon d'obtenir la liste de vos instances de serveur et de vous y connecter via la console de gestion AWS, consultez [Affichage de la liste des instances de serveur et connexion à ces instances \(p. 881\)](#). La section suivante vous explique comment afficher vos instances de serveur et vous y connecter via AWS Toolkit for Visual Studio.

Pour afficher les instances Amazon EC2 d'un environnement et s'y connecter

1. Dans Visual Studio, dans AWS Explorer (Explorateur AWS), développez le nœud Amazon EC2 et double-cliquez sur Instances.
2. Effectuez un clic droit sur l'ID d'instance pour l'instance Amazon EC2 en cours d'exécution dans l'équilibrage de charge de l'application dans la colonne Instance (Instance) et sélectionnez Open remote desktop (Ouvrir le bureau à distance) dans le menu contextuel.



3. Sélectionnez Use EC2 keypair to log on (Utiliser la paire de clés EC2 pour se connecter) et collez le contenu de votre fichier de clé privée que vous avez utilisé pour déployer votre application dans la zone Private key (Clé privée). Sinon, entrez vos nom utilisateur et mot de passe dans les zones de texte User name (Nom d'utilisateur) and Password (Mot de passe).

Note

Si la paire de clés est stockée à l'intérieur du Toolkit, la zone de texte ne s'affiche pas.

4. Cliquez sur OK.

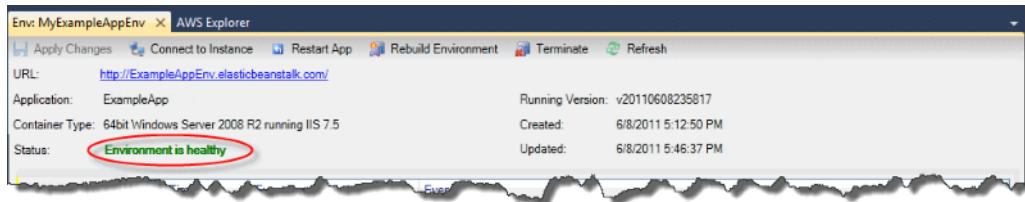
## Surveillance de l'intégrité d'une application

Lorsque vous exécutez un site web de production, il est important de savoir que votre application est disponible et répond aux demandes. Pour aider à la surveillance de la réactivité de votre application, Elastic Beanstalk offre des fonctions où vous pouvez surveiller des statistiques de votre application et créer des alertes qui se déclenchent quand des seuils sont dépassés.

Pour de plus amples informations sur la surveillance de l'état fournie par Elastic Beanstalk, veuillez consulter [Création de rapports d'intégrité de base \(p. 834\)](#).

Vous pouvez accéder aux informations opérationnelles sur votre application en utilisant soit l'AWS Toolkit for Visual Studio, soit la console de gestion AWS.

La boîte à outils affiche le statut de votre environnement et l'état de votre application dans le champ Status (Statut).



Pour surveiller l'intégrité de l'application

1. Dans AWS Toolkit for Visual Studio, dans AWS Explorer (Explorateur AWS), développez le nœud Elastic Beanstalk, puis le nœud de votre application.
2. Cliquez avec le bouton droit sur votre environnement Elastic Beanstalk, puis cliquez sur View Status (Afficher le statut).
3. Dans l'onglet de l'environnement de votre application, cliquez sur Monitoring (Surveillance).

Le panneau Monitoring (Surveillance) comprend un ensemble de graphiques illustrant l'utilisation des ressources pour votre environnement d'application spécifique.



#### Note

Par défaut, la plage de temps est définie sur la dernière heure. Pour modifier ce paramètre, dans la liste Time Range (Plage horaire), cliquez sur une plage de temps différente.

Vous pouvez utiliser AWS Toolkit for Visual Studio ou la console de gestion AWS pour afficher les événements associés à votre application.

Pour afficher des événements d'application

1. Dans AWS Toolkit for Visual Studio, sous AWS Explorer (Explorateur AWS), développez le nœud Elastic Beanstalk et le nœud de votre application.
2. Cliquez avec le bouton droit sur votre environnement Elastic Beanstalk dans AWS Explorer (Explorateur AWS), puis cliquez sur View Status (Afficher le statut).
3. Dans l'onglet de l'environnement de votre application, cliquez sur Events (Événements).

Events	Filter:			
	Event Time	Event Type	Version Label	Event Details
Monitoring	12/2/2011 3:43:19 PM	INFO	v20111202232102	Environment update completed successfully.
Resources	12/2/2011 3:43:19 PM	INFO	v20111202232102	New application version was deployed to running EC2 instances.
Server	12/2/2011 3:43:06 PM	INFO	v20111202232102	Waiting for 2 seconds while EC2 instances download the updated application version.
Load Balancer	12/2/2011 3:43:04 PM	INFO	v20111202232102	Deploying version v20111202232102 to 1 instance(s).
	12/2/2011 3:42:59 PM	INFO	v2011120223009	Environment update is starting.
	12/2/2011 3:30:38 PM	INFO	v2011120223009	Environment health has transitioned from RED to GREEN
Auto Scaling	12/2/2011 3:29:37 PM	WARN	v2011120223009	Environment health has been set to RED
Notifications	12/2/2011 3:28:35 PM	INFO	v2011120223009	Launched environment: MyExampleAppEnv. However, there were issues during launch. See event log for details.
Container	12/2/2011 3:19:13 PM	INFO	v2011120223009	Exceeded maximum amount of time to wait for the application to become available. Setting environment Ready.
Advanced	12/2/2011 3:18:50 PM	INFO	v2011120223009	Added EC2 instance i-93d6e6f0 to Auto Scaling Group awseb-MyExampleAppEnv-5y33OGVvOm.
	12/2/2011 3:18:47 PM	INFO	v2011120223009	An EC2 instance has been launched. Waiting for it to be added to Auto Scaling...
	12/2/2011 3:18:34 PM	INFO	v2011120223009	Waiting for an EC2 instance to be launched...
	12/2/2011 3:18:33 PM	INFO	v2011120223009	Adding Auto Scaling Group 'awseb-MyExampleAppEnv-5y33OGVvOm' to your environment.
	12/2/2011 3:18:33 PM	INFO	v2011120223009	Added URLCheck healthcheck for 'http://MyExampleAppEnv.elasticbeanstalk.com:80/'
	12/2/2011 3:18:31 PM	INFO	v2011120223009	Created Auto Scaling trigger named: awseb-MyExampleAppEnv-5y33OGVvOm.
	12/2/2011 3:18:30 PM	INFO	v2011120223009	Created Auto Scaling group named: awseb-MyExampleAppEnv-5y33OGVvOm.
	12/2/2011 3:18:30 PM	INFO	v2011120223009	Created Auto Scaling launch configuration named: awseb-MyExampleAppEnv-JDwosdTJA.
	12/2/2011 3:18:29 PM	INFO	v2011120223009	Created load balancer named: awseb-MyExampleAppEnv.
	12/2/2011 3:18:28 PM	INFO	v2011120223009	Created security group named: elasticbeanstalk-windows.
	12/2/2011 3:18:28 PM	INFO	v2011120223009	Using elasticbeanstalk-us-east-1-049020475370 as Amazon S3 storage bucket for environment data.

## Déploiement d'applications Elastic Beanstalk dans .NET à l'aide de l'outil de déploiement

AWS Toolkit for Visual Studio inclut un outil de déploiement. Cet outil de ligne de commande offre les mêmes fonctionnalités que l'assistant de déploiement d'AWS Toolkit. Vous pouvez utiliser l'outil de déploiement dans votre pipeline de build ou dans d'autres scripts afin d'automatiser les déploiements dans Elastic Beanstalk.

L'outil de déploiement prend en charge les déploiements initiaux et les redéploiements. Si vous avez déjà déployé votre application via l'outil de déploiement, vous pouvez le redéployer via l'assistant de déploiement dans Visual Studio. De même, si vous avez effectué un déploiement via l'assistant, vous pouvez procéder à un redéploiement via l'outil de déploiement.

### Note

L'outil de déploiement n'applique pas de [valeurs recommandées \(p. 660\)](#) pour les options de configuration, contrairement à la console ou à l'interface de ligne de commande (CLI) EB. Utilisez des [fichiers de configuration \(p. 737\)](#) pour vous assurer que tous les paramètres dont vous avez besoin sont configurés lorsque vous lancez votre environnement.

Ce chapitre vous explique comment déployer un exemple d'application .NET dans Elastic Beanstalk via l'outil de déploiement, puis comment redéployer l'application par le biais d'un déploiement incrémental. Pour obtenir des informations plus détaillées sur l'outil de déploiement, y compris sur les options des paramètres, veuillez consulter [Outil de déploiement](#).

### Prerequisites

Pour utiliser l'outil de déploiement, vous devez installer AWS Toolkit for Visual Studio. Pour de plus amples informations sur les conditions préalables et les instructions d'installation, veuillez consulter [AWS Toolkit for Microsoft Visual Studio](#).

L'outil de déploiement est généralement installé dans l'un des répertoires suivants sous Windows :

32 bits	64 bits
C:\Program Files\AWS Tools\Deployment Tool\awsdeploy.exe	C:\Program Files (x86)\AWS Tools\Deployment Tool\awsdeploy.exe

## Déploiement sur Elastic Beanstalk

Pour déployer l'exemple d'application dans Elastic Beanstalk via l'outil de déploiement, vous devez commencer par modifier le fichier de configuration `ElasticBeanstalkDeploymentSample.txt`, qui est inclus dans le répertoire `Samples`. Ce fichier de configuration contient les informations nécessaires pour déployer votre application, telles que le nom de l'application, la version de l'application, le nom de l'environnement et vos informations d'identification pour accéder à AWS. Une fois que vous avez modifié le fichier de configuration, vous devez utiliser la ligne de commande pour déployer l'exemple d'application. Votre fichier de déploiement web est chargé dans Amazon S3 et enregistré en tant que nouvelle version de l'application avec Elastic Beanstalk. Quelques minutes sont nécessaires au déploiement de votre application. Une fois que l'environnement est sain, l'outil de déploiement génère une URL pour l'application en cours d'exécution.

Pour déployer une application .NET sur Elastic Beanstalk

1. À partir du sous-répertoire `Samples` dans lequel l'outil de déploiement est installé, ouvrez `ElasticBeanstalkDeploymentSample.txt` et saisissez votre clé d'accès AWS et la clé secrète AWS, comme dans l'exemple suivant.

```
### AWS Access Key and Secret Key used to create and deploy the application instance
AWSAccessKey = AKIAIOSFODNN7EXAMPLE
AWSSecretKey = wJalrXUtnFEMI/K7MDENG/bPxRfICYEXAMPLEKEY
```

### Note

Pour accéder à l'API, vous avez besoin d'un ID de clé d'accès et d'une clé d'accès secrète. Utilisez les clés d'accès utilisateur IAM au lieu des clés d'accès utilisateur racine du Compte AWS . Pour de plus amples informations sur la création de clés d'accès, veuillez consulter [Gestion des clés d'accès pour les utilisateurs IAM](#) dans le Guide de l'utilisateur IAM.

2. Dans l'invite de commande, saisissez la chaîne suivante :

```
C:\Program Files (x86)\AWS Tools\Deployment Tool>awsdeploy.exe /w Samples
\ElasticBeanstalkDeploymentSample.txt
```

Quelques minutes sont nécessaires au déploiement de votre application. Si le déploiement réussit, le message s'affiche, `Application deployment completed; environment health is Green.`

### Note

Si le message d'erreur suivant s'affiche, cela signifie que le CNAME existe déjà.

```
[Error]: Deployment to AWS Elastic Beanstalk failed with exception: DNS name
(MyAppEnv.elasticbeanstalk.com) is not available.
```

Vous devez modifier `Environment.CNAME` dans `ElasticBeanstalkDeploymentSample.txt`, car le CNAME doit être unique.

3. Dans votre navigateur web, accédez à l'URL de votre application en cours d'exécution. L'URL s'affiche sous la forme <CNAME.elasticbeanstalk.com> (par exemple, `MyAppEnv.elasticbeanstalk.com`).

## Migration de votre application .NET sur site vers Elastic Beanstalk

Si vous envisagez de migrer votre application .NET à partir de serveurs sur site vers Amazon Web Services (AWS), l'assistant de migration .NET pour AWS Elastic Beanstalk peut vous être utile. L'assistant est

un utilitaire PowerShell interactif qui migre une application .NET à partir de Windows Server avec IIS s'exécutant sur site vers AWS Elastic Beanstalk. L'assistant peut migrer l'intégralité d'un site web vers Elastic Beanstalk avec un minimum de modifications ou sans aucune modification.

Pour plus d'informations sur l'assistant de migration .NET pour AWS Elastic Beanstalk et sur la manière de le télécharger, consultez le référentiel <https://github.com/awslabs/windows-web-app-migration-assistant> sur GitHub.

Si votre application inclut des bases de données Microsoft SQL Server, la documentation de l'assistant sur GitHub comprend plusieurs options permettant de les migrer.

## Resources

Il existe plusieurs endroits auxquels vous pouvez accéder pour obtenir une aide supplémentaire lors du développement de vos applications .NET :

Ressource	Description
<a href="#">.Forum de développement .NET</a>	Posez vos questions et obtenez des commentaires.
<a href="#">.Centre pour développeurs .NET</a>	Guichet unique pour l'exemple de code, la documentation, les outils et les ressources supplémentaires.
<a href="#">Documentation sur le kit de développement logiciel (SDK) AWS pour .NET</a>	Lisez des informations concernant la configuration du SDK et l'exécution d'exemples de code, les fonctionnalités du SDK et des informations détaillées sur les opérations de l'API pour le SDK.

## Déploiement d'applications Node.js sur Elastic Beanstalk

### Rubriques

- [Démarrer avec Node.js sur Elastic Beanstalk \(p. 252\)](#)
- [Configuration de votre environnement de développement Node.js \(p. 253\)](#)
- [Utilisation de la plateforme Elastic Beanstalk Node.js \(p. 255\)](#)
- [Déploiement d'une application Express sur Elastic Beanstalk \(p. 265\)](#)
- [Déploiement d'une application Express avec une mise en cluster dans Elastic Beanstalk \(p. 269\)](#)
- [Déploiement d'une application Node.js avec DynamoDB vers Elastic Beanstalk \(p. 278\)](#)
- [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Node.js \(p. 287\)](#)
- [Resources \(p. 290\)](#)

AWS Elastic Beanstalk pour Node.js permet de déployer, de gérer et de dimensionner facilement vos applications web Node.js à l'aide d'Amazon Web Services. Elastic Beanstalk pour Node.js est disponible pour toute personne développant ou hébergeant une application web à l'aide de Node.js. Ce chapitre fournit, étape par étape, des instructions permettant de déployer votre application web Node.js dans Elastic Beanstalk à l'aide de la console de gestion Elastic Beanstalk et décrit des procédures permettant de réaliser des tâches courantes telles que l'intégration de bases de données et leur utilisation avec le cadre Express.

Après avoir déployé votre application Elastic Beanstalk, vous pouvez continuer à utiliser la CLI EB pour gérer votre application et votre environnement, ou vous pouvez utiliser la console Elastic Beanstalk, AWS CLI ou les API.

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

## Démarrer avec Node.js sur Elastic Beanstalk

Pour commencer à utiliser des applications Node.js sur AWS Elastic Beanstalk, il vous suffit de charger le [bundle source \(p. 415\)](#) d'une application en tant que première version de l'application et de la déployer dans un environnement. Lorsque vous créez un environnement, Elastic Beanstalk alloue toutes les ressources AWS nécessaires pour exécuter une application web hautement évolutive.

### Lancement d'un environnement avec un exemple d'application Node.js

Elastic Beanstalk fournit des exemples d'applications à page unique pour chaque plateforme, ainsi que des exemples plus complexes qui illustrent l'utilisation de ressources AWS supplémentaires, comme Amazon RDS, des fonctions propres aux différents langages ou plateformes, et des API.

#### Samples

Type d'environnement	Solution groupée source	Description	
Serveur web	<a href="#">nodejs.zip</a>	Application avec page unique.  Utilisez la procédure de <a href="#">Création d'un exemple d'application (p. 3)</a> pour lancer cet exemple.	
Serveur web avec Amazon RDS	<a href="#">nodejs-express-hiking-v1.zip</a>	Application de suivi pour la randonnée, qui utilise l'infrastructure Express et une base de données RDS.  <a href="#">Didacticiel (p. 265)</a>	
Serveur web avec Amazon ElastiCache	<a href="#">nodejs-example-express-elasticsearch.zip</a>	Application web Express qui utilise Amazon ElastiCache pour la mise en cluster. Le clustering permet d'améliorer la haute disponibilité, les performances et la sécurité de votre application web.  <a href="#">Didacticiel (p. 269)</a>	
Serveur web avec DynamoDB, Amazon SNS et Amazon SQS	<a href="#">eb-node-express-sample-v1.0.zip</a>  <a href="#">Clonez le référentiel sur GitHub.com</a>	Site web Express qui collecte les informations de contact des utilisateurs pour la nouvelle campagne de marketing d'une entreprise. Utilise le kit SDK AWS pour JavaScript dans Node.js pour écrire des entrées dans une table DynamoDB, et les fichiers de configuration Elastic Beanstalk pour créer des ressources dans DynamoDB, Amazon SNS et Amazon SQS.  <a href="#">Didacticiel (p. 278)</a>	

## Étapes suivantes

Une fois que vous disposez d'un environnement exécutant une application, vous pouvez déployer une nouvelle version de l'application ou une application totalement différente à tout moment. Le déploiement d'une nouvelle version d'application est très rapide, car il n'est pas nécessaire de mettre en service ni de redémarrer les instances EC2. Pour plus d'informations sur le déploiement d'applications, voir [Déploiement d'une nouvelle version de votre application \(p. 7\)](#).

Une fois que vous avez déployé un ou deux exemples d'application et que vous êtes prêt à développer et à exécuter des applications Node.js localement, consultez la [section suivante \(p. 253\)](#) afin de configurer un environnement de développement Node.js avec tous les outils dont vous avez besoin.

## Configuration de votre environnement de développement Node.js

Configurez un environnement de développement Node.js pour tester votre application localement avant de la déployer dans AWS Elastic Beanstalk. Cette rubrique décrit les étapes de configuration de l'environnement de développement et des liens vers les pages d'installation pour des outils utiles.

Pour accéder aux outils et aux étapes de configuration courants qui s'appliquent à toutes les langues, veuillez consulter [Configuration de votre machine de développement \(p. 1024\)](#).

### Rubriques

- [Installation de Node.js. \(p. 253\)](#)
- [Confirmation de l'installation de npm \(p. 253\)](#)
- [Installez le kit SDK AWS pour Node.js \(p. 253\)](#)
- [Installation d'Express \(p. 254\)](#)

## Installation de Node.js.

Installez Node.js pour exécuter des applications Node.js localement. Si vous n'avez pas de préférence, téléchargez la dernière version prise en charge par Elastic Beanstalk. Pour obtenir la liste des versions prises en charge, accédez à [Node.js](#) dans le document Plateformes prises en charge par AWS Elastic Beanstalk.

Téléchargez Node.js sur [nodejs.org](#).

## Confirmation de l'installation de npm

Node.js utilise le gestionnaire de package npm pour vous aider à installer les outils et les infrastructures à utiliser dans votre application. Comme npm est distribué avec Node.js, vous l'installez automatiquement lorsque vous téléchargez et installez Node.js. Pour confirmer que npm est installé, vous pouvez exécuter la commande suivante :

```
$ npm -v
```

Pour plus d'informations sur npm, visitez le site Web de [npmjs](#).

## Installez le kit SDK AWS pour Node.js

Si vous souhaitez gérer les ressources AWS à partir de votre application, installez le kit SDK AWS pour JavaScript dans Node.js. Installez le SDK avec npm :

```
$ npm install aws-sdk
```

Pour de plus amples informations, veuillez consulter la page d'accueil [Kit SDK AWS pour JavaScript dans Node.js](#).

## Installation d'Express

Express est une infrastructure d'application web qui s'exécute sur Node.js. Pour l'utiliser, configurez Express et créez la structure du projet. Les éléments suivants vous guident à travers la mise en place d'Express sur un système d'exploitation Linux.

### Note

En fonction de votre niveau d'autorisation aux répertoires du système, il se peut que vous ayez besoin de préfixer certaines de ces commandes sudo.

Pour configurer votre environnement de développement Express sur votre ordinateur local

1. Créez un répertoire pour votre application Express.

```
~$ mkdir node-express  
~$ cd node-express
```

2. Installez Express globalement afin de pouvoir accéder à la commande express.

```
/node-express$ npm install -g express-generator
```

3. En fonction de votre système d'exploitation, vous pouvez avoir besoin de définir le chemin d'accès pour exécuter la commande express. Si vous devez définir votre chemin d'accès, utilisez la sortie de l'étape précédente, lorsque vous avez installé Express. Voici un exemple de.

```
/node-express$ export PATH=$PATH:/usr/local/share/npm/bin/express
```

4. Exécutez la commande express. Il en résulte la génération de package.json, app.js, et de quelques répertoires.

```
/node-express$ express
```

Lorsque vous êtes invité à continuer, tapez y.

5. Configurez les dépendances locales.

```
/node-express$ npm install
```

6. Vérifiez si cela fonctionne.

```
/node-express$ npm start
```

Vous devez voir des résultats similaires à ce qui suit :

```
> nodejs@0.0.0 start /home/local/user/node-express  
> node ./bin/www
```

Par défaut, le serveur s'exécute sur le port 3000. Pour le tester, exécutez curl `http://localhost:3000` sur un autre terminal, ou ouvrez un navigateur sur l'ordinateur local et accédez à `http://localhost:3000`.

Appuyez sur Ctrl+C afin d'arrêter le serveur.

## Utilisation de la plateforme Elastic Beanstalk Node.js

### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [the section called “Mise à niveau vers Amazon Linux 2” \(p. 508\)](#).

La plateforme Node.js AWS Elastic Beanstalk est un ensemble de [versions de plateformes](#) pour les applications Web Node.js qui s'exécutent derrière un serveur proxy NGINX.

Elastic Beanstalk fournit des [options de configuration \(p. 658\)](#) que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute sur des instances EC2 dans votre environnement Elastic Beanstalk. Vous pouvez configurer des variables d'environnement nécessaires pour votre application, activer la rotation des journaux sur Amazon S3 et mapper des dossiers dans la source de votre application contenant des fichiers statiques vers des chemins desservis par le serveur proxy.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour [modifier la configuration d'un environnement en cours d'exécution \(p. 671\)](#). Pour éviter de perdre la configuration de votre environnement en le résilient, vous pouvez utiliser des [configurations enregistrées \(p. 779\)](#) pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des [fichiers de configuration \(p. 737\)](#). Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Vous pouvez [inclure un fichier Package.json \(p. 260\)](#) dans votre solution groupée source pour installer des packages pendant le déploiement, fournir une commande start et spécifier la version Node.js que votre application doit utiliser. Vous pouvez inclure un [fichier npm-shrinkwrap.json \(p. 262\)](#) pour verrouiller les versions de dépendances.

La plateforme Node.js inclut un serveur proxy qui diffuse les ressources statiques, achemine le trafic vers votre application et compresse les réponses. Vous pouvez [étendre ou remplacer la configuration du serveur proxy par défaut \(p. 262\)](#) pour les scénarios avancés.

Il existe plusieurs options pour démarrer votre application. Vous pouvez ajouter un [Procfile \(p. 260\)](#) à votre bundle source pour spécifier la commande qui démarre votre application. Lorsque vous ne fournissez pas de Procfile, Elastic Beanstalk exécute `npm start` si vous fournissez un fichier package.json. Si vous n'en fournissez pas non plus, Elastic Beanstalk recherche le fichier app.js ou server.js, dans cet ordre, et l'exécute.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour de plus amples informations sur la priorité et les autres méthodes de modification des paramètres, veuillez consulter [Options de configuration \(p. 658\)](#).

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter [the section called “Extension des plateformes Linux” \(p. 34\)](#).

## Configuration de votre environnement Node.js

Vous pouvez utiliser les paramètres de la plateforme Node.js pour affiner le comportement de vos instances Amazon EC2. Vous pouvez modifier la configuration des instances Amazon EC2 de votre environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour permettre la rotation des journaux sur Amazon S3 et configurer des variables que votre application peut lire à partir de l'environnement.

Pour configurer votre environnement Node.js dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

## Options du conteneur

Vous pouvez spécifier ces options spécifiques à la plateforme :

- Proxy server (Serveur proxy) – Serveur proxy à utiliser sur vos instances d'environnement. Le serveur NGNIX est utilisé par défaut.

## Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

## Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour de plus amples informations sur la configuration des fichiers statiques à l'aide de la console Elastic Beanstalk, veuillez consulter [the section called "Fichiers statiques" \(p. 790\)](#).

## Propriétés de l'environnement

Utilisez la section Environment Properties (Propriétés de l'environnement) pour spécifier des paramètres de configuration d'environnement sur les instances Amazon EC2 exécutant votre application. Ces paramètres sont passés en tant que paires clé-valeur à l'application.

Dans l'environnement Node.js en cours d'exécution dans AWS Elastic Beanstalk, vous pouvez accéder aux variables d'environnement en utilisant `process.env.ENV_VARIABLE` comme dans l'exemple ci-dessous.

```
var endpoint = process.env.API_ENDPOINT
```

La plateforme Node.js définit la variable d'environnement PORT sur le port vers lequel le serveur proxy transfère le trafic. Pour de plus amples informations, veuillez consulter [Configuration du serveur proxy \(p. 262\)](#).

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Configuration d'un environnement Node.js AMI Linux Amazon (antérieure à Amazon Linux 2)

Les catégories de configuration logicielles de console suivantes sont prises en charge uniquement sur un environnement Elastic Beanstalk Node.js qui utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2).

### Options du conteneur

Sur la page de configuration, spécifiez les informations suivantes :

- Serveur proxy) – Indique le serveur web à utiliser comme proxy pour se connecter à Node.js. Le serveur NGINX est utilisé par défaut. Si vous sélectionnez none (aucun), les mappages de fichiers statiques ne prendront pas effet et la compression GZIP sera désactivée.
- Version Node.js – Indique la version de Node.js. Pour obtenir la liste des versions Node.js prises en charge, consultez [Node.js](#) dans le guide AWS Elastic Beanstalk Plateformes.
- Gzip compression (Compression Gzip) – Indique si la compression GZIP est activée. Par défaut, la compression GZIP est activée.
- Commande Node – Vous permet de saisir la commande utilisée pour démarrer l'application Node.js. Une chaîne vide (par défaut) signifie qu'Elastic Beanstalk utilise `app.js`, puis `server.js` et ensuite `npm start`.

## Espaces de noms de la configuration Node.js

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

Vous pouvez choisir le proxy à utiliser sur les instances de votre environnement à l'aide de l'espace de noms `aws:elasticbeanstalk:environment:proxy`. L'exemple suivant configure votre environnement pour qu'il utilise le serveur proxy HTTPD Apache.

Example `.ebextensions/nodejs-settings.config`

```
option_settings:  
  aws:elasticbeanstalk:environment:proxy:  
    ProxyServer: apache
```

Vous pouvez configurer le proxy pour qu'il traite les fichiers statiques à l'aide de l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles`. Pour plus d'informations et pour voir un exemple, veuillez consulter [the section called "Fichiers statiques" \(p. 790\)](#).

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide

de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## La plateforme Node.js de l'AMI Linux Amazon (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Node.js utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), tenez compte des configurations et recommandations spécifiques de cette section.

### Options de configuration spécifiques à la plateforme Node.js

Elastic Beanstalk prend en charge certaines options de configuration spécifiques à la plateforme pour les versions de plateforme Node.js de l'AMI Linux Amazon. Vous pouvez choisir le serveur proxy à exécuter devant votre application, une version spécifique de Node.js à exécuter et la commande utilisée pour exécuter votre application.

Pour le serveur proxy, vous pouvez utiliser un serveur proxy NGINX ou Apache. Vous pouvez définir la valeur `none` sur l'option `ProxyServer`. Dans ce cas, Elastic Beanstalk exécute votre application en mode autonome, pas derrière un serveur proxy. Si votre environnement exécute une application autonome, mettez à jour votre code pour écouter le port vers lequel NGINX transfère le trafic.

```
var port = process.env.PORT || 8080;  
  
app.listen(port, function() {  
    console.log('Server running at http://127.0.0.1:%s', port);  
});
```

### Versions en langage Node.js

En termes de version linguistique prise en charge, la plateforme AMI Amazon Linux Node.js est différente des autres plateformes gérées par Elastic Beanstalk. En effet, chaque version de la plateforme Node.js ne prend en charge que quelques versions linguistiques Node.js. Pour obtenir la liste des versions Node.js prises en charge, consultez [Node.js](#) dans le guide AWS Elastic Beanstalk Plateformes.

Vous pouvez utiliser une option de configuration spécifique à la plateforme pour définir la version de langage. Pour obtenir des instructions, veuillez consulter [the section called “Configuration de votre environnement Node.js” \(p. 256\)](#). Vous pouvez également utiliser la console Elastic Beanstalk pour mettre à jour la version Node.js que votre environnement utilise dans le cadre de la mise à jour de la version de votre plateforme.

#### Note

Lorsque la prise en charge de la version de Node.js que vous utilisez est supprimée de la plateforme, vous devez modifier ou supprimer le paramètre de version avant de procéder à une [mise à jour de la plateforme \(p. 496\)](#). Cela peut se produire lorsqu'une faille de sécurité est identifiée pour une ou plusieurs versions de Node.js.

Lorsque cela se produit, toute tentative de mise à jour vers une nouvelle version de la plateforme qui ne prend pas en charge le paramètre [NodeVersion \(p. 731\)](#) configuré échoue. Pour éviter d'avoir besoin de créer un nouvel environnement, remplacez l'option de configuration `NodeVersion` par une version de Node.js qui est prise en charge à la fois par l'ancienne version de plateforme et par la nouvelle, ou bien [supprimez le paramètre de l'option \(p. 671\)](#), puis effectuez la mise à jour de la plateforme.

Pour configurer la version Node.js de votre environnement dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans la page de présentation de l'environnement, sous Platform (Plateforme), choisissez Change (Changer).
4. Dans la boîte de dialogue Update platform version (Mettre à jour la version de la plateforme), sélectionnez une version de Node.js.

The screenshot shows the 'Update platform version' dialog box. At the top, there's a warning message: "Availability warning" with an info icon. The message states: "This operation replaces your instances; your application is unavailable during the update. instance in service during the update, enable rolling updates. Another option is to clone the environment, which creates a newer version of the platform, and then swap the CNAME of the environment when you are ready to deploy the clone. Learn more at [Updating AWS Elastic Beanstalk Environment](#) Rolling Updates and Deploying Version with Zero Downtime." Below the warning, there are two columns of information:

Platform branch	Current Node.js version
Node.js running on 64bit Amazon Linux	12.14.0
Current platform version	New Node.js version
4.13.0	12.14.1
New platform version	
4.13.0 (Recommended)	

5. Choisissez Enregistrer.

#### Espaces de noms de configuration Node.js

La plateforme d'AMI Amazon Linux Node.js définit des options supplémentaires dans les espaces de noms `aws:elasticbeanstalk:container:nodejs:staticfiles` et `aws:elasticbeanstalk:container:nodejs`.

Le fichier de configuration suivant indique à Elastic Beanstalk d'utiliser `npm start` pour exécuter l'application. Il définit également le type de proxy sur Apache et active la compression. Enfin, il configure le proxy pour qu'il serve des fichiers statiques à partir de deux répertoires sources. Une source est constituée

de fichiers HTML situés sur le chemin `html` sous la racine du site web à partir du répertoire source `statichtml`. L'autre source est constituée de fichiers image situés sur le chemin `images` sous la racine du site web à partir du répertoire source `staticimages`.

#### Example .ebextensions/node-settings.config

```
option_settings:  
  aws:elasticbeanstalk:container:nodejs:  
    NodeCommand: "npm start"  
    ProxyServer: apache  
    GzipCompression: true  
  aws:elasticbeanstalk:container:nodejs:staticfiles:  
    /html: statichtml  
    /images: staticimages
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Configuration du processus de l'application avec un Procfile

Vous pouvez inclure un fichier appelé `Procfile` à la racine de votre bundle source pour spécifier la commande qui démarre votre application.

#### Example Procfile

```
web: node index.js
```

Pour de plus amples informations sur l'utilisation de `Procfile`, veuillez développer la section Buildfile and Procfile (Buildfile et Procfile) dans [the section called “Extension des plateformes Linux” \(p. 34\)](#).

#### Note

Cette fonctionnalité remplace l'option héritée `NodeCommand` dans l'espace de noms `aws:elasticbeanstalk:container:nodejs`.

## Configuration des dépendances de votre application

Votre application peut avoir des dépendances sur certains modules Node.js, comme ceux que vous spécifiez dans les instructions `require()`. Vous pouvez spécifier ces dépendances à l'aide d'un fichier `package.json`. Vous pouvez également inclure les dépendances de votre application dans le bundle source et les déployer avec l'application. Ces deux méthodes alternatives sont détaillées dans les sections suivantes.

### Spécification des dépendances Node.js avec un fichier package.json

Utilisez un fichier `package.json` à la racine de votre source de projet pour spécifier des packages de dépendance et pour fournir une commande `start`. Lorsqu'un fichier `package.json` est présent, Elastic Beanstalk exécute `npm install` pour installer les dépendances. Il utilise également la commande `start` pour démarrer votre application. Pour plus d'informations sur le fichier `package.json`, consultez le [guide package.json](#) sur le site web de Node.js.

Utilisez le mot-clé `scripts` pour fournir une commande `start`. Le mot-clé `scripts` est maintenant utilisé à la place de l'option héritée `NodeCommand` dans l'espace de noms `aws:elasticbeanstalk:container:nodejs`.

### Example package.json – Express

```
{  
  "name": "my-app",  
  "version": "0.0.1",  
  "private": true,  
  "dependencies": {  
    "ejs": "latest",  
    "aws-sdk": "latest",  
    "express": "latest",  
    "body-parser": "latest"  
  },  
  "scripts": {  
    "start": "node app.js"  
  }  
}
```

Utilisez le mot-clé `engines` du fichier `package.json` pour spécifier la version de Node.js que vous souhaitez que votre application utilise. Vous pouvez également spécifier une plage de versions à l'aide de la notation npm. Pour plus d'informations sur la syntaxe des plages de versions, consultez [Gestion sémantique des versions à l'aide de npm](#) sur le site web de Node.js. Le mot-clé `engines` du fichier Node.js `package.json` remplace l'option héritée `NodeVersion` dans l'espace de noms `aws:elasticbeanstalk:container:nodejs`.

### Example `package.json`– Version Node.js unique

```
{  
  ...  
  "engines": { "node" : "14.16.0" }  
}
```

### Example `package.json`– Plage de versions Node.js

```
{  
  ...  
  "engines": { "node" : ">=10 <11" }  
}
```

Lorsqu'une plage de versions est indiquée, Elastic Beanstalk installe la dernière version de Node.js à la disposition de la plateforme dans cette plage. Dans cet exemple, la plage indique que la version doit être supérieure ou égale à la version 10, mais inférieure à la version 11. Par conséquent, Elastic Beanstalk installe la dernière version 10.x.y de Node.js, disponible sur la [plateforme prise en charge](#).

Sachez que vous ne pouvez spécifier qu'une version de Node.js qui correspond à votre branche de plateforme. Par exemple, si vous utilisez la branche de plateforme Node.js 14, vous ne pouvez spécifier qu'une version 14.x.y de Node.js. Vous pouvez utiliser les options de plage de versions prises en charge par npm pour offrir plus de flexibilité. Pour les versions Node.js valides pour chaque branche de plateforme, consultez [Node.js](#) dans le guide Plateformes AWS Elastic Beanstalk.

Par défaut, Elastic Beanstalk installe les dépendances en mode production (`npm install --production`). Si vous souhaitez installer des dépendances de développement sur les instances de votre environnement, définissez la [propriété d'environnement \(p. 634\)](#) `NPM_USE_PRODUCTION` sur `false`.

#### Note

Lorsque la prise en charge de la version de Node.js que vous utilisez est supprimée de la plateforme, vous devez modifier ou supprimer le paramètre de version de Node.js avant de

procéder à une [mise à jour de la plateforme \(p. 496\)](#). Cela peut se produire lorsqu'une faille de sécurité est identifiée pour une ou plusieurs versions de Node.js.

Lorsque cela se produit, toute tentative de mise à jour vers une nouvelle version de la plateforme qui ne prend pas en charge la version Node.js configurée échoue. Pour éviter de créer un nouvel environnement, modifiez le paramètre de version de Node.js dans `package.json` sur une version de Node.js qui est prise en charge à la fois par l'ancienne version de la plateforme et par la nouvelle. Vous avez la possibilité de spécifier une plage de versions Node.js qui inclut une version prise en charge, comme décrit plus haut dans cette rubrique. Vous pouvez également supprimer le paramètre, puis déployer le nouveau bundle source.

## Inclusion des dépendances Node.js dans un répertoire node\_modules

Pour déployer les packages de dépendances sur des instances d'environnement avec votre code d'application, incluez-les dans un répertoire nommé `node_modules` à la racine de la source de votre projet. Node.js recherche les dépendances dans ce répertoire. Pour plus de détails, consultez [Chargement à partir des dossiers node\\_modules](#) dans la documentation Node.js.

### Note

Lorsque vous déployez un répertoire `node_modules` sur une version de plateforme Amazon Linux 2 Node.js, Elastic Beanstalk suppose que vous fournissez vos propres packages de dépendance et évite d'installer les dépendances spécifiées dans un fichier `package.json` ([p. 260](#)).

## Verrouillage des dépendances avec NPM Shrinkwrap

La plateforme Node.js exécute `npm install` lors de chaque déploiement. Lorsque de nouvelles versions de vos dépendances sont disponibles, elles sont installées lorsque vous déployez l'application, ce qui peut ralentir de manière significative le déploiement.

Pour éviter la mise à jour des dépendances, il est possible de créer un fichier `npm-shrinkwrap.json` qui verrouille les dépendances de votre application sur la version actuelle.

```
$ npm install
$ npm shrinkwrap
wrote npm-shrinkwrap.json
```

Incluez ce fichier dans votre bundle de fichiers source afin que les dépendances ne soient installées qu'à une seule reprise.

## Configuration du serveur proxy

Elastic Beanstalk peut utiliser nginx ou Apache HTTPD comme proxy inverse pour mapper votre application à votre équilibrEUR de charge Elastic Load Balancing sur le port 80. La valeur par défaut est nginx. Elastic Beanstalk fournit une configuration de proxy par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Par défaut, Elastic Beanstalk configure le serveur proxy pour transmettre les demandes à votre application sur le port 8080. Vous pouvez remplacer le port par défaut en définissant la `PORT`propriété d'environnement ([p. 256](#)) sur le port écouté par votre application.

### Notes

- Le port que votre application écoute n'affecte pas le port que le serveur nginx écoute pour recevoir des demandes de l'équilibrEUR de charge.
- Sur les versions Node.js de la plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), Elastic Beanstalk configure le proxy pour transférer les demandes à votre application sur le

port 8081. Pour obtenir des instructions, consultez [the section called “Configuration du proxy sur l'AMI Amazon Linux \(antérieure à Amazon Linux 2\)” \(p. 263\)](#) sur cette page.

Toutes les plateformes Amazon Linux 2 prennent en charge une configuration de proxy uniforme. Pour obtenir des instructions sur la manière de configurer le serveur proxy sur les nouvelles versions de plateforme Amazon Corretto qui exécutent Amazon Linux 2, développez la section Configuration du proxy inverse dans [the section called “Extension des plateformes Linux” \(p. 34\)](#).

## Configuration du proxy sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)

Si votre environnement Elastic Beanstalk Node.js utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations de cette section.

### Extension et remplacement de la configuration du proxy par défaut

La plateforme Node.js utilise un proxy inverse pour acheminer les requêtes du port 80 de l'instance vers le port 8081 d'écoute de votre application. Elastic Beanstalk fournit une configuration de proxy par défaut que vous pouvez étendre ou remplacer totalement par votre propre configuration.

Pour étendre la configuration par défaut, ajoutez des fichiers .conf à /etc/nginx/conf.d avec un fichier de configuration. Pour un exemple spécifique, veuillez consulter [Suspension des connexions HTTPS sur des instances EC2 exécutant Node.js \(p. 807\)](#).

La plateforme Node.js définit la variable d'environnement PORT sur le port vers lequel le serveur proxy transfère le trafic. Lisez cette variable dans votre code pour configurer le port de votre application.

```
var port = process.env.PORT || 3000;

var server = app.listen(port, function () {
  console.log('Server running at http://127.0.0.1:' + port + '/');
});
```

La configuration NGINX par défaut achemine le trafic vers un serveur en amont nommé nodejs à l'adresse 127.0.0.1:8081. Vous pouvez supprimer la configuration par défaut et fournir une configuration personnalisée dans un [fichier de configuration \(p. 737\)](#).

### Example .ebextensions/proxy.config

L'exemple suivant supprime la configuration par défaut et ajoute une configuration personnalisée qui achemine le trafic vers le port 5000 au lieu du port 8081.

```
files:
  /etc/nginx/conf.d/proxy.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      upstream nodejs {
        server 127.0.0.1:5000;
        keepalive 256;
      }

      server {
        listen 8080;

        if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
          set $year $1;
          set $month $2;
          set $day $3;
        }
      }
```

```
        set $hour $4;
    }
access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour healthd;
access_log /var/log/nginx/access.log main;

location / {
    proxy_pass http://nodejs;
    proxy_set_header Connection "";
    proxy_http_version 1.1;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
}

gzip on;
gzip_comp_level 4;
gzip_types text/html text/plain text/css application/json application/x-javascript
text/xml application/xml+rss text/javascript;

location /static {
    alias /var/app/current/static;
}

}

/opt/elasticbeanstalk/hooks/configdeploy/post/99_kill_default_nginx.sh:
mode: "000755"
owner: root
group: root
content: |
#!/bin/bash -xe
rm -f /etc/nginx/conf.d/00_elastic_beanstalk_proxy.conf
service nginx stop
service nginx start

container_commands:
removeconfig:
command: "rm -f /tmp/deployment/config/
/etc/nginx/conf.d/00_elastic_beanstalk_proxy.conf /etc/nginx/
conf.d/00_elastic_beanstalk_proxy.conf"
```

L'exemple de configuration (`/etc/nginx/conf.d/proxy.conf`) utilise la configuration par défaut dans `/etc/nginx/conf.d/00_elastic_beanstalk_proxy.conf` comme base pour inclure le bloc de serveur par défaut avec des paramètres de compression et de journalisation et un mappage de fichiers statiques.

La commande `removeconfig` supprime la configuration par défaut du conteneur pour que le serveur proxy utilise bien la configuration personnalisée. Elastic Beanstalk recrée la configuration par défaut lorsque chaque configuration est déployée. Pour ce faire, dans l'exemple suivant, un hook de déploiement après configuration (`/opt/elasticbeanstalk/hooks/configdeploy/post/99_kill_default_nginx.sh`) est ajouté. Celui-ci supprime la configuration par défaut et redémarre le serveur proxy.

#### Note

Il est possible que la configuration par défaut change dans les prochaines versions de la plateforme Node.js. Utilisez la version la plus récente de la configuration comme base pour vos personnalisations afin de garantir leur compatibilité.

Si vous remplacez la configuration par défaut, vous devez définir les mappages de fichiers statiques et la compression GZIP. Cela s'explique par le fait que la plateforme ne peut pas appliquer les [paramètres standard](#) (p. 257).

# Déploiement d'une application Express sur Elastic Beanstalk

Cette section vous guide dans le déploiement d'un exemple d'application sur Elastic Beanstalk en utilisant l'interface de ligne de commande Elastic Beanstalk (CLI EB) et Git, puis en mettant à jour l'application afin d'utiliser le cadre [Express](#).

## Prerequisites

Ce didacticiel nécessite l'utilisation du langage Node.js, de son gestionnaire de packages appelé npm et de l'infrastructure d'application web Express. Pour de plus amples informations sur l'installation de ces composants et la configuration de votre environnement de développement local, veuillez consulter [Configuration de votre environnement de développement Node.js \(p. 253\)](#).

### Note

Dans le cadre de ce didacticiel, vous n'avez pas besoin d'installer le kit SDK AWS pour Node.js, qui est également mentionné dans [Configuration de votre environnement de développement Node.js \(p. 253\)](#).

Ce didacticiel nécessite également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter [Installation de l'interface de ligne de commande EB \(p. 1028\)](#) et [Configuration de l'interface de ligne de commande EB \(p. 1036\)](#).

## Initialisation de Git

La configuration Node.js et Express prérequis entraîne la création d'une structure de projet Express dans le dossier `node-express`. Si vous n'avez pas encore généré de projet Express, exécutez la commande suivante. Pour en savoir plus, consultez [Installation d'Express \(p. 254\)](#).

```
~/node-express$ express && npm install
```

A présent, nous allons configurer un référentiel Git dans ce dossier.

### Configuration d'un référentiel Git

1. Initialisez le référentiel Git. Si Git n'est pas déjà installé, téléchargez-le depuis le site [Téléchargements Git](#).

```
~/node-express$ git init
```

2. Créez un fichier nommé `.gitignore` et ajoutez-y les fichiers et répertoires suivants. Ces fichiers ne seront pas ajoutés au référentiel. Cette étape n'est pas obligatoire, mais elle est recommandée.

```
node-express/.gitignore
```

```
node_modules/  
.gitignore  
.elasticbeanstalk/
```

## Créer un environnement Elastic Beanstalk

Configurez un référentiel de la CLI EB pour votre application et créez un environnement Elastic Beanstalk qui exécute la plateforme Node.js.

1. Créez un référentiel à l'aide de la commande eb init.

```
~/node-express$ eb init --platform node.js --region us-east-2
Application node-express has been created.
```

Cette commande crée un fichier de configuration dans un dossier nommé `.elasticbeanstalk` qui spécifie les paramètres de création d'environnements pour votre application et crée une application Elastic Beanstalk dont le nom est basé sur le dossier actif.

2. Créez un environnement qui exécute un exemple d'application à l'aide de la commande eb create.

```
~/node-express$ eb create --sample node-express-env
```

Cette commande crée un environnement à charge équilibrée avec les paramètres par défaut de la plateforme Node.js et les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

#### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021. Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Équilibreur de charge – Équilibreur de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibreur de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibreur de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibreur de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.

- Alarms Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
  - Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
  - Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme [sous-domaine.région.elasticbeanstalk.com](#).
3. Une fois l'environnement créé, utilisez la commande eb open pour ouvrir l'URL de l'environnement dans le navigateur par défaut.

```
~/node-express$ eb open
```

## Mise à jour de l'application

Une fois que vous avez créé un environnement avec un exemple d'application, vous pouvez le mettre à jour à l'aide de votre propre application. Dans cette étape, nous mettons à jour l'exemple d'application afin d'utiliser l'infrastructure Express.

Pour mettre à jour votre application afin d'utiliser Express

1. Sur votre ordinateur local, créez un répertoire .ebextensions dans le répertoire de niveau supérieur de votre groupe source. Dans cet exemple, nous utilisons node-express/.ebextensions.
2. Ajoutez un fichier de configuration qui définit la commande Node sur « npm start » :

```
node-express/.ebextensions/nodecommand.config
```

```
option_settings:  
  aws:elasticbeanstalk:container:nodejs:  
    NodeCommand: "npm start"
```

Pour de plus amples informations, veuillez consulter [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

3. Indexez les fichiers :

```
~/node-express$ git add .  
~/node-express$ git commit -m "First express app"
```

4. Déployez les modifications :

```
~/node-express$ eb deploy
```

5. Une fois que votre environnement est prêt (il apparaît en vert), actualisez l'URL pour vérifier s'il a fonctionné. Une page web intitulée Welcome to Express doit s'afficher.

Ensuite, nous devons mettre à jour l'application Express afin de traiter les fichiers statiques et d'ajouter une nouvelle page.

Pour configurer les fichiers statiques et ajouter une nouvelle page à votre application Express

1. Ajoutez un deuxième fichier de configuration comportant le contenu suivant :

**node-express/.ebextensions/staticfiles.config**

```
option_settings:  
  aws:elasticbeanstalk:container:nodejs:staticfiles:  
    /public: /public
```

Ce paramètre configure le serveur proxy afin qu'il transmette les fichiers du dossier public au chemin /public de l'application. [La distribution de fichiers de manière statique \(p. 257\)](#) depuis le proxy réduit la charge sur votre application.

2. Mettez en commentaire le mappage statique dans node-express/app.js. Cette étape n'est pas obligatoire, mais elle constitue un bon test pour vérifier si les mappages statiques sont correctement configurés.

```
// app.use(express.static(path.join(__dirname, 'public')));
```

3. Ajoutez vos fichiers mis à jour à votre répertoire local et validez les modifications.

```
~/node-express$ git add .ebextensions/ app.js  
~/node-express$ git commit -m "Serve stylesheets statically with nginx."
```

4. Addition node-express/routes/hike.js. Saisissez les données ci-dessous :

```
exports.index = function(req, res) {  
  res.render('hike', {title: 'My Hiking Log'});  
};  
  
exports.add_hike = function(req, res) {  
};
```

5. Mettez à jour node-express/app.js en y incluant trois nouvelles lignes.

Commencez par ajouter la ligne suivante afin d'ajouter un require pour cette route :

```
var hike = require('./routes/hike');
```

Votre fichier doit être similaire à l'extrait suivant :

```
var express = require('express');  
var path = require('path');  
var hike = require('./routes/hike');
```

Ajoutez ensuite les deux lignes suivantes au fichier node-express/app.js après var app = express();

```
app.get('/hikes', hike.index);  
app.post('/add_hike', hike.add_hike);
```

Votre fichier doit être similaire à l'extrait suivant :

```
var app = express();  
app.get('/hikes', hike.index);  
app.post('/add_hike', hike.add_hike);
```

6. Copiez node-express/views/index.jade dans node-express/views/hike.jade.

```
~/node-express$ cp views/index.jade views/hike.jade
```

7. Ajoutez vos fichiers au répertoire local, validez les modifications et déployez votre application mise à jour.

```
~/node-express$ git add .
~/node-express$ git commit -m "Add hikes route and template."
~/node-express$ eb deploy
```

8. Votre environnement est mis à jour au bout de quelques minutes. Une fois que votre environnement est prêt (il apparaît en vert), assurez-vous qu'il fonctionne. Pour ce faire, actualisez la page de votre navigateur et ajoutez **hikes** à la fin de l'URL (par exemple, <http://node-express-env-syypntcz2q.elasticbeanstalk.com/hikes>).

Vous devriez voir une page web intitulée My Hiking Log.

## Nettoyage

Si vous avez terminé de travailler avec Elastic Beanstalk, vous pouvez arrêter votre environnement.

Utilisez la commande eb terminate pour suspendre votre environnement et toutes les ressources qu'il contient.

```
~/node-express$ eb terminate
The environment "node-express-env" and all associated instances will be terminated.
To confirm, type the environment name: node-express-env
INFO: terminateEnvironment is starting.
...
```

# Déploiement d'une application Express avec une mise en cluster dans Elastic Beanstalk

Ce tutoriel explique comment déployer un exemple d'application dans Elastic Beanstalk via l'interface de ligne de commande Elastic Beanstalk (CLI EB), puis comment mettre à jour l'application afin d'utiliser le cadre [Express](#), [Amazon ElastiCache](#) et la mise en cluster. Le clustering permet d'améliorer la haute disponibilité, les performances et la sécurité de votre application web. Pour en savoir plus sur Amazon ElastiCache, veuillez consulter [Qu'est-ce qu'Amazon ElastiCache for Memcached ?](#) dans le Guide de l'utilisateur Amazon ElastiCache for Memcached.

### Note

Cet exemple crée des ressources AWS, qui peuvent éventuellement vous être facturées. Pour plus d'informations sur la tarification AWS, consultez <http://aws.amazon.com/pricing/>. Certains services font partie du niveau d'offre gratuite d'AWS. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour plus d'informations, consultez <http://aws.amazon.com/free/>.

## Prerequisites

Ce didacticiel nécessite l'utilisation du langage Node.js, de son gestionnaire de packages appelé npm et de l'infrastructure d'application web Express. Pour de plus amples informations sur l'installation de ces composants et la configuration de votre environnement de développement local, veuillez consulter [Configuration de votre environnement de développement Node.js \(p. 253\)](#).

### Note

Dans le cadre de ce didacticiel, vous n'avez pas besoin d'installer le kit SDK AWS pour Node.js, qui est également mentionné dans [Configuration de votre environnement de développement Node.js \(p. 253\)](#).

Ce didacticiel nécessite également l'interface de ligne de commande Elastic Beanstalk (CLI EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter [Installation de l'interface de ligne de commande EB \(p. 1028\)](#) et [Configuration de l'interface de ligne de commande EB \(p. 1036\)](#).

## Créer un environnement Elastic Beanstalk

Configurez un référentiel de la CLI EB pour votre application et créez un environnement Elastic Beanstalk qui exécute la plateforme Node.js.

1. Créez un référentiel à l'aide de la commande eb init.

```
~/node-express$ eb init --platform node.js --region us-east-2
Application node-express has been created.
```

Cette commande crée un fichier de configuration dans un dossier nommé `.elasticbeanstalk` qui spécifie les paramètres de création d'environnements pour votre application et crée une application Elastic Beanstalk dont le nom est basé sur le dossier actif.

2. Créez un environnement qui exécute un exemple d'application à l'aide de la commande eb create.

```
~/node-express$ eb create --sample node-express-env
```

Cette commande crée un environnement à charge équilibrée avec les paramètres par défaut de la plateforme Node.js et les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021. Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibrEUR de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
  - ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
  - Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
  - Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
  - Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
  - Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
  - Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
  - Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme [\*\*sous-domaine.région.elasticbeanstalk.com\*\*](#).
3. Une fois l'environnement créé, utilisez la commande eb open pour ouvrir l'URL de l'environnement dans le navigateur par défaut.

```
~/node-express$ eb open
```

## Mise à jour de l'application

Mettez à jour l'exemple d'application dans l'environnement Elastic Beanstalk pour utiliser le cadre Express.

Vous pouvez télécharger le code source final à partir de [nodejs-example-express-elasticache.zip](#).

### Note

La configuration prérequise de l'environnement de développement génère une structure de projet Express dans le dossier node-express. Si vous n'avez pas encore généré de projet Express, exécutez la commande suivante. Pour en savoir plus, consultez [Installation d'Express \(p. 254\)](#).

```
~/node-express$ express && npm install
```

Pour mettre à jour votre application afin d'utiliser Express

1. Renommez node-express/app.js en node-express/express-app.js.

```
node-express$ mv app.js express-app.js
```

2. Mettez à jour la ligne var app = express(); dans node-express/express-app.js avec les éléments suivants :

```
var app = module.exports = express();
```

3. Sur votre ordinateur local, créez un fichier nommé `node-express/app.js` avec le code suivant.

```
var cluster = require('cluster'),
    app = require('./express-app');

var workers = {},
    count = require('os').cpus().length;

function spawn(){
    var worker = cluster.fork();
    workers[worker.pid] = worker;
    return worker;
}

if (cluster.isMaster) {
    for (var i = 0; i < count; i++) {
        spawn();
    }
    cluster.on('death', function(worker) {
        console.log('worker ' + worker.pid + ' died. spawning a new process...');
        delete workers[worker.pid];
        spawn();
    });
} else {
    app.listen(process.env.PORT || 5000);
}
```

4. Déployez l'application mise à jour.

```
node-express$ eb deploy
```

5. Votre environnement est mis à jour au bout de quelques minutes. Une fois que votre environnement est prêt (il apparaît en vert), actualisez l'URL pour vérifier s'il a fonctionné. Vous devriez voir une page web indiquant « Welcome to Express ».

Vous pouvez accéder aux journaux pour vos instances EC2 qui exécutent votre application. Pour plus d'informations sur l'accès à vos journaux, consultez [Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk \(p. 884\)](#).

Nous allons maintenant mettre à jour l'application Express afin d'utiliser Amazon ElastiCache.

#### Mise à jour de votre application Express pour utiliser Amazon ElastiCache

1. Sur votre ordinateur local, créez un répertoire `.ebextensions` dans le répertoire de niveau supérieur de votre groupe source. Dans cet exemple, nous utilisons `node-express/.ebextensions`.
2. Créez un fichier de configuration `node-express/.ebextensions/elasticache-iam-with-script.config` avec l'extrait suivant. Pour plus d'informations sur le fichier de configuration, consultez [Espaces de noms de la configuration Node.js \(p. 257\)](#). Il en résulte la création d'un utilisateur IAM avec les autorisations requises pour découvrir les nœuds ElastiCache et l'écriture dans un fichier chaque fois que le cache change. Vous pouvez également copier le fichier à partir de [nodejs-example-express-elasticache.zip](#). Pour plus d'informations sur les propriétés d'ElastiCache, consultez [Exemple : ElastiCache \(p. 766\)](#).

#### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

```
Resources:
  MyCacheSecurityGroup:
    Type: 'AWS::EC2::SecurityGroup'
    Properties:
      GroupDescription: "Lock cache down to webserver access only"
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort:
            Fn::GetOptionSetting:
              OptionName: CachePort
              DefaultValue: 11211
          ToPort:
            Fn::GetOptionSetting:
              OptionName: CachePort
              DefaultValue: 11211
          SourceSecurityGroupName:
            Ref: AWSEBSecurityGroup
  MyElastiCache:
    Type: 'AWS::ElastiCache::CacheCluster'
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName: CacheNodeType
          DefaultValue: cache.t2.micro
      NumCacheNodes:
        Fn::GetOptionSetting:
          OptionName: NumCacheNodes
          DefaultValue: 1
      Engine:
        Fn::GetOptionSetting:
          OptionName: Engine
          DefaultValue: redis
    VpcSecurityGroupIds:
      -
        Fn::GetAtt:
          - MyCacheSecurityGroup
          - GroupId
  AWSEBAutoScalingGroup :
    Metadata :
      ElasticCacheConfig :
        CacheName :
          Ref : MyElastiCache
        CacheSize :
          Fn::GetOptionSetting:
            OptionName : NumCacheNodes
            DefaultValue: 1
    WebServerUser :
      Type : AWS::IAM::User
      Properties :
        Path : "/"
      Policies:
        -
          PolicyName: root
          PolicyDocument :
            Statement :
              -
                Effect : Allow
                Action :
                  - cloudformation:DescribeStackResource
                  - cloudformation>ListStackResources
                  - elasticache:DescribeCacheClusters
            Resource : "*"
  WebServerKeys :
    Type : AWS::IAM::AccessKey
```

```
Properties :  
    UserName :  
        Ref: WebServerUser  
  
Outputs:  
    WebsiteURL:  
        Description: sample output only here to show inline string function parsing  
        Value: |  
            http://`{ "Fn::GetAtt" : [ "AWSEBLoadBalancer", "DNSName" ] }`  
    MyElastiCacheName:  
        Description: Name of the elasticache  
        Value:  
            Ref : MyElastiCache  
    NumCacheNodes:  
        Description: Number of cache nodes in MyElastiCache  
        Value:  
            Fn::GetOptionSetting:  
                OptionName : NumCacheNodes  
                DefaultValue: 1  
  
files:  
    "/etc/cfn/cfn-credentials" :  
        content : |  
            AWSAccessKeyId=`{ "Ref" : "WebServerKeys" }`  
            AWSSecretKey=`{ "Fn::GetAtt" : ["WebServerKeys", "SecretAccessKey"] }`  
        mode : "000400"  
        owner : root  
        group : root  
  
    "/etc/cfn/get-cache-nodes" :  
        content : |  
            # Define environment variables for command line tools  
            export AWS_ELASTICACHE_HOME="/home/ec2-user/elasticache/$(ls /home/ec2-user/elasticache/)"  
            export AWS_CLOUDFORMATION_HOME=/opt/aws/apitools/cfn  
            export PATH=$AWS_CLOUDFORMATION_HOME/bin:$AWS_ELASTICACHE_HOME/bin:$PATH  
            export AWS_CREDENTIAL_FILE=/etc/cfn/cfn-credentials  
            export JAVA_HOME=/usr/lib/jvm/jre  
  
            # Grab the Cache node names and configure the PHP page  
            aws cloudformation list-stack-resources --stack `{"Ref" : "AWS::StackName"}`  
--region `{"Ref" : "AWS::Region"}` --output text | grep MyElastiCache | awk '{print $4}' | xargs -I {} aws elasticache describe-cache-clusters --cache-cluster-id {}  
--region `{"Ref" : "AWS::Region"}` --show-cache-node-info --output text | grep '^ENDPOINT' | awk '{print $2 ":" $3}' > `{"Fn::GetOptionSetting" : {"OptionName" : "NodeListPath", "DefaultValue" : "/var/www/html/nodelist" }}`  
        mode : "000500"  
        owner : root  
        group : root  
  
    "/etc/cfn/hooks.d/cfn-cache-change.conf" :  
        "content": |  
            [cfn-cache-size-change]  
            triggers=post.update  
            path=Resources.AWSEBAutoScalingGroup.Metadata.ElastiCacheConfig  
            action=/etc/cfn/get-cache-nodes  
            runas=root  
  
sources :  
    "/home/ec2-user/elasticache" : "https://elasticache-downloads.s3.amazonaws.com/  
AmazonElastiCacheCli-latest.zip"  
  
commands:  
    make-elasticache-executable:  
        command: chmod -R ugo+x /home/ec2-user/elasticache/*bin/*
```

```
packages :  
  "yum" :  
    "aws-apitools-cfn" : []  
  
container_commands:  
  initial_cache_nodes:  
    command: /etc/cfn/get-cache-nodes
```

3. Sur votre ordinateur local, créez un fichier de configuration `node-express/.ebextensions/elasticache_settings.config` avec l'extrait suivant pour configurer ElastiCache.

```
option_settings:  
  "aws:elasticbeanstalk:customoption":  
    CacheNodeType: cache.t2.micro  
    NumCacheNodes: 1  
    Engine: memcached  
    NodeListPath: /var/nodelist
```

4. Sur votre ordinateur local, remplacez `node-express/express-app.js` par l'extrait suivant. Ce fichier lit la liste des nœuds depuis le disque (`/var/nodelist`) et configure Express de façon à utiliser memcached comme magasin de sessions s'il existe des nœuds. Votre fichier doit se présenter comme suit :

```
/**  
 * Module dependencies.  
 */  
  
var express = require('express'),  
    session = require('express-session'),  
    bodyParser = require('body-parser'),  
    methodOverride = require('method-override'),  
    cookieParser = require('cookie-parser'),  
    fs = require('fs'),  
    filename = '/var/nodelist',  
    app = module.exports = express();  
  
var MemcachedStore = require('connect-memcached')(session);  
  
function setup(cacheNodes) {  
  app.use(bodyParser.raw());  
  app.use(methodOverride());  
  if (cacheNodes) {  
    app.use(cookieParser());  
  
    console.log('Using memcached store nodes:');  
    console.log(cacheNodes);  
  
    app.use(session({  
      secret: 'your secret here',  
      resave: false,  
      saveUninitialized: false,  
      store: new MemcachedStore({'hosts': cacheNodes})  
    }));  
  } else {  
    console.log('Not using memcached store.');//  
    app.use(cookieParser('your secret here'));  
    app.use(session());  
  }  
  
  app.get('/', function(req, resp){  
    if (req.session.views) {  
      req.session.views++  
      resp.setHeader('Content-Type', 'text/html')  
      resp.write('Views: ' + req.session.views)  
    }  
  })  
}
```

```
        resp.end()
    } else {
        req.session.views = 1
        resp.end('Refresh the page!')
    }
});

if (!module.parent) {
    console.log('Running express without cluster.');
    app.listen(process.env.PORT || 5000);
}
}

// Load elasticache configuration.
fs.readFile(filename, 'UTF8', function(err, data) {
    if (err) throw err;

    var cacheNodes = [];
    if (data) {
        var lines = data.split('\n');
        for (var i = 0 ; i < lines.length ; i++) {
            if (lines[i].length > 0) {
                cacheNodes.push(lines[i]);
            }
        }
    }
    setup(cacheNodes);
});
}
```

5. Sur votre ordinateur local, mettez à jour `node-express/package.json` pour ajouter quatre dépendances.

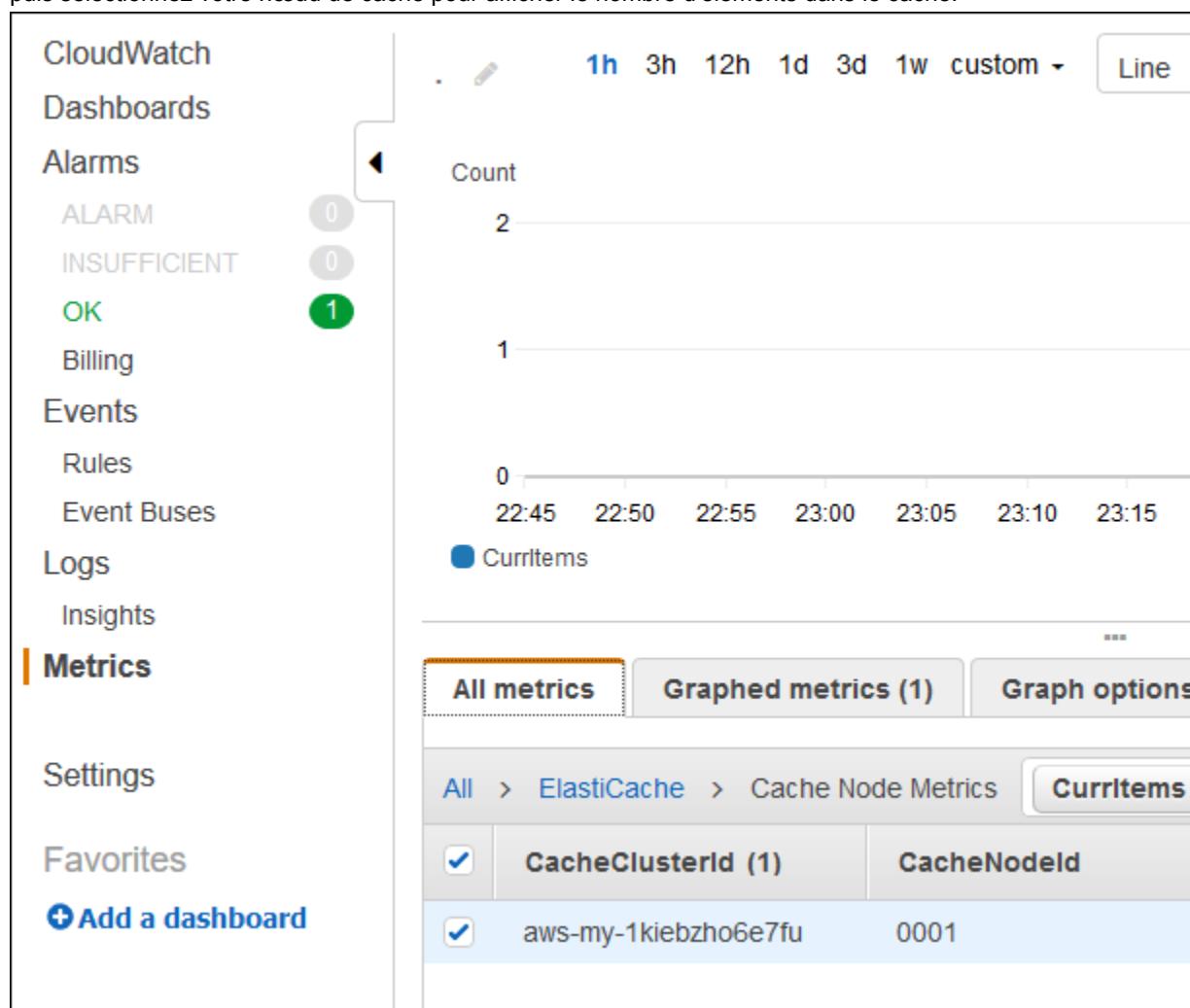
```
{
  "name": "node-express",
  "version": "0.0.0",
  "private": true,
  "scripts": {
    "start": "node ./bin/www"
  },
  "dependencies": {
    "cookie-parser": "*",
    "debug": "~2.6.9",
    "express": "~4.16.0",
    "http-errors": "~1.6.2",
    "jade": "~1.11.0",
    "morgan": "~1.9.0",
    "connect-memcached": "*",
    "express-session": "*",
    "body-parser": "*",
    "method-override": "*"
  }
}
```

6. Déployez l'application mise à jour.

```
node-express$ eb deploy
```

7. Votre environnement est mis à jour au bout de quelques minutes. Une fois que votre environnement est prêt (il apparaît en vert), assurez-vous que le code a fonctionné.
  - a. Accédez à la [Amazon CloudWatch console \(console Amazon CloudWatch\)](#) pour consulter vos métriques ElastiCache. Pour afficher vos métriques ElastiCache, sélectionnez Metrics dans le

volet de gauche, puis recherchez CurrItems. Sélectionnez ElastiCache > Cache Node Metrics, puis sélectionnez votre nœud de cache pour afficher le nombre d'éléments dans le cache.



#### Note

Vérifiez que vous consultez bien la même région que celle dans laquelle vous avez déployé votre application.

Si vous copiez et collez l'URL de votre application dans un autre navigateur web, la valeur de CurrItem devrait augmenter au bout de 5 minutes.

- b. Prenez un instantané de vos journaux. Pour de plus amples informations sur la récupération des journaux, veuillez consulter [Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk \(p. 884\)](#).
- c. Vérifiez le fichier `/var/log/nodejs/nodejs.log` dans le groupe des journaux. Le résultat devrait être similaire à ce qui suit :

```
Using memcached store nodes:  
[ 'aws-my-1oys9co8ztluo.liwtrn.0001.use1.cache.amazonaws.com:11211' ]
```

## Nettoyage

Si vous ne souhaitez plus exécuter votre application, vous pouvez effectuer un nettoyage en suspendant votre environnement et en supprimant votre application.

Utilisez la commande `eb terminate` pour mettre votre environnement hors service et la commande `eb delete` pour supprimer votre application.

Pour résilier votre environnement

Depuis le répertoire où vous avez créé votre référentiel local, exécutez `eb terminate`.

```
$ eb terminate
```

Ce processus peut prendre quelques minutes. Elastic Beanstalk affiche un message une fois que l'environnement est arrêté.

## Déploiement d'une application Node.js avec DynamoDB vers Elastic Beanstalk

Ce tutoriel et ce [modèle d'application](#) vous guident tout au long du processus de déploiement d'une application Node.js qui utilise le kit SDK AWS pour JavaScript dans Node.js pour interagir avec Amazon DynamoDB. Vous allez créer une table DynamoDB externe à l'environnement AWS Elastic Beanstalk et configurer l'application pour qu'elle utilise cette table externe au lieu d'en créer une dans l'environnement. Dans un environnement de production, la table reste indépendante de l'environnement Elastic Beanstalk afin de garantir une protection contre la perte accidentelle de données et vous permettre d'exécuter des [déploiements bleu/vert](#) (p. 486).

L'exemple d'application du tutoriel utilise une table DynamoDB pour stocker les données de texte fournies par l'utilisateur. L'exemple d'application utilise des [fichiers de configuration](#) (p. 737) pour créer la table et une rubrique Amazon Simple Notification Service. Il montre également comment utiliser un [fichier package.json](#) (p. 260) pour installer les packages au cours du déploiement.

### Sections

- [Prerequisites](#) (p. 278)
- [Lancer un environnement Elastic Beanstalk](#) (p. 279)
- [Ajout d'autorisations aux instances de votre environnement](#) (p. 281)
- [Déploiement de l'exemple d'application](#) (p. 281)
- [Créez une table DynamoDB](#) (p. 283)
- [Mise à jour des fichiers de configuration de l'application](#) (p. 284)
- [Configuration de votre environnement pour une haute disponibilité](#) (p. 286)
- [Cleanup](#) (p. 286)
- [Étapes suivantes](#) (p. 287)

## Prerequisites

- Avant de commencer, téléchargez le bundle de fichiers source de l'exemple d'application sur GitHub : [eb-node-express-sample-v1.1.zip](#).
- Pour exécuter les commandes détaillées dans les procédures, vous aurez également besoin d'un shell ou d'un terminal de ligne de commande. Les exemples de commande sont précédés d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant :

```
~/eb-project$ this is a command
this is output
```

### Note

Vous pouvez exécuter toutes les commandes de ce tutoriel sur une machine virtuelle Linux, un ordinateur OS X ou une instance Amazon EC2 exécutant Amazon Linux. Si vous avez besoin d'un environnement de développement, vous pouvez lancer un environnement Elastic Beanstalk à instance unique et vous y connecter avec SSH.

- Ce tutoriel utilise un utilitaire ZIP de ligne de commande pour créer un bundle de fichiers source que vous pouvez déployer sur Elastic Beanstalk. Pour utiliser la commande `zip` sous Windows, vous pouvez installer `UnxUtils`, une collection simple d'utilitaires de ligne de commande utiles tels que `zip` et `ls`. (Sinon, vous pouvez utiliser l'[Explorateur Windows \(p. 416\)](#) ou tout autre utilitaire ZIP pour créer des archives de bundle source.)

### Pour installer UnxUtils

1. Téléchargement [UnxUtils](#).
2. Extrayez l'archive dans un répertoire local. Par exemple, `C:\Program Files (x86)`.
3. Ajoutez le chemin d'accès aux fichiers binaires à votre variable utilisateur PATH sous Windows. Par exemple, `C:\Program Files (x86)\UnxUtils\usr\local\wbin`.
  - a. Appuyez sur la touche Windows et entrez **environment variables**.
  - b. Choisissez Modifier les variables d'environnement pour votre compte.
  - c. Choisissez PATH, puis Modifier.
  - d. Ajoutez des chemins d'accès dans le champ Valeur de la variable, en les séparant par des points virgules. Par exemple: `C:\item1\path;C:\item2\path`
  - e. Choisissez OK deux fois pour appliquer les nouveaux paramètres.
  - f. Fermez toutes les fenêtres d'invite de commande en cours d'exécution, puis rouvrez une fenêtre d'invite de commande.
4. Ouvrez une nouvelle fenêtre d'invite de commande et exécutez la commande `zip` pour vérifier qu'elle fonctionne.

```
> zip -h
Copyright (C) 1990-1999 Info-ZIP
Type 'zip "-L"' for software license.
...
```

## Lancer un environnement Elastic Beanstalk

Vous utilisez la console Elastic Beanstalk pour lancer un environnement Elastic Beanstalk. Vous choisissez la plateforme Node.js et acceptez les paramètres par défaut et l'exemple de code. Une fois que vous avez configuré les autorisations de l'environnement, vous déployez l'exemple d'application que vous avez téléchargé à partir de GitHub.

### Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créez une application.

Elastic Beanstalk prend environ cinq minutes pour créer l'environnement avec les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

#### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme [sous-domaine.région.elasticbeanstalk.com](#).

Elastic Beanstalk gère toutes ces ressources. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Ajout d'autorisations aux instances de votre environnement

Votre application s'exécute sur une ou plusieurs instances EC2 derrière un équilibrEUR de charge, traitant les demandes HTTP provenant d'Internet. Lorsqu'elle reçoit une demande qui requiert l'utilisation des services AWS, l'application utilise les autorisations de l'instance sur laquelle elle s'exécute pour accéder à ces services.

L'exemple d'application utilise les autorisations d'instance pour écrire des données dans une table DynamoDB et pour envoyer des notifications à une rubrique Amazon SNS avec le kit SDK pour JavaScript dans Node.js. Ajoutez les stratégies gérées suivantes au [profil d'instance \(p. 22\)](#) par défaut pour accorder aux instances EC2 de votre environnement des autorisations pour accéder à DynamoDB et Amazon SNS :

- AmazonDynamoDBFullAccess
- AmazonSNSFullAccess

Pour ajouter des stratégies au profil d'instance par défaut

1. Ouvrez la page [Roles \(Rôles\)](#) dans la console IAM.
2. Choisissez aws-elasticbeanstalk-ec2-role.
3. Sous l'onglet Autorisations, choisissez Attach policies (Attacher des stratégies).
4. Sélectionnez la stratégie gérée relative aux services supplémentaires utilisés par votre application. Par exemple, [AmazonSNSFullAccess](#) ou [AmazonDynamoDBFullAccess](#).
5. Choisissez Attacher la stratégie.

Pour de plus amples informations sur la gestion des profils d'instance, veuillez consulter [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#).

## Déploiement de l'exemple d'application

Votre environnement est maintenant prêt et vous pouvez déployer l'exemple d'application dans celui-ci, puis l'exécuter.

#### Note

Téléchargez le bundle de fichiers source depuis GitHub si vous ne l'avez pas déjà fait : [eb-node-express-sample-v1.1.zip](#).

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Le site collecte les informations de contact des utilisateurs et stocke les données dans une table DynamoDB. Pour ajouter une entrée, choisissez Sign up today (Inscrivez-vous aujourd'hui), entrez un nom et une adresse e-mail et choisissez Sign Up! (Inscrivez-vous). L'application web écrit le contenu du formulaire dans la table et déclenche une notification Amazon SNS par e-mail.

The screenshot shows a website with the title "A New Startup" in the top left. In the top right, there are three buttons: "Home" (dark blue), "About" (light blue), and "Blog" (light blue). The main content area features a large, bold, dark blue text "The next big thing is coming...". Below it, a message reads: "We're pretty thrilled to unveil our latest creation. Sign up below to be notified when we officially launch!". A prominent green button with white text says "Sign up today". At the bottom left, there is a copyright notice: "© A New Startup 2013".

Pour le moment, la rubrique Amazon SNS est configurée avec un e-mail à espace réservé pour les notifications. Vous allez mettre à jour la configuration prochainement, mais en attendant, vous pouvez vérifier la table DynamoDB et la rubrique Amazon SNS dans Management Console AWS Management Console.

#### Pour afficher la table

1. Ouvrez la [page Tables \(Tables\)](#) dans la console DynamoDB.
2. Recherchez la table créée par l'application. Le nom commence par awseb et contient StartupSignupsTable.
3. Sélectionnez la table, choisissez Items (Éléments), puis Start search (Commencer la recherche) afin d'afficher tous les éléments de la table.

La table contient une entrée pour chaque adresse e-mail fournie sur le site d'inscription. En plus d'écrire les données dans la table, l'application envoie un message à une rubrique Amazon SNS qui compte deux abonnements. Le premier abonnement est pour les notifications par e-mail qui vous sont adressées et l'autre pour une file d'attente Amazon Simple Queue Service qu'une application de travail peut lire afin de traiter les demandes et d'envoyer des e-mails aux clients intéressés.

#### Pour afficher la rubrique

1. Ouvrez la [page Topics \(Rubriques\)](#) dans la console Amazon SNS.
2. Recherchez la rubrique créée par l'application. Le nom commence par awseb et contient NewSignupTopic.
3. Choisissez la rubrique pour afficher ses abonnements.

L'application ([app.js](#)) définit deux routes. Le chemin d'accès racine (/) retourne une page web rendue à partir d'un modèle EJS (Embedded JavaScript) avec un formulaire que l'utilisateur remplit pour enregistrer ses nom et adresse e-mail. La soumission du formulaire envoie une demande POST avec les données du formulaire vers l'acheminement /signup, qui écrit une entrée dans la table DynamoDB et publie un message dans la rubrique Amazon SNS pour informer le propriétaire de l'inscription.

L'exemple d'application inclut des [fichiers de configuration \(p. 737\)](#) qui créent la table DynamoDB, la rubrique Amazon SNS et la file d'attente Amazon SQS utilisées par l'application. Cela vous permet de créer un nouvel environnement et de tester la fonctionnalité immédiatement, mais en revanche la table DynamoDB est liée à l'environnement. Pour un environnement de production, vous devez créer la table DynamoDB en dehors de l'environnement afin d'éviter de la perdre lorsque vous arrêtez l'environnement ou que vous mettez sa configuration à jour.

## Créez une table DynamoDB

Pour utiliser une table DynamoDB externe avec une application exécutée dans Elastic Beanstalk, créez d'abord une table dans DynamoDB. Lorsque vous créez une table en dehors d'Elastic Beanstalk, elle est complètement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk et ne sera pas arrêtée par Elastic Beanstalk.

Créez une table avec les paramètres suivants :

- Nom de la table – **nodejs-tutorial**
- Clé primaire – **email**
- Type de clé primaire – Chaîne

Pour créer une table DynamoDB

1. Ouvrez la [page Tables](#) dans la console de gestion DynamoDB.

2. Choisissez Create table (Créer une table).
3. Entrez un nom de table et une clé primaire.
4. Choisissez le type de clé primaire.
5. Sélectionnez Créer.

## Mise à jour des fichiers de configuration de l'application

Mettez à jour les [fichiers de configuration \(p. 737\)](#) dans la source de l'application afin d'utiliser la table nodejs-tutorial au lieu d'en créer une nouvelle.

Pour mettre à jour l'exemple d'application pour l'utilisation en production

1. Extrayez les fichiers de projet du bundle de fichiers source :

```
~$ mkdir nodejs-tutorial
~$ cd nodejs-tutorial
~/nodejs-tutorial$ unzip ~/Downloads/eb-node-express-sample-v1.0.zip
```

2. Ouvrez .ebextensions/options.config et modifiez les valeurs des paramètres suivants :
  - NewSignupEmail – Votre adresse e-mail.
  - STARTUP\_SIGNUP\_TABLE – nodejs-tutorial

Example .ebextensions/options.config

```
option_settings:
  aws:elasticbeanstalk:customoption:
    NewSignupEmail: you@example.com
  aws:elasticbeanstalk:application:environment:
    THEME: "flatly"
    AWS_REGION: `{"Ref" : "AWS::Region"}`
    STARTUP_SIGNUP_TABLE: nodejs-tutorial
    NEW_SIGNUP_TOPIC: `{"Ref" : "NewSignupTopic"}`
  aws:elasticbeanstalk:container:nodejs:
    ProxyServer: nginx
  aws:elasticbeanstalk:container:nodejs:staticfiles:
    /static: /static
  aws:autoscaling:asg:
    Cooldown: "120"
  aws:autoscaling:trigger:
    Unit: "Percent"
    Period: "1"
    BreachDuration: "2"
    UpperThreshold: "75"
    LowerThreshold: "30"
    MeasureName: "CPUUtilization"
```

Cela configure l'application de manière à utiliser la table nodejs-tutorial au lieu de celle créée par .ebextensions/create-dynamodb-table.config, et définit l'adresse e-mail que la rubrique Amazon SNS utilise pour les notifications.

3. Supprimez .ebextensions/create-dynamodb-table.config.

```
~/nodejs-tutorial$ rm .ebextensions/create-dynamodb-table.config
```

La prochaine fois que vous déployez l'application, la table créée par ce fichier de configuration sera supprimée.

4. Créez un bundle de fichiers source à partir du code modifié.

```
~/nodejs-tutorial$ zip nodejs-tutorial.zip -r * .[^.]*  
adding: LICENSE (deflated 65%)  
adding: README.md (deflated 56%)  
adding: app.js (deflated 63%)  
adding: iam_policy.json (deflated 47%)  
adding: misc/ (stored 0%)  
adding: misc/theme-flow.png (deflated 1%)  
adding: npm-shrinkwrap.json (deflated 87%)  
adding: package.json (deflated 40%)  
adding: static/ (stored 0%)  
adding: static/bootstrap/ (stored 0%)  
adding: static/bootstrap/css/ (stored 0%)  
adding: static/bootstrap/css/jumbotron-narrow.css (deflated 59%)  
adding: static/bootstrap/css/theme/ (stored 0%)  
adding: static/bootstrap/css/theme/united/ (stored 0%)  
adding: static/bootstrap/css/theme/united/bootstrap.css (deflated 86%)  
adding: static/bootstrap/css/theme/amelia/ (stored 0%)  
adding: static/bootstrap/css/theme/amelia/bootstrap.css (deflated 86%)  
adding: static/bootstrap/css/theme/slate/ (stored 0%)  
adding: static/bootstrap/css/theme/slate/bootstrap.css (deflated 87%)  
adding: static/bootstrap/css/theme/default/ (stored 0%)  
adding: static/bootstrap/css/theme/default/bootstrap.css (deflated 86%)  
adding: static/bootstrap/css/theme/flatly/ (stored 0%)  
adding: static/bootstrap/css/theme/flatly/bootstrap.css (deflated 86%)  
adding: static/bootstrap/LICENSE (deflated 65%)  
adding: static/bootstrap/js/ (stored 0%)  
adding: static/bootstrap/js/bootstrap.min.js (deflated 74%)  
adding: static/bootstrap/fonts/ (stored 0%)  
adding: static/bootstrap/fonts/glyphicons-halflings-regular.eot (deflated 1%)  
adding: static/bootstrap/fonts/glyphicons-halflings-regular.svg (deflated 73%)  
adding: static/bootstrap/fonts/glyphicons-halflings-regular.woff (deflated 1%)  
adding: static/bootstrap/fonts/glyphicons-halflings-regular.ttf (deflated 44%)  
adding: static/jquery/ (stored 0%)  
adding: static/jquery/jquery-1.11.3.min.js (deflated 65%)  
adding: static/jquery/MIT-LICENSE.txt (deflated 41%)  
adding: views/ (stored 0%)  
adding: views/index.ejs (deflated 67%)  
adding: .ebextensions/ (stored 0%)  
adding: .ebextensions/options.config (deflated 47%)  
adding: .ebextensions/create-sns-topic.config (deflated 56%)
```

Déployez le bundle de fichiers source nodejs-tutorial.zip dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).

6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Lors du déploiement, Elastic Beanstalk met à jour la configuration de la rubrique Amazon SNS et supprime la table DynamoDB créée lorsque vous avez déployé la première version de l'application.

Désormais, lorsque vous résiliez l'environnement, la table nodejs-tutorial n'est pas supprimée. Cela vous permet d'effectuer des déploiements bleu/vert, de modifier les fichiers de configuration ou d'arrêter votre site web sans risque de perte de données.

Ouvrez votre site dans un navigateur et vérifiez que le formulaire fonctionne comme prévu. Créez quelques entrées et vérifiez la table dans la console DynamoDB.

Pour afficher la table

1. Ouvrez la [page Tables \(Tables\)](#) dans la console DynamoDB.
2. Recherchez la table nodejs-tutorial.
3. Sélectionnez la table, choisissez Items (Éléments), puis Start search (Commencer la recherche) afin d'afficher tous les éléments de la table.

Vous pouvez également voir qu'Elastic Beanstalk a supprimé la table créée précédemment.

## Configuration de votre environnement pour une haute disponibilité

Enfin, configurez le groupe Auto Scaling de votre environnement avec un nombre minimum d'instances plus élevé. Exécutez au moins deux instances en permanence afin d'empêcher que les serveurs web de votre environnement constituent un point de défaillance unique et pour vous permettre de déployer des modifications sans mettre votre site hors service.

Pour configurer le groupe Auto Scaling de votre environnement pour une haute disponibilité

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling) définissez les Min instances (Instances min.) sur **2**.
6. Choisissez Apply.

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Vous pouvez également supprimer les tables DynamoDB externes que vous avez créées.

Pour supprimer une table DynamoDB

1. Ouvrez la [page Tables \(Tables\)](#) dans la console DynamoDB.
2. Sélectionnez une table.
3. Choisissez Actions, puis Delete table (Supprimer la table).
4. Sélectionnez Supprimer.

## Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez probablement gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

L'exemple d'application utilise des fichiers de configuration pour configurer les paramètres du logiciel et créer des ressources AWS dans le cadre de votre environnement. Pour de plus amples informations sur les fichiers de configuration et leur utilisation, veuillez consulter [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

L'exemple d'application de ce didacticiel utilise l'infrastructure web Express pour Node.js. Pour plus d'informations sur Express, consultez la documentation officielle à l'adresse [expressjs.com](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, [configurez un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activez HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Node.js

Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être associée à votre environnement et gérée par Elastic Beanstalk, ou créée et gérée en externe.

Si vous utilisez Amazon RDS pour la première fois, [ajoutez une instance de base de données \(p. 288\)](#) à un environnement de test avec la console de gestion Elastic Beanstalk et assurez-vous que votre application peut s'y connecter.

Pour vous connecter à une base de données, [ajoutez le pilote \(p. 289\)](#) à votre application, chargez-le dans votre code et [créez un objet de connexion \(p. 289\)](#) avec les propriétés d'environnement fournies par Elastic Beanstalk. La configuration et le code de connexion varient selon le moteur de base de données et l'infrastructure que vous utilisez.

#### Sections

- [Ajout d'une instance de base de données à votre environnement \(p. 288\)](#)
- [Téléchargement d'un pilote \(p. 289\)](#)
- [Connexion à une base de données \(p. 289\)](#)

## Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Choisissez Apply.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).

Nom de la propriété	Description	Valeur de la propriété
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour de plus amples informations sur la configuration d'une instance de base de données interne, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

## Téléchargement d'un pilote

Ajoutez le pilote de base de données au fichier [package.json \(p. 260\)](#) de votre projet sous `dependencies`.

Example `package.json` – Express avec MySQL

```
{  
  "name": "my-app",  
  "version": "0.0.1",  
  "private": true,  
  "dependencies": {  
    "ejs": "latest",  
    "aws-sdk": "latest",  
    "express": "latest",  
    "body-parser": "latest",  
    "mysql": "latest"  
  },  
  "scripts": {  
    "start": "node app.js"  
  }  
}
```

Packages de pilotes courants pour Node.js

- MySQL – [mysql](#)
- PostgreSQL – [node-postgres](#)
- SQL Server – [node-mssql](#)
- Oracle – [node-oracledb](#)

## Connexion à une base de données

Elastic Beanstalk fournit des informations de connexion pour les instances de base de données attachées dans les propriétés de l'environnement. Utilisez `process.env.VARIABLE` pour lire les propriétés et configurer une connexion de base de données.

Example `app.js` – Connexion à une base de données MySQL

```
var mysql = require('mysql');  
  
var connection = mysql.createConnection({  
  host      : process.env.RDS_HOSTNAME,  
  user      : process.env.RDS_USERNAME,
```

```
    password : process.env.RDS_PASSWORD,
    port      : process.env.RDS_PORT
});

connection.connect(function(err) {
  if (err) {
    console.error('Database connection failed: ' + err.stack);
    return;
  }

  console.log('Connected to database.');
});

connection.end();
```

Pour plus d'informations sur la construction d'une chaîne de connexion à l'aide de node-mysql, consultez [npmjs.org/package/mysql](https://npmjs.org/package/mysql).

## Resources

Voici plusieurs ressources vous permettant d'obtenir une aide supplémentaire lors du développement de vos applications Node.js :

Ressource	Description
<a href="#">GitHub</a>	Installez le kit SDK AWS pour Node.js à l'aide de GitHub.
<a href="#">Forum de développement Node.js</a>	Posez vos questions et obtenez des commentaires.
<a href="#">Kit de développement logiciel (SDK) AWS pour Node.js (version préliminaire destinée aux développeurs)</a>	Guichet unique pour l'exemple de code, la documentation, les outils et les ressources supplémentaires.

## Création et déploiement d'applications PHP sur Elastic Beanstalk

AWS Elastic Beanstalk pour PHP permet de déployer, de gérer et de mettre à l'échelle facilement vos applications web PHP à l'aide d'Amazon Web Services. Elastic Beanstalk pour PHP est accessible à toute personne développant ou hébergeant une application web à l'aide de PHP. Cette section fournit des instructions permettant de déployer votre application web PHP dans Elastic Beanstalk. Vous pouvez déployer votre application en quelques minutes à l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI) ou de l'Elastic Beanstalk Management Console. Elle inclut également des procédures détaillées pour des infrastructures couramment utilisées, telles que CakePHP et Symfony.

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

Si vous avez besoin d'aide sur le développement d'une application PHP , vous pouvez en trouver à plusieurs endroits :

Ressource	Description
<a href="#">GitHub</a>	Installez le kit SDK AWS pour PHP à l'aide de GitHub.

Ressource	Description
Forum de développement PHP	Posez vos questions et obtenez des commentaires.
Centre pour développeurs PHP	Guichet unique pour l'exemple de code, la documentation, les outils et les ressources supplémentaires.
FAQ sur le kit SDK AWS pour PHP	Obtenez des réponses aux questions fréquentes.

#### Rubriques

- [Configuration de votre environnement de développement PHP \(p. 291\)](#)
- [Utilisation de la plateforme PHP Elastic Beanstalk \(p. 293\)](#)
- [Déploiement d'une application Laravel sur Elastic Beanstalk \(p. 298\)](#)
- [Déploiement d'une application CakePHP sur Elastic Beanstalk \(p. 305\)](#)
- [Déploiement d'une application Symfony sur Elastic Beanstalk \(p. 311\)](#)
- [Déploiement d'une application PHP haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk \(p. 316\)](#)
- [Déploiement d'un site Web WordPress haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk \(p. 325\)](#)
- [Déploiement d'un site web Drupal haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk \(p. 339\)](#)
- [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application PHP \(p. 353\)](#)

## Configuration de votre environnement de développement PHP

Configurez un environnement de développement PHP pour tester votre application localement avant de la déployer dans AWS Elastic Beanstalk. Cette rubrique décrit les étapes de configuration de l'environnement de développement et des liens vers les pages d'installation pour des outils utiles.

Pour accéder aux outils et aux étapes de configuration courants qui s'appliquent à toutes les langues, veuillez consulter [Configuration de votre machine de développement \(p. 1024\)](#).

#### Sections

- [Installation de PHP \(p. 291\)](#)
- [Installation de Composer \(p. 292\)](#)
- [Installation du kit SDK AWS pour PHP \(p. 293\)](#)
- [Installation d'un IDE ou d'un éditeur de texte \(p. 293\)](#)

## Installation de PHP

Installez PHP et certaines extensions courantes. Si vous n'avez pas de préférence, téléchargez la dernière version. En fonction de votre plateforme et de votre gestionnaire de package disponible, les étapes varieront.

Sur Amazon Linux, utilisez yum :

```
$ sudo yum install php
$ sudo yum install php-mbstring
```

```
$ sudo yum install php-intl
```

#### Note

Pour obtenir les versions de package PHP spécifiques correspondant à votre [version de plateforme PHP](#) Elastic Beanstalk, utilisez la commande `yum search php` pour trouver les versions de packages disponibles, telles que `php72`, `php72-mbstring` et `php72-intl`. Utilisez ensuite `sudo yum install package` pour les installer.

Sur Ubuntu, utilisez apt :

```
$ sudo apt install php-all-dev
$ sudo apt install php-intl
$ sudo apt install php-mbstring
```

Sur OSX, utilisez brew :

```
$ brew install php
$ brew install php-intl
```

#### Note

Pour obtenir les versions de package PHP spécifiques correspondant à votre [version de plateforme PHP](#) Elastic Beanstalk, veuillez consulter [Homebrew Formulae](#) pour connaître les versions PHP disponibles (par exemple `php@7.2`). Utilisez ensuite `brew install package` pour les installer.

En fonction de la version, `php-intl` peut être inclus dans le package PHP principal et ne pas exister comme package autonome.

Sous Windows 10, [installez Windows Subsystem pour Linux](#) afin d'obtenir Ubuntu et d'installer PHP avec apt. Pour les versions antérieures, visitez la page de téléchargement à l'adresse [windows.php.net](#) pour obtenir PHP et lisez [cette page](#) pour plus d'informations sur les extensions.

Après avoir installé PHP, rouvrez votre terminal et exécutez `php --version` afin de garantir que la nouvelle version a été installée et est celle par défaut.

## Installation de Composer

Composer est un outil de gestion des dépendances pour PHP. Vous pouvez l'utiliser pour installer des bibliothèques, suivre les dépendances de votre application, et générer des projets pour les infrastructures PHP populaires.

Installez Composer avec le script PHP de [getcomposer.org](#).

```
$ curl -s https://getcomposer.org/installer | php
```

Le programme d'installation génère un fichier PHAR dans le répertoire en cours. Déplacez ce fichier dans un emplacement de votre environnement PATH afin de pouvoir l'utiliser en tant qu'exécutable.

```
$ mv composer.phar ~/.local/bin/composer
```

Installez les bibliothèques avec la commande `require`.

```
$ composer require twig/twig
```

Composer ajoute des bibliothèques que vous installez localement dans le fichier [composer.json de votre projet \(p. 296\)](#). Lorsque vous déployez le code de votre projet, Elastic Beanstalk utilise Composer pour installer les bibliothèques répertoriées dans ce fichier sur les instances d'application de votre environnement.

Si vous rencontrez des problèmes lors de l'installation de Composer, consultez la [documentation de Composer](#).

## Installation du kit SDK AWS pour PHP

Si vous devez gérer les ressources AWS à partir de votre application, installez AWS SDK for PHP. Par exemple, avec le kit SDK pour PHP, vous pouvez utiliser Amazon DynamoDB (DynamoDB) pour stocker les informations relatives à l'utilisateur et à la session sans créer de base de données relationnelle.

Installez le kit SDK pour PHP avec Composer.

```
$ composer require aws/aws-sdk-php
```

Consultez la [page d'accueil AWS SDK for PHP](#) pour plus d'informations et pour connaître les instructions d'installation.

## Installation d'un IDE ou d'un éditeur de texte

Les environnements de développement intégré (IDE) offrent un large éventail de fonctionnalités qui facilitent le développement d'applications. Si vous n'avez pas utilisé un PHP pour le développement Java, testez Eclipse et PHPStorm puis évaluez voir ce qui vous convient le mieux.

- [Installez Eclipse](#)
- [Installez PhpStorm](#)

### Note

Un IDE peut ajouter des fichiers dans votre dossier de projet que vous pouvez ne pas souhaiter engager sur le contrôle de code source. Pour empêcher la validation de ces fichiers de contrôle de code source, utilisez `.gitignore` ou l'équivalent de votre outil de contrôle de source.

Si vous souhaitez simplement commencer le codage et que vous n'avez pas besoin de toutes les fonctionnalités d'un IDE, pensez à [installer Sublime Text](#).

## Utilisation de la plateforme PHP Elastic Beanstalk

### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [called "Mise à niveau vers Amazon Linux 2" \(p. 508\)](#).

AWS Elastic Beanstalk prend en charge un certain nombre de plateformes pour différentes versions du langage de programmation PHP. Ces plateformes prennent en charge des applications web PHP qui peuvent s'exécuter seules ou sous Composer. Découvrez-en plus sur [PHP](#) dans le document Plateformes AWS Elastic Beanstalk.

Elastic Beanstalk fournit des [options de configuration \(p. 658\)](#) que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute sur des instances EC2 dans votre environnement Elastic Beanstalk. Vous pouvez configurer les variables d'environnement requises par votre application, activer la rotation des journaux sur

Amazon S3, mapper les dossiers de la source de votre application contenant les fichiers statiques avec les chemins desservis par le serveur proxy, et définir les paramètres d'initialisation PHP communs.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour [modifier la configuration d'un environnement en cours d'exécution \(p. 671\)](#). Pour éviter de perdre la configuration de votre environnement en le résilient, vous pouvez utiliser des [configurations enregistrées \(p. 779\)](#) pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des [fichiers de configuration \(p. 737\)](#). Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Si vous utilisez Composer, vous pouvez [inclure un fichier composer.json \(p. 296\)](#) dans votre bundle source afin d'installer des packages pendant le déploiement.

Pour des paramètres avancés PHP et de configuration PHP non fournis en tant qu'options de configuration, vous pouvez [utiliser des fichiers de configuration pour fournir un fichier INI \(p. 298\)](#) qui peut étendre et remplacer les paramètres par défaut appliqués par Elastic Beanstalk, ou installer des extensions supplémentaires.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour de plus amples informations sur la priorité et les autres méthodes de modification des paramètres, veuillez consulter [Options de configuration \(p. 658\)](#).

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter [the section called "Extension des plateformes Linux" \(p. 34\)](#).

## Configuration de votre environnement PHP

Vous pouvez utiliser la console Elastic Beanstalk pour activer la rotation de journal sur Amazon S3, configurer des variables que votre application peut lire depuis l'environnement et modifier des paramètres PHP.

Pour configurer votre environnement PHP dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

## Paramètres PHP

- Proxy server (Serveur proxy) – Serveur proxy à utiliser sur vos instances d'environnement. Le serveur nginx est utilisé par défaut.
- Racine du document – Dossier qui contient la page par défaut de votre site. Si votre page d'accueil n'est pas à la racine de votre groupe source, spécifiez le dossier qui la contient par rapport aux chemin d'accès racine. Par exemple, /public si la page d'accueil est dans un dossier nommé public.

- Limite de mémoire – Volume maximum de mémoire qu'un script est autorisé à allouer. Par exemple, 512M.
- Compression de sortie Zlib – Définissez sur On pour compresser les réponses.
- Permettre de faire un fopen d'une URL – Définissez cette option sur Off pour empêcher les scripts de télécharger des fichiers à partir d'emplacements distants.
- Afficher les erreurs – Définissez cette option sur On pour afficher les messages d'erreur internes à des fins de débogage.
- Durée d'exécution maximum – La durée maximale en secondes pendant laquelle un script est autorisé à s'exécuter avant que l'environnement ne l'arrête.

## Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

## Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour de plus amples informations sur la configuration des fichiers statiques à l'aide de la console Elastic Beanstalk, veuillez consulter [the section called "Fichiers statiques" \(p. 790\)](#).

## Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Ces paramètres sont passés en tant que paires clé-valeur à l'application.

Votre code d'application peut accéder aux propriétés de l'environnement à l'aide de `$_SERVER` ou à la fonction `get_cfg_var`.

```
$endpoint = $_SERVER['API_ENDPOINT'];
```

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Espace de noms aws:elasticbeanstalk:container:php:phpini

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

Vous pouvez utiliser l'espace de noms `aws:elasticbeanstalk:environment:proxy` pour choisir le serveur proxy de l'environnement.

Vous pouvez utiliser l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles` pour configurer le proxy d'environnement afin de traiter les fichiers statiques. Vous définissez les mappages des chemins virtuels vers les répertoires d'applications.

La plateforme PHP définit des options dans l'espace de noms `aws:elasticbeanstalk:container:php:phpini`, dont une qui n'est pas disponible dans la console Elastic Beanstalk. `composer_options` définit les options personnalisées à utiliser lors de l'installation de dépendances à l'aide Composer via `composer.phar install`. Pour plus d'informations et pour connaître les options disponibles, consultez <http://getcomposer.org/doc/03-cli.md#install>.

L'exemple de [fichier de configuration \(p. 737\)](#) suivant spécifie une option de fichiers statiques qui mappe un répertoire nommé `staticimages` au chemin `/images`, d'accès et affiche les paramètres de chacune des options disponibles dans l'espace de noms `aws:elasticbeanstalk:container:php:phpini`:

Example `.ebextensions/php-settings.config`

```
option_settings:
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /images: staticimages
  aws:elasticbeanstalk:container:php:phpini:
    document_root: /public
    memory_limit: 128M
    zlib.output_compression: "Off"
    allow_url_fopen: "On"
    display_errors: "Off"
    max_execution_time: 60
    composer_options: vendor/package
```

#### Note

L'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles` n'est pas défini sur les branches PHP de l'AMI Amazon Linux (précédant Amazon Linux 2).

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Installation des dépendances de votre application

Votre application peut avoir des dépendances sur d'autres packages PHP. Vous pouvez configurer votre application pour installer ces dépendances sur les instances Amazon Elastic Compute Cloud (Amazon EC2) de l'environnement. Vous pouvez également inclure les dépendances de votre application dans le bundle source et les déployer avec l'application. La section ci-après décrit chacune de ces méthodes.

### Utilisation d'un fichier Composer pour installer des dépendances sur les instances

Utilisez un fichier `composer.json` à la racine de votre source de projet afin d'utiliser Composer pour installer les packages dont votre application a besoin sur les instances Amazon EC2 de votre environnement.

Example `composer.json`

```
{
  "require": {
    "monolog/monolog": "1.0.*"
}
```

}

Lorsqu'un fichier `composer.json` est présent, Elastic Beanstalk exécute `composer.phar install` pour installer les dépendances. Vous pouvez ajouter des options à joindre à la commande en définissant l'option [composer\\_options \(p. 295\)](#) dans l'espace de noms `aws:elasticbeanstalk:container:php:phpini`.

## Inclusion des dépendances dans la solution groupée source

Si votre application possède un grand nombre de dépendances, leur installation peut prendre beaucoup de temps. Il peut en résulter un nombre accru d'opérations de déploiement et de mise à l'échelle, car les dépendances sont installées sur chaque nouvelle instance.

Pour éviter d'accroître le temps de déploiement, utilisez Composer dans votre environnement de développement afin de résoudre les dépendances et de les installer dans le dossier `vendor`.

Pour inclure des dépendances dans le bundle source de votre application

1. Exécutez la commande suivante :

```
% composer install
```

2. Incluez le dossier `vendor` généré dans la racine du bundle source de votre application.

Lorsque Elastic Beanstalk trouve un dossier `vendor` sur l'instance, il ignore le fichier `composer.json` (même s'il existe). Votre application utilise alors les dépendances issues du dossier `vendor`.

## Mise à jour de Composer

Vous pouvez être amené à mettre à jour Composer si une erreur s'affiche lorsque vous tentez d'installer des packages avec un fichier Composer, ou si vous ne parvenez pas à utiliser la dernière version de la plateforme. Entre les mises à jour de plateforme, vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) afin de mettre à jour Composer sur les instances de votre environnement.

Example `.ebextensions/composer.config`

```
commands:
  01updateComposer:
    command: export COMPOSER_HOME=/root && /usr/bin/composer.phar self-update 1.9.3

option_settings:
  - namespace: aws:elasticbeanstalk:application:environment
    option_name: COMPOSER_HOME
    value: /root
```

Ce fichier de configuration configure Composer pour qu'il se mette à jour vers la version 1.9.3. Pour trouver la dernière version, consultez les [pages de mise à jour](#) sur GitHub.

Pour plus d'informations sur les plateformes PHP Elastic Beanstalk, ainsi que sur la version de Composer, consultez la section [Versions des plateformes PHP](#) du document Plateformes AWS Elastic Beanstalk.

### Note

Si vous n'indiquez pas le numéro de version dans la commande `composer.phar self-update`, Composer met à jour vers la dernière version disponible chaque fois que vous déployez votre code source et que de nouvelles instances sont provisionnées par Auto Scaling. Si une version de Composer est disponible qui est incompatible avec votre application, elle peut provoquer l'échec des opérations de dimensionnement et des déploiements.

## Extension de php.ini

Utilisez un fichier de configuration avec un bloc `files` pour ajouter un fichier `.ini` à `/etc/php.d/` sur les instances de votre environnement. Le fichier de configuration principal, `php.ini`, recherche des paramètres dans des fichiers dans ce dossier dans l'ordre alphabétique. De nombreuses extensions sont activées par défaut par les fichiers dans ce dossier.

Example `.ebextensions/mongo.config`

```
files:
  "/etc/php.d/99mongo.ini" :
    mode: "000755"
    owner: root
    group: root
    content: |
      extension=mongo.so
```

## Déploiement d'une application Laravel sur Elastic Beanstalk

Laravel est une infrastructure open source, modèle-vue-contrôleur (MVC) pour PHP. Ce tutoriel vous guide tout au long du processus de génération d'une application Laravel, dans le cadre d'un déploiement dans un environnement AWS Elastic Beanstalk et de sa configuration pour une connexion à une instance de base de données Amazon Relational Database Service (Amazon RDS).

### Sections

- [Prerequisites \(p. 298\)](#)
- [Lancer un environnement Elastic Beanstalk \(p. 299\)](#)
- [Installation de Laravel et génération d'un site web \(p. 300\)](#)
- [Déploiement de votre application \(p. 300\)](#)
- [Configuration des paramètres Composer \(p. 301\)](#)
- [Ajout d'une base de données à votre environnement \(p. 302\)](#)
- [Cleanup \(p. 304\)](#)
- [Étapes suivantes \(p. 305\)](#)

## Prerequisites

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

Laravel 6 nécessite PHP 7.2 ou version ultérieure. Il requiert également les extensions PHP répertoriées dans la rubrique [Prérequis du serveur](#) de la documentation officielle de Laravel. Suivez les instructions de la rubrique [Configuration de votre environnement de développement PHP \(p. 291\)](#) pour installer PHP et Composer.

Pour plus d'informations sur la prise en charge et la maintenance de Laravel, consultez la rubrique [Stratégie de prise en charge](#) de la documentation officielle de Laravel.

## Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créer une application.

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibrEUR de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.

- Équilibrer de charge – Équilibrer de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrer de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrer de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrer de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Installation de Laravel et génération d'un site web

Composer peut installer Laravel et créer un projet de travail avec une commande :

```
~$ composer create-project --prefer-dist laravel/laravel eb-laravel
```

Composer installe Laravel et ses dépendances et génère un projet par défaut.

Si vous rencontrez des problèmes lors de l'installation de Laravel, accédez à la rubrique consacrée à l'installation dans la documentation officielle : <https://laravel.com/docs/6.x>.

## Déploiement de votre application

Créez un [bundle source \(p. 415\)](#) contenant les fichiers créés par Composer. La commande suivante permet de créer une solution groupée source nommée `laravel-default.zip`. Elle exclut les fichiers du dossier `vendor`, lesquels prennent beaucoup de place et ne sont pas nécessaires pour le déploiement de votre application dans Elastic Beanstalk.

```
~/eb-laravel$ zip ..../laravel-default.zip -r * .[^.]* -x "vendor/*"
```

Téléchargez l'offre groupée source sur Elastic Beanstalk pour déployer Laravel dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Note

En vue d'optimiser davantage le bundle source, initialisez un référentiel Git et utilisez la commande [git archive \(p. 416\)](#) pour créer le bundle source. Le projet Laravel par défaut inclut un fichier `.gitignore` qui indique à Git d'exclure le dossier `vendor` et d'autres fichiers qui ne sont pas nécessaires pour le déploiement.

## Configuration des paramètres Composer

Lorsque le déploiement est terminé, cliquez sur l'URL pour ouvrir votre application Laravel dans le navigateur :

# Forbidden

You don't have permission to access / on this server.

De quoi s'agit-il ? Par défaut, Elastic Beanstalk constitue la racine de votre projet sur le chemin d'accès racine du site Web. Dans ce cas, cependant, la page par défaut (`index.php`) est un niveau en-dessous dans le dossier `public`. Vous pouvez vérifier cela en ajoutant `/public` à l'URL. Par exemple, <http://laravel.us-east-2.elasticbeanstalk.com/public>.

Pour servir l'application Laravel à la racine, utilisez la console Elastic Beanstalk afin de configurer la racine du document du site Web.

Pour configurer la racine du document de votre site Web

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Pour Racine du document, entrez `/public`.
6. Choisissez Apply.

7. Lorsque la mise à jour est terminée, cliquez sur l'URL pour rouvrir votre site dans le navigateur.

# Laravel

DOCUMENTATION

LARACASTS

NEWS

FORGE

GITHUB

Jusqu'ici, tout va bien. Vous ajoutez ensuite une base de données à votre environnement et vous configurez Laravel pour s'y connecter.

## Ajout d'une base de données à votre environnement

Lancez une instance de base de données RDS dans votre environnement Elastic Beanstalk. Vous pouvez utiliser des bases de données MySQL, SQLServer ou PostgreSQL avec Laravel sur Elastic Beanstalk. Pour cet exemple, nous utiliserons MySQL.

Pour ajouter une instance de base de données RDS à votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Pour Moteur, choisissez mysql.
6. Saisissez un nom d'utilisateur principal et un mot de passe. Elastic Beanstalk fournit ces valeurs à votre application en utilisant les propriétés d'environnement.
7. Choisissez Apply.

La création d'une instance de base de données prend environ 10 minutes. Pour plus d'informations sur les bases de données couplées à un environnement Elastic Beanstalk, consultez [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

En attendant, vous pouvez mettre à jour votre code source afin de lire les informations de connexion depuis l'environnement. Elastic Beanstalk fournit des détails de connexion en utilisant les variables d'environnement telles que `RDS_HOSTNAME` auxquelles vous pouvez accéder depuis votre application.

La configuration de la base de données Laravel est stockée dans un fichier nommé `database.php` dans le dossier `config` dans votre code de projet. Recherchez l'entrée `mysql` et modifiez les variables `host`, `database`, `username` et `password` pour lire les valeurs correspondantes à partir d'Elastic Beanstalk :

Example ~/Eb-laravel/config/database.php

```
...
'connections' => [
    'sqlite' => [
        'driver' => 'sqlite',
        'database' => env('DB_DATABASE', database_path('database.sqlite')),
        'prefix' => '',
    ],
    'mysql' => [
        'driver' => 'mysql',
        'host' => env('RDS_HOSTNAME', '127.0.0.1'),
        'port' => env('RDS_PORT', '3306'),
        'database' => env('RDS_DB_NAME', 'forge'),
        'username' => env('RDS_USERNAME', 'forge'),
        'password' => env('RDS_PASSWORD', ''),
        'unix_socket' => env('DB_SOCKET', ''),
        'charset' => 'utf8mb4',
        'collation' => 'utf8mb4_unicode_ci',
        'prefix' => '',
        'strict' => true,
        'engine' => null,
    ],
...
]
```

Pour vérifier que la connexion de base de données est configurée correctement, ajoutez le code à index.php pour vous connecter à la base de données et ajoutez du code à la réponse par défaut :

Example ~/Eb-laravel/public/index.php

```
...
if(DB::connection()->getDatabaseName())
{
    echo "Connected to database ".DB::connection()->getDatabaseName();
}
$response->send();
...
```

Lorsque l'instance DB a terminé son lancement, regroupez et déployez l'application mise à jour dans votre environnement.

Pour mettre à jour votre environnement Elastic Beanstalk

1. Créez un nouveau groupe source :

```
~/eb-laravel$ zip ..../laravel-v2-rds.zip -r * .[^.]* -x "vendor/*"
```

2. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
3. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

4. Choisissez Upload and Deploy.
5. Choisissez Browse (Parcourir) et chargez laravel-v2-rds.zip.

6. Choisissez Deploy (Déployer).

Le déploiement d'une nouvelle version de votre application prend moins d'une minute. Lorsque le déploiement est terminé, actualisez la page web à nouveau pour vérifier que la connexion de base de données a abouti :

Connected to database ebdb



DOCUMENTATION

LARACASTS

NEWS

FORGE

GITHUB

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis choisissez Delete.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Supprimer.

## Étapes suivantes

Pour plus d'informations sur Laravel, consultez le site officiel à l'adresse [laravel.com](https://laravel.com).

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

Dans ce tutoriel, vous avez utilisé la console Elastic Beanstalk pour configurer les options du composeur. Pour que cette configuration fasse partie de votre source d'application, vous pouvez utiliser un fichier de configuration comme celui qui suit.

Example .ebextensions/composer.config

```
option_settings:  
  aws:elasticbeanstalk:container:php:phpini:  
    document_root: /public
```

Pour de plus amples informations, veuillez consulter [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

L'exécution d'une instance de base de données Amazon RDS dans votre environnement Elastic Beanstalk est excellente pour le développement et les tests, mais elle lie le cycle de vie de votre base de données à votre environnement. Pour de plus amples informations sur la connexion à une base de données s'exécutant en dehors de votre environnement, consultez [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application PHP \(p. 353\)](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez configurer un nom de domaine personnalisé (p. 656) pour votre environnement et activer HTTPS (p. 792) pour des connexions sécurisées.

## Déploiement d'une application CakePHP sur Elastic Beanstalk

CakePHP est une infrastructure MVC, open source pour PHP. Ce tutoriel vous guide à travers le processus permettant la génération d'un projet CakePHP, son déploiement dans un environnement Elastic Beanstalk et sa configuration pour vous connecter à une instance de base de données Amazon RDS.

### Sections

- [Prerequisites \(p. 305\)](#)
- [Lancer un environnement Elastic Beanstalk \(p. 306\)](#)
- [Installation de CakePHP et génération d'un site web \(p. 307\)](#)
- [Déploiement de votre application \(p. 307\)](#)
- [Ajout d'une base de données à votre environnement \(p. 308\)](#)
- [Cleanup \(p. 310\)](#)
- [Étapes suivantes \(p. 311\)](#)

## Prerequisites

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command  
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

CakePHP 4 nécessite PHP 7.2 ou version ultérieure. Il requiert également les extensions PHP répertoriées dans la [documentation d'installation officielle de CakePHP](#). Suivez les instructions de la rubrique [Configuration de votre environnement de développement PHP \(p. 291\)](#) pour installer PHP et Composer.

## Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créer une application.

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des

AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibrEUR de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Installation de CakePHP et génération d'un site web

Composer peut installer CakePHP et créer un projet de travail avec une commande :

```
~$ composer create-project --prefer-dist cakephp/app eb-cake
```

Composer installe CakePHP et environ 20 dépendances et génère un projet par défaut.

Si vous rencontrez des problèmes lors de l'installation de CakePHP, consultez la rubrique consacrée à l'installation dans la documentation officielle : <http://book.cakephp.org/4.0/en/installation.html>

## Déploiement de votre application

Créez un [bundle source \(p. 415\)](#) contenant les fichiers créés par Composer. La commande suivante permet de créer une solution groupée source nommée `cake-default.zip`. Elle exclut les fichiers du dossier `vendor`, lesquels prennent beaucoup de place et ne sont pas nécessaires pour le déploiement de votre application dans Elastic Beanstalk.

```
eb-cake zip ../cake-default.zip -r * .[^.]* -x "vendor/*"
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer CakePHP dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

#### Note

En vue d'optimiser davantage le bundle source, initialisez un référentiel Git et utilisez la commande [git archive \(p. 416\)](#) pour créer le bundle source. Le projet Symfony par défaut inclut un fichier `.gitignore` qui indique à Git d'exclure le dossier `vendor` et d'autres fichiers qui ne sont pas nécessaires pour le déploiement.

Au terme du processus, cliquez sur l'URL pour ouvrir votre application CakePHP dans le navigateur.

Jusqu'ici, tout va bien. Vous ajoutez ensuite une base de données à votre environnement et vous configurez CakePHP pour s'y connecter.

## Ajout d'une base de données à votre environnement

Lancez une instance de base de données Amazon RDS dans votre environnement Elastic Beanstalk. Vous pouvez utiliser des bases de données MySQL, SQLServer ou PostgreSQL avec CakePHP sur Elastic Beanstalk. Pour cet exemple, nous utiliserons PostgreSQL.

Pour ajouter une instance de base de données Amazon RDS à votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Sous Database (Base de données), choisissez Edit (Modifier).
5. Pour Moteur de base de données, choisissez postgres.
6. Saisissez un nom d'utilisateur principal et un mot de passe. Elastic Beanstalk fournit ces valeurs à votre application en utilisant les propriétés d'environnement.

## 7. Choisissez Apply.

La création d'une instance de base de données prend environ 10 minutes. En attendant, vous pouvez mettre à jour votre code source afin de lire les informations de connexion depuis l'environnement. Elastic Beanstalk fournit des détails de connexion en utilisant les variables d'environnement telles que `RDS_HOSTNAME` auxquelles vous pouvez accéder depuis votre application.

La configuration de base de données de CakePHP est dans un fichier nommé `app.php` dans le dossier `config` dans votre code de projet. Ouvrez ce fichier et ajoutez du code qui lit les variables d'environnement de `$_SERVER` et les attribue à des variables locales. Insérez les lignes en surbrillance dans l'exemple ci-dessous, après la première ligne () : (<?php):

Example `~/Eb-cake/config/app.php`

```
<?php
if (!defined('RDS_HOSTNAME')) {
    define('RDS_HOSTNAME', $_SERVER['RDS_HOSTNAME']);
    define('RDS_USERNAME', $_SERVER['RDS_USERNAME']);
    define('RDS_PASSWORD', $_SERVER['RDS_PASSWORD']);
    define('RDS_DB_NAME', $_SERVER['RDS_DB_NAME']);
}
return [
...
]
```

La connexion de base de données est configurée plus bas dans `app.php`. Trouvez la section suivante et modifiez la configuration des sources de données par défaut avec le nom du pilote qui correspond à votre moteur de base de données (Mysql, Sqlserver ou Postgres) et configurez les variables `host`, `username`, `password` et `database` pour lire les valeurs correspondantes à partir d'Elastic Beanstalk :

Example `~/Eb-cake/config/app.php`

```
...
/**
 * Connection information used by the ORM to connect
 * to your application's datastores.
 * Drivers include Mysql Postgres Sqlite Sqlserver
 * See vendor\cakephp\cakephp\src\Database\Driver for complete list
 */
'Datasources' => [
    'default' => [
        'className' => 'Cake\Database\Connection',
        'driver' => 'Cake\Database\Driver\Postgres',
        'persistent' => false,
        'host' => RDS_HOSTNAME,
        /*
         * CakePHP will use the default DB port based on the driver selected
         * MySQL on MAMP uses port 8889, MAMP users will want to uncomment
         * the following line and set the port accordingly
         */
        //'port' => 'non_standard_port_number',
        'username' => RDS_USERNAME,
        'password' => RDS_PASSWORD,
        'database' => RDS_DB_NAME,
        /*
         * You do not need to set this flag to use full utf-8 encoding (internal
         * default since CakePHP 3.6).
         */
        //'encoding' => 'utf8mb4',
        'timezone' => 'UTC',
        'flags' => [],
        'cacheMetadata' => true,
```

```
'log' => false,  
...
```

Lorsque l'instance DB a terminé son lancement, regroupez et déployez l'application mise à jour dans votre environnement :

Pour mettre à jour votre environnement Elastic Beanstalk

1. Créez un nouveau groupe source :

```
~/eb-cake$ zip ../cake-v2-rds.zip -r * .[^.]* -x "vendor/*"
```

2. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
3. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

4. Choisissez Upload and Deploy.
5. Choisissez Browse (Parcourir) et chargez `cake-v2-rds.zip`.
6. Choisissez Deploy (Déployer).

Le déploiement d'une nouvelle version de votre application prend moins d'une minute. Lorsque le déploiement est terminé, actualisez la page web à nouveau pour vérifier que la connexion de base de données a abouti :

## Database



CakePHP is able to connect to the database.

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis choisissez Delete.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Supprimer.

## Étapes suivantes

Pour plus d'informations sur CakePHP, lisez l'ouvrage à l'adresse [book.cakephp.org](http://book.cakephp.org).

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

L'exécution d'une instance de base de données Amazon RDS dans votre environnement Elastic Beanstalk est excellente pour le développement et les tests, mais elle lie le cycle de vie de votre base de données à votre environnement. Pour de plus amples informations sur la connexion à une base de données s'exécutant en dehors de votre environnement, consultez [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application PHP \(p. 353\)](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez [configurer un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activer HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Déploiement d'une application Symfony sur Elastic Beanstalk

[Symfony](#) est une infrastructure open source destinée au développement d'applications web PHP dynamiques. Ce didacticiel vous guide tout au long du processus de génération d'une application Symfony 3.0 et de son déploiement dans un environnement AWS Elastic Beanstalk.

Sections

- [Prerequisites \(p. 312\)](#)
- [Lancer un environnement Elastic Beanstalk \(p. 312\)](#)
- [Installation de Symfony et génération d'un site web \(p. 313\)](#)
- [Déploiement de votre application \(p. 314\)](#)
- [Configuration des paramètres Composer \(p. 314\)](#)

- [Cleanup \(p. 315\)](#)
- [Étapes suivantes \(p. 315\)](#)

## Prerequisites

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command  
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

Symfony 4.4.9 nécessite PHP 7.1.3 ou version ultérieure. Il requiert également les extensions PHP répertoriées dans la rubrique [Exigences techniques](#) de la documentation d'installation officielle de Symfony. Dans le cadre de ce tutoriel, nous utilisons PHP 7.2 et la [version](#) correspondante de la plateforme Elastic Beanstalk. Suivez les instructions de la rubrique [Configuration de votre environnement de développement PHP \(p. 291\)](#) pour installer PHP et Composer.

Pour plus d'informations sur la prise en charge et la maintenance de Symfony, consultez la rubrique [Versions de Symfony](#) sur le site Web de Symfony. Pour plus d'informations sur les mises à jour liées à la prise en charge des versions de PHP pour Symfony 4.4.9, consultez la rubrique [Notes de mise à jour de Symfony 4.4.9](#) sur le site Web de Symfony.

## Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créer une application.

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques,

ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

#### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou

[mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les

[notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme [\*sous-domaine.région\*.elasticbeanstalk.com](#).

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Installation de Symfony et génération d'un site web

Composer peut installer Symfony et créer un projet de travail avec une commande :

```
~$ composer create-project symfony/website-skeleton eb-symfony
```

Composer installe Symfony et ses dépendances et génère un projet par défaut.

Si vous rencontrez des problèmes lors de l'installation de Symfony, accédez à la rubrique consacrée à l'[Installation](#) dans la documentation officielle de Symfony.

## Déploiement de votre application

Accédez au répertoire du projet.

```
~$ cd eb-symfony
```

Créez un [bundle source](#) (p. 415) contenant les fichiers créés par Composer. La commande suivante permet de créer une solution groupée source nommée `symfony-default.zip`. Elle exclut les fichiers du dossier `vendor`, lesquels prennent beaucoup de place et ne sont pas nécessaires pour le déploiement de votre application dans Elastic Beanstalk.

```
eb-symfony$ zip ../symfony-default.zip -r * .[^.]* -x "vendor/*"
```

Chargez le bundle source sur Elastic Beanstalk pour déployer Symfony dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

### Note

En vue d'optimiser davantage le bundle source, initialisez un référentiel Git et utilisez la commande [git archive](#) (p. 416) pour créer le bundle source. Le projet Symfony par défaut inclut un fichier `.gitignore` qui indique à Git d'exclure le dossier `vendor` et d'autres fichiers qui ne sont pas nécessaires pour le déploiement.

## Configuration des paramètres Composer

Au terme du déploiement, cliquez sur l'URL pour ouvrir votre application Symfony dans le navigateur.

De quoi s'agit-il ? Par défaut, Elastic Beanstalk constitue la racine de votre projet sur le chemin d'accès racine du site web. Dans ce cas, cependant, la page par défaut (`app.php`) est un niveau en-dessous dans le dossier web. Vous pouvez vérifier cela en ajoutant `/public` à l'URL. Par exemple, <http://symfony.us-east-2.elasticbeanstalk.com/public>.

Pour servir l'application Symfony dans le chemin d'accès racine, utilisez la console Elastic Beanstalk pour configurer le document racine du site web.

Pour configurer la racine du document de votre site web

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Pour Racine du document, entrez **/public**.
6. Choisissez Apply.

7. Lorsque la mise à jour est terminée, cliquez sur l'URL pour rouvrir votre site dans le navigateur.

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

## Étapes suivantes

Pour plus d'informations sur Symfony, consultez la rubrique [Qu'est-ce que Symfony ?](#) sur le site [symfony.com](#).

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

Dans ce tutoriel, vous avez utilisé la console Elastic Beanstalk pour configurer les options du compositeur. Pour que cette configuration fasse partie de votre source d'application, vous pouvez utiliser un fichier de configuration comme celui qui suit.

#### Example .ebextensions/composer.config

```
option_settings:  
  aws:elasticbeanstalk:container:php:phpini:  
    document_root: /public
```

Pour de plus amples informations, veuillez consulter [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

Symfony utilise ses propres fichiers de configuration pour configurer les connexions de base de données. Pour de plus amples informations sur la connexion à une base de données avec Symfony, veuillez consulter [Connexion à une base de données avec Symfony \(p. 356\)](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez [configurer un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activer HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Déploiement d'une application PHP haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk

Ce didacticiel vous explique comment [lancer une instance de base de données RDS \(p. 991\)](#) externe à AWS Elastic Beanstalk et comment configurer un environnement à haute disponibilité exécutant une application PHP pour vous y connecter. L'exécution d'une instance de base de données externe à Elastic Beanstalk découpe la base de données du cycle de vie de votre environnement. Cela vous permet de connecter la même base de données depuis plusieurs environnements, de remplacer une base de données par une autre ou d'effectuer un déploiement bleu/vert sans affecter votre base de données.

Ce didacticiel utilise un [exemple d'application PHP](#) qui utilise une base de données MySQL pour stocker les données de texte fournies par l'utilisateur. L'exemple d'application utilise des [fichiers de configuration \(p. 737\)](#) en vue de configurer les [paramètres PHP \(p. 295\)](#) et de créer une table dans la base de données pour l'application à utiliser. Il montre également comment utiliser un [fichier Composer \(p. 296\)](#) pour installer les packages au cours du déploiement.

### Sections

- [Prerequisites \(p. 316\)](#)
- [Lancement d'une instance de base de données dans Amazon RDS \(p. 317\)](#)
- [Créer un environnement Elastic Beanstalk \(p. 319\)](#)
- [Configuration des groupes de sécurité, des propriétés de l'environnement et de la mise à l'échelle \(p. 320\)](#)
- [Déploiement de l'exemple d'application \(p. 323\)](#)
- [Cleanup \(p. 324\)](#)
- [Étapes suivantes \(p. 325\)](#)

## Prerequisites

Avant de commencer, téléchargez le bundle source de l'exemple d'application sur GitHub : [eb-demo-php-simple-app-1.3.zip](#).

Les procédures décrites dans ce tutoriel pour les tâches Amazon Relational Database Service (Amazon RDS) supposent que vous lancez des ressources dans un [Amazon Virtual Private Cloud](#) (Amazon VPC) par défaut. Tous les nouveaux comptes incluent un VPC par défaut dans chaque région. Si vous n'avez pas de VPC par défaut, les procédures seront différentes. Pour obtenir des instructions relatives aux plateformes EC2-Classic et VPC personnalisées, consultez [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#).

## Lancement d'une instance de base de données dans Amazon RDS

Pour utiliser une base de données externe avec une application exécutée dans Elastic Beanstalk, lancez d'abord une instance de base de données avec Amazon RDS. Lorsque vous lancez une instance avec Amazon RDS, elle est totalement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk et ne sera pas résiliée ou surveillée par Elastic Beanstalk.

Utilisez la console Amazon RDS pour lancer une instance de base de données DB MySQL Multi-AZ. Le choix d'un déploiement multi-AZ garantit que votre base de données basculera et qu'elle sera toujours disponible si l'instance de base de données source se retrouve suspendue.

Pour lancer une instance DB RDS dans un VPC par défaut

1. Ouvrez la [console RDS](#).
2. Dans la panneau de navigation, choisissez Databases (Bases de données).
3. Choisissez Create database (Créer une base de données).
4. Choisissez Création standard.

### Important

Ne choisissez pas Création facile. Si vous choisissez cette option, vous ne pouvez pas configurer les paramètres nécessaires pour lancer cette base de données RDS.

5. Sous Configuration supplémentaire, pour Nom initial de la base de données, tapez **ebdb**.
6. Vérifiez les paramètres par défaut et ajustez ces paramètres en fonction de vos exigences spécifiques. Prêtez attention aux options suivantes :
  - DB instance class (Classe d'instance de base de données) : choisissez une taille d'instance avec un niveau approprié de puissance d'UC et de mémoire pour votre charge de travail.
  - Multi-AZ deployment (Déploiement multi-AZ) : pour une haute disponibilité, définissez cette option sur Create an Aurora Replica/Reader node in a different AZ (Créer un nœud de réplica/lecteur Aurora dans une autre zone de disponibilité).
  - Master username (Identifiant principal) et Master password (Mot de passe principal) : nom d'utilisateur et mot de passe de la base de données. Notez les valeurs de ces paramètres, car vous en aurez besoin par la suite.
7. Vérifiez les paramètres par défaut pour les autres options, puis cliquez sur Crée une base de données.

Ensuite, modifiez le groupe de sécurité associé à votre instance DB pour autoriser le trafic entrant sur le port approprié. Il s'agit du même groupe de sécurité que celui que vous associerez plus tard à votre environnement Elastic Beanstalk. Par conséquent, la règle que vous ajoutez accordera une autorisation d'entrée aux autres ressources de ce même groupe de sécurité.

Pour modifier les règles de trafic entrant sur le groupe de sécurité associé à votre instance RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.

4. Dans la section Connectivity (Connectivité), prenez note des Subnets (Sous-réseaux), des Security groups (Groupes de sécurité) et du Endpoint (Point de terminaison) affichés sur cette page. Ainsi, vous pourrez utiliser ces informations ultérieurement.
5. Sous Security (Sécurité), vous voyez le groupe de sécurité associé à l'instance de base de données. Ouvrez le lien pour afficher le groupe de sécurité dans la console Amazon EC2.

Connectivity	Monitoring	Logs & events	Configuration	Maintenance & backups	Tags
<b>Connectivity</b>					
<b>Endpoint &amp; port</b>		<b>Networking</b>		<b>Security</b>	
Endpoint	ebdb...us-east-1.rds.amazonaws.com	Availability zone	us-east-1b	VPC security groups	VPC security groups
ebdb...		vpc-5732152e		rds-launch-wizard-4 (active)	rds-launch-wizard-4 (active)
Port	3306	Subnet group	default	Public accessibility	Yes
		Subnets	subnet-778f5359 subnet-7cc75e73 subnet-68432522 subnet-8762b1db subnet-a7578ac0 subnet-a0e4069e	Certificate authority	rds-ca-2015
				Certificate authority	Mar 5th, 2020

6. Dans les détails du groupe de sécurité, choisissez l'onglet Inbound (Entrant).
7. Choisissez Modifier.
8. Choisissez Add Rule.
9. Pour Type, choisissez le moteur de base de données utilisé par votre application.
10. Pour Source, entrez sg- pour afficher la liste des groupes de sécurité disponibles. Choisissez le groupe de sécurité associé au groupe Auto Scaling utilisé avec votre environnement Elastic Beanstalk. Cela permet aux instances Amazon EC2 de l'environnement d'accéder à la base de données.

### Edit inbound rules

Type	Protocol	Port Range	Source	Description
MySQL/Aurora	TCP	3306	Custom	72.21.198.67/32
MySQL/Aurora	TCP	3306	Custom	sg-07ecc7ed5b2c0c099 - rds-launch-wizard-4
<input type="button" value="Add Rule"/> <p>NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic to be dropped for a very brief period of time until the new rule can be created.</p>				

## 11. Choisissez Enregistrer.

La création d'une instance DB prend environ 10 minutes. En attendant, créez votre environnement Elastic Beanstalk.

# Créer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code. Une fois l'environnement lancé, vous pouvez le configurer pour vous connecter à la base de données, puis déployer l'exemple d'application que vous avez téléchargé depuis GitHub.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créez une application.

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou mettez à jour vos plateformes manuellement ([p. 498](#)). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.

- Équilibrer de charge – Équilibrer de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrer de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrer de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrer de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme *sous-domaine.région.elasticbeanstalk.com*.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient. L'instance DB RDS que vous avez lancée se trouvant en dehors de votre environnement, vous êtes chargé de la gestion de son cycle de vie.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Configuration des groupes de sécurité, des propriétés de l'environnement et de la mise à l'échelle

Ajoutez le groupe de sécurité de votre instance DB à votre environnement en cours d'exécution. Cette procédure conduit Elastic Beanstalk à remettre en service toutes les instances de votre environnement avec le groupe de sécurité supplémentaire associé.

Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    - a. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
    - b. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

- c. Dans le panneau de navigation, choisissez Configuration.
- d. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).
- e. Sous EC2 security groups (Groupes de sécurité EC2), choisissez les groupes de sécurité à attacher aux instances, en plus du groupe de sécurité de l'instance créé par Elastic Beanstalk.

- f. Choisissez Apply.
- g. Lisez l'avertissement, puis choisissez Confirmer.
- Pour ajouter un groupe de sécurité à l'aide d'un [fichier de configuration \(p. 737\)](#), utilisez l'exemple de fichier [securitygroup-addeexisting.config](#).

Ensuite, utilisez les propriétés de l'environnement pour transmettre les informations de connexion à votre environnement. L'exemple d'application utilise un ensemble de propriétés par défaut qui correspondent à celles qui sont configurées par Elastic Beanstalk lorsque vous mettez en service une base de données dans votre environnement.

Pour configurer les propriétés d'environnement pour une instance de base de données Amazon RDS

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Dans la section Environment properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance DB RDS intégrée, utilisez les noms et valeurs suivants : Vous pouvez trouver toutes les valeurs, à l'exception de votre mot de passe, dans la [console RDS](#).

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).

Nom de la propriété	Description	Valeur de la propriété
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

## Environment Properties

The following properties are passed into the application as environment variables. [Learn more.](#)

Property Name	Property Value
RDS_DB_NAME	ebdb
RDS_HOSTNAME	webapp-db.jxccb5mpan
RDS_PORT	5432
RDS_USERNAME	webapp-admin
RDS_PASSWORD	kUj5uKxmWDMYc403

[Cancel](#)

6. Choisissez Apply.

Enfin, configurez le groupe Auto Scaling de votre environnement avec un nombre minimum d'instances plus élevé. Exécutez au moins deux instances en permanence afin d'empêcher que les serveurs web de votre environnement constituent un point de défaillance unique et pour vous permettre de déployer des modifications sans mettre votre site hors service.

Pour configurer le groupe Auto Scaling de votre environnement pour une haute disponibilité

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling) définissez les Min instances (Instances min.) sur **2**.
6. Choisissez Apply.

## Déploiement de l'exemple d'application

Votre environnement est maintenant prêt à exécuter l'exemple d'application et à se connecter à Amazon RDS. Déployez l'exemple d'application dans votre environnement.

### Note

Téléchargez le bundle source depuis GitHub si vous ne l'avez pas déjà fait : [eb-demo-php-simple-app-1.3.zip](#).

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Le site collecte les commentaires des utilisateurs et tire parti d'une base de données MySQL pour stocker les données. Pour ajouter un commentaire, cliquez sur Share Your Thought, saisissez un commentaire, puis sélectionnez Submit Your Thought. L'application web écrit le commentaire dans la base de données pour que n'importe quelle instance de l'environnement puisse le lire et qu'il ne soit pas perdu si les instances se retrouvent suspendues.

# Your Thoughts

 Share Your Thought

Elastic Beanstalk makes it easy to get up and running with web development on

— Beanstalk User

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 563\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la [console Amazon RDS](#).

2. Choisissez Bases de données.
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis choisissez Delete.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Supprimer.

## Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

L'exemple d'application utilise des fichiers de configuration pour configurer les paramètres PHP et pour créer une table dans la base de données si elle n'existe pas déjà. Vous pouvez également utiliser un fichier de configuration pour configurer les paramètres de groupe de sécurité de vos instances lors de la création de l'environnement afin d'éviter les mises à jour de configuration, qui prennent du temps. Pour plus d'informations, consultez [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

Pour le développement et les tests, vous pouvez utiliser la fonctionnalité Elastic Beanstalk permettant d'ajouter directement une instance de base de données gérée à votre environnement. Pour savoir comment configurer une base de données dans votre environnement, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

Si vous avez besoin d'une base de données hautes performances, envisagez d'utiliser [Amazon Aurora](#). Amazon Aurora est un moteur de base de données compatible MySQL qui offre des fonctionnalités de base de données commerciales à faible coût. Pour connecter votre application à une autre base de données, répétez la procédure de [configuration du groupe de sécurité \(p. 317\)](#) et [mettez à jour les propriétés d'environnement associées à RDS \(p. 320\)](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez [configurer un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activer HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Déploiement d'un site Web WordPress haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk

Ce didacticiel décrit comment [lancer une instance de base de données Amazon RDS externe \(p. 991\)](#) à AWS Elastic Beanstalk, puis comment configurer un environnement haute disponibilité exécutant un site web WordPress pour se connecter à elle. Le site web utilise Amazon Elastic File System (Amazon EFS) comme stockage partagé pour les fichiers téléchargés.

L'exécution d'une instance de base de données externe à Elastic Beanstalk découpe la base de données du cycle de vie de votre environnement. Cela vous permet de connecter la même base de données depuis plusieurs environnements, de remplacer une base de données par une autre ou d'effectuer un [déploiement bleu/vert \(p. 486\)](#) sans affecter votre base de données.

Ce didacticiel a été développé avec la version WordPress 4.9.5 et PHP 7.0.

### Note

Pour obtenir des informations à jour sur la compatibilité des versions de PHP avec les versions de WordPress, consultez [Compatibilité de PHP et versions de WordPress](#) sur le site Web de

WordPress. Vous devez vous référer à ces informations avant de passer à une nouvelle version de PHP pour vos implémentations WordPress.

#### Rubriques

- [Prerequisites \(p. 326\)](#)
- [Lancement d'une instance de base de données dans Amazon RDS \(p. 327\)](#)
- [Téléchargement de WordPress \(p. 329\)](#)
- [Lancer un environnement Elastic Beanstalk \(p. 329\)](#)
- [Configuration des groupes de sécurité et des propriétés de l'environnement \(p. 331\)](#)
- [Configuration et déploiement de votre application \(p. 333\)](#)
- [Installer WordPress \(p. 335\)](#)
- [Mise à jour des clés et des valeurs salt \(p. 335\)](#)
- [Retrait des restrictions d'accès \(p. 336\)](#)
- [Configuration de votre groupe Auto Scaling \(p. 336\)](#)
- [Mise à niveau de WordPress \(p. 337\)](#)
- [Nettoyage \(p. 337\)](#)
- [Étapes suivantes \(p. 338\)](#)

## Prerequisites

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

#### VPC par défaut

Les procédures Amazon Relational Database Service (Amazon RDS) décrites dans ce tutoriel supposent que vous lancez des ressources dans un [Amazon Virtual Private Cloud \(Amazon VPC\)](#) par défaut. Tous les nouveaux comptes incluent un VPC par défaut dans chaque région AWS. Si vous n'avez pas de VPC par défaut, les procédures seront différentes. Pour obtenir des instructions relatives aux plateformes EC2-Classic et VPC personnalisées, consultez [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#).

#### AWSRégions

L'exemple d'application utilise Amazon EFS, qui ne fonctionne que dans les régions AWS prenant en charge Amazon EFS. Pour de plus amples informations sur les régions AWS prises en charge, consultez [Points de terminaison et quotas Amazon Elastic File System](#) dans la Référence générale AWS.

## Lancement d'une instance de base de données dans Amazon RDS

Lorsque vous lancez une instance avec Amazon RDS, elle est complètement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk et ne sera pas résiliée ou surveillée par Elastic Beanstalk.

Dans les étapes suivantes, vous allez utiliser la console Amazon RDS pour :

- Lancez une base de données avec le moteur MySQL .
- Activez un déploiement Multi-AZ. Cela crée une veille dans une autre zone de disponibilité (AZ) pour fournir la redondance des données, éliminer les blocages d'I/O et minimiser les pics de latence pendant les sauvegardes du système.

Pour lancer une instance DB RDS dans un VPC par défaut

1. Ouvrez la [console RDS](#).
2. Dans la panneau de navigation, choisissez Databases (Bases de données).
3. Choisissez Create database (Créer une base de données).
4. Choisissez Création standard.

### Important

Ne choisissez pas Création facile. Si vous choisissez cette option, vous ne pouvez pas configurer les paramètres nécessaires pour lancer cette base de données RDS.

5. Sous Configuration supplémentaire, pour Nom initial de la base de données, tapez **ebdb**.
6. Vérifiez les paramètres par défaut et ajustez ces paramètres en fonction de vos exigences spécifiques. Prêtez attention aux options suivantes :
  - DB instance class (Classe d'instance de base de données) : choisissez une taille d'instance avec un niveau approprié de puissance d'UC et de mémoire pour votre charge de travail.
  - Multi-AZ deployment (Déploiement multi-AZ) : pour une haute disponibilité, définissez cette option sur Create an Aurora Replica/Reader node in a different AZ (Créer un nœud de réplica/lecteur Aurora dans une autre zone de disponibilité).
  - Master username (Identifiant principal) et Master password (Mot de passe principal) : nom d'utilisateur et mot de passe de la base de données. Notez les valeurs de ces paramètres, car vous en aurez besoin par la suite.
7. Vérifiez les paramètres par défaut pour les autres options, puis cliquez sur Crée une base de données.

Une fois votre instance de base de données créée, modifiez le groupe de sécurité qui lui est associé afin d'autoriser le trafic entrant sur le port approprié.

### Note

Il s'agit du même groupe de sécurité que celui que vous associez plus tard à votre environnement Elastic Beanstalk. Par conséquent, la règle que vous ajoutez maintenant accordera une autorisation d'entrée aux autres ressources de ce même groupe de sécurité.

Pour modifier les règles de trafic entrant sur le groupe de sécurité associé à votre instance RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.

4. Dans la section Connectivity (Connectivité), prenez note des Subnets (Sous-réseaux), des Security groups (Groupes de sécurité) et du Endpoint (Point de terminaison) affichés sur cette page. Ainsi, vous pourrez utiliser ces informations ultérieurement.
5. Sous Security (Sécurité), vous voyez le groupe de sécurité associé à l'instance de base de données. Ouvrez le lien pour afficher le groupe de sécurité dans la console Amazon EC2.

Connectivity	Monitoring	Logs & events	Configuration	Maintenance & backups	Tags
<b>Connectivity</b>					
<b>Endpoint &amp; port</b>		<b>Networking</b>		<b>Security</b>	
Endpoint	ebdb...us-east-1.rds.amazonaws.com	Availability zone	us-east-1b	VPC security groups	VPC security groups
Port	3306	VPC	vpc-5732152e	rds-launch-wizard-4 (active)	Public accessibility
		Subnet group	default		Yes
		Subnets	subnet-778f5359 subnet-7cc75e73 subnet-68432522 subnet-8762b1db subnet-a7578ac0 subnet-a0e4069e	Certificate authority	rds-ca-2015
				Certificate authority	Mar 5th, 2020

6. Dans les détails du groupe de sécurité, choisissez l'onglet Inbound (Entrant).
7. Choisissez Modifier.
8. Choisissez Add Rule.
9. Pour Type, choisissez le moteur de base de données utilisé par votre application.
10. Pour Source, entrez sg- pour afficher la liste des groupes de sécurité disponibles. Choisissez le groupe de sécurité associé au groupe Auto Scaling utilisé avec votre environnement Elastic Beanstalk. Cela permet aux instances Amazon EC2 de l'environnement d'accéder à la base de données.

### Edit inbound rules

Type	Protocol	Port Range	Source	Description
MySQL/Aurora	TCP	3306	Custom	72.21.198.67/32
MySQL/Aurora	TCP	3306	Custom	sg-07ecc7ed5b2c0c099 - rds-launch-wizard-4
<input type="button" value="Add Rule"/> <p>NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic to be dropped for a very brief period of time until the new rule can be created.</p>				

## 11. Choisissez Enregistrer.

La création d'une instance DB prend environ 10 minutes. En attendant, téléchargez WordPress et créez votre environnement Elastic Beanstalk.

## Téléchargement de WordPress

Pour préparer le déploiement de WordPress à l'aide d'AWS Elastic Beanstalk, vous devez copier les fichiers WordPress sur votre ordinateur et fournir les informations de configuration correctes.

Pour créer un projet WordPress

1. Téléchargez WordPress à partir de [wordpress.org](https://wordpress.org).

```
~$ curl https://wordpress.org/wordpress-4.9.5.tar.gz -o wordpress.tar.gz
```

2. Téléchargez les fichiers de configuration à partir de l'exemple de référentiel.

```
~$ wget https://github.com/aws-samples/eb-php-wordpress/releases/download/v1.1/eb-php-wordpress-v1.zip
```

3. Extrayez WordPress et modifiez le nom du dossier.

```
~$ tar -xvf wordpress.tar.gz
~$ mv wordpress wordpress-beanstalk
~$ cd wordpress-beanstalk
```

4. Extrayez les fichiers de configuration sur l'installation de WordPress.

```
~/wordpress-beanstalk$ unzip ..../eb-php-wordpress-v1.zip
  creating: .ebextensions/
  inflating: .ebextensions/dev.config
  inflating: .ebextensions/efs-create.config
  inflating: .ebextensions/efs-mount.config
  inflating: .ebextensions/loadbalancer-sg.config
  inflating: .ebextensions/wordpress.config
  inflating: LICENSE
  inflating: README.md
  inflating: wp-config.php
```

## Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Une fois l'environnement lancé, vous pouvez le configurer pour vous connecter à la base de données, puis déployer le code WordPress dans l'environnement.

Dans les étapes suivantes, vous allez utiliser la console Elastic Beanstalk pour :

- Créez une application Elastic Beanstalk à l'aide de la plateforme PHP gérée.
- Acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)

2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créer une application.

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes.

### Elastic Beanstalk a créé des ressources

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

#### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021. Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.

- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

L'instance Amazon RDS que vous avez lancée se trouvant en dehors de votre environnement, vous êtes chargé de la gestion de son cycle de vie.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Configuration des groupes de sécurité et des propriétés de l'environnement

Ajoutez le groupe de sécurité de votre instance DB à votre environnement en cours d'exécution. Cette procédure conduit Elastic Beanstalk à remettre en service toutes les instances de votre environnement avec le groupe de sécurité supplémentaire associé.

Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    - Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
    - Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

- Dans le panneau de navigation, choisissez Configuration.
  - Dans la catégorie de configuration Instances, choisissez Edit (Modifier).
  - Sous EC2 security groups (Groupes de sécurité EC2), choisissez les groupes de sécurité à attacher aux instances, en plus du groupe de sécurité de l'instance créé par Elastic Beanstalk.
  - Choisissez Apply.
  - Lisez l'avertissement, puis choisissez Confirmer.
- Pour ajouter un groupe de sécurité à l'aide d'un [fichier de configuration \(p. 737\)](#), utilisez l'exemple de fichier [securitygroup-addeexisting.config](#).

Ensuite, utilisez les propriétés de l'environnement pour transmettre les informations de connexion à votre environnement.

L'application WordPress utilise un ensemble de propriétés par défaut qui correspondent à celles qui sont configurées par Elastic Beanstalk lorsque vous mettez en service une base de données dans votre environnement.

Pour configurer les propriétés d'environnement pour une instance de base de données Amazon RDS

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Dans la section Environment properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance DB RDS intégrée, utilisez les noms et valeurs suivants : Vous pouvez trouver toutes les valeurs, à l'exception de votre mot de passe, dans la [console RDS](#).

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

## Environment Properties

The following properties are passed into the application as environment variables. [Learn more.](#)

Property Name	Property Value
RDS_DB_NAME	ebdb
RDS_HOSTNAME	webapp-db.jxccb5mpan
RDS_PORT	5432
RDS_USERNAME	webapp-admin
RDS_PASSWORD	kUj5uKxmWDMYc403

[Cancel](#)

6. Choisissez Apply.

## Configuration et déploiement de votre application

Vérifiez que la structure de votre dossier `wordpress-beanstalk` est correcte, comme illustré.

```
wordpress-beanstalk$ tree -al 1
.
### .ebextensions
### index.php
### LICENSE
### license.txt
### readme.html
### README.md
### wp-activate.php
### wp-admin
### wp-blog-header.php
### wp-comments-post.php
### wp-config.php
### wp-config-sample.php
### wp-content
### wp-cron.php
### wp-includes
### wp-links-opml.php
### wp-load.php
### wp-login.php
```

```
### wp-mail.php
### wp-settings.php
### wp-signup.php
### wp-trackback.php
### xmlrpc.php
```

Le fichier `wp-config.php` personnalisé du référentiel de projet utilise les variables d'environnement que vous avez définies à l'étape précédente pour configurer la connexion de base de données. Le dossier `.ebextensions` contient des fichiers de configuration qui créent des ressources supplémentaires au sein de votre environnement Elastic Beanstalk.

Les fichiers de configuration doivent être modifiés pour être compatibles avec votre compte. Remplacez les valeurs d'espace réservé dans les fichiers par les ID appropriés et créez un bundle de fichiers source.

Pour mettre à jour les fichiers de configuration et créer une solution groupée source

1. Modifiez les fichiers de configuration comme suit.

- `.ebextensions/dev.config` – Restreint l'accès à votre environnement afin de le protéger lors du processus d'installation de WordPress. Remplacez l'adresse IP de l'espace réservé vers le haut du fichier par l'adresse IP publique de l'ordinateur que vous utiliserez pour accéder au site web de votre environnement afin de terminer votre installation WordPress.

#### Note

En fonction de votre réseau, il se peut que vous ayez besoin d'utiliser un bloc d'adresses IP.

- `.ebextensions/efs-create.config` – Crée un système de fichiers EFS et des points de montage dans chaque zone de disponibilité/sous-réseau de votre VPC. Identifiez vos ID de sous-réseau et de VPC par défaut dans la console [Amazon VPC](#).

2. Créez un [bundle source \(p. 415\)](#) contenant les fichiers dans votre dossier de projet. La commande suivante permet de créer une solution groupée source nommée `wordpress-beanstalk.zip`.

```
~/eb-wordpress$ zip .../wordpress-beanstalk.zip -r *.[^.]*
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer WordPress dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

## Installer WordPress

Pour terminer l'installation de WordPress

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez l'URL d'environnement pour ouvrir votre site dans un navigateur. Vous êtes redirigé vers un assistant d'installation WordPress, car vous n'avez pas encore configuré le site.
4. Effectuez une installation standard. Le fichier `wp-config.php` est déjà présent dans le code source et configuré pour lire les informations de connexion à la base de données à partir de l'environnement. Vous ne devriez donc pas être invité à configurer la connexion.

L'installation prend environ une minute.

## Mise à jour des clés et des valeurs salt

Le fichier de configuration WordPress `wp-config.php` lit également les valeurs pour les clés et les valeurs salt à partir des propriétés de l'environnement. Actuellement, ces propriétés sont toutes définies sur `test` par le fichier `wordpress.config` dans le dossier `.ebextensions`.

La valeur salt de hachage peut être n'importe quelle valeur qui respecte les [exigences de propriété d'environnement \(p. 634\)](#), mais vous ne devez pas la stocker dans le contrôle de code source. Utilisez la console Elastic Beanstalk pour définir ces propriétés directement sur l'environnement.

Pour mettre à jour des propriétés d'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Sous Software (Logiciel), choisissez Edit (Modifier).
5. Pour `Environment properties`, modifiez les propriétés suivantes :
  - `AUTH_KEY` – Valeur choisie pour `AUTH_KEY`.
  - `SECURE_AUTH_KEY` – Valeur choisie pour `SECURE_AUTH_KEY`.
  - `LOGGED_IN_KEY` – Valeur choisie pour `LOGGED_IN_KEY`.
  - `NONCE_KEY` – Valeur choisie pour `NONCE_KEY`.
  - `AUTH_SALT` – Valeur choisie pour `AUTH_SALT`.
  - `SECURE_AUTH_SALT` – Valeur choisie pour `SECURE_AUTH_SALT`.
  - `LOGGED_IN_SALT` – Valeur choisie pour `LOGGED_IN_SALT`.
  - `NONCE_SALT` — Valeur choisie pour `NONCE_SALT`.

6. Choisissez Apply.

#### Note

La définition des propriétés sur l'environnement remplace directement les valeurs dans `wordpress.config`.

## Retrait des restrictions d'accès

L'exemple de projet inclut le fichier de configuration `loadbalancer-sg.config`. Il crée un groupe de sécurité et l'affecte à l'équilibrer de charge de l'environnement, à l'aide de l'adresse IP que vous avez configurée dans `dev.config`. Il restreint l'accès HTTP sur le port 80 aux connexions provenant de votre réseau. Sinon, un tiers pourrait se connecter éventuellement à votre site avant que vous ayez installé WordPress et configuré votre compte d'administrateur.

Maintenant que vous avez installé WordPress, supprimez le fichier de configuration pour ouvrir le site au monde.

Pour supprimer la restriction et mettre à jour votre environnement

1. Supprimez le fichier `.ebextensions/loadbalancer-sg.config` de votre répertoire de projet.

```
~/wordpress-beanstalk$ rm .ebextensions/loadbalancer-sg.config
```

2. Créez une solution groupée source.

```
~/eb-wordpress$ zip ../wordpress-beanstalk-v2.zip -r *.[^.]*
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer WordPress dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

## Configuration de votre groupe Auto Scaling

Enfin, configurez le groupe Auto Scaling de votre environnement avec un nombre minimum d'instances plus élevé. Exécutez au moins deux instances à tout moment pour éviter que les serveurs Web de votre environnement constituent un point de défaillance unique. Cela vous permet également de déployer des modifications sans mettre votre site hors service.

Pour configurer le groupe Auto Scaling de votre environnement pour une haute disponibilité

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling) définissez les Min instances (Instances min.) sur **2**.
6. Choisissez Apply.

Pour la prise en charge des chargements de contenu sur plusieurs instances, l'exemple de projet utilise Amazon EFS pour créer un système de fichiers partagé. Créez une publication sur le site et téléchargez le contenu afin de le stocker sur le système de fichiers partagé. Affichez la publication et actualisez la page plusieurs fois pour atteindre les deux instances et vérifier que le système de fichiers partagé fonctionne.

## Mise à niveau de WordPress

Pour effectuer une mise à niveau vers une nouvelle version de WordPress, sauvegardez votre site et déployez-le dans un nouvel environnement.

#### Important

N'utilisez pas la fonctionnalité de mise à jour dans WordPress ou mettez à jour vos fichiers source pour utiliser une nouvelle version. Ces deux actions peuvent entraîner des erreurs dans vos URL de publication renvoyant des erreurs 404, même si elles sont toujours dans la base de données et le système de fichiers.

Pour mettre à niveau WordPress

1. Dans la console d'administration WordPress, utilisez l'outil d'exportation pour exporter vos publications dans un fichier XML.
2. Déployez et installez la nouvelle version de WordPress dans Elastic Beanstalk en suivant les mêmes étapes que celles utilisées pour installer la version précédente. Pour éviter les temps d'arrêt, vous pouvez créer un environnement avec la nouvelle version.
3. Sur la nouvelle version, installez l'outil d'importation WordPress dans la console d'administration et utilisez-le pour importer le fichier XML contenant vos publications. Si les publications ont été créées par l'utilisateur administrateur dans l'ancienne version, affectez-les à l'utilisateur administrateur sur le nouveau site au lieu d'essayer d'importer l'utilisateur admin.
4. Si vous avez déployé la nouvelle version dans un autre environnement, effectuez un [échange CNAME \(p. 486\)](#) pour rediriger les utilisateurs de l'ancien site vers le nouveau site.

## Nettoyage

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

### Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

### Pour résilier votre instance DB RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis choisissez Delete.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Supprimer.

## Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

L'exemple d'application utilise des fichiers de configuration pour configurer les paramètres PHP et pour créer une table dans la base de données, si elle n'existe pas déjà. Vous pouvez également utiliser un fichier de configuration pour configurer les paramètres de groupe de sécurité de vos instances lors de la création de l'environnement afin d'éviter les mises à jour de configuration, qui prennent du temps. Pour plus d'informations, consultez [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

Pour le développement et les tests, vous pouvez utiliser la fonctionnalité Elastic Beanstalk permettant d'ajouter directement une instance de base de données gérée à votre environnement. Pour savoir comment configurer une base de données dans votre environnement, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

Si vous avez besoin d'une base de données hautes performances, envisagez d'utiliser [Amazon Aurora](#). Amazon Aurora est un moteur de base de données compatible MySQL qui offre des fonctionnalités de base de données commerciales à faible coût. Pour connecter votre application à une autre base de données, répétez la procédure de [configuration du groupe de sécurité \(p. 317\)](#) et [mettez à jour les propriétés d'environnement associées à RDS \(p. 320\)](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez [configurer un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activer HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Déploiement d'un site web Drupal haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk

Ce didacticiel vous guide tout au long du processus de [lancement d'une instance de base de données RDS \(p. 991\)](#) externe dans AWS Elastic Beanstalk. Il décrit ensuite comment configurer un environnement à haute disponibilité en exécutant un site web Drupal pour s'y connecter. Le site web utilise Amazon Elastic File System (Amazon EFS) comme stockage partagé pour les fichiers téléchargés. L'exécution d'une instance de base de données externe à Elastic Beanstalk découpe la base de données du cycle de vie de votre environnement et vous permet de vous connecter à la même base de données à partir de plusieurs environnements, de changer de base de données ou d'exécuter un déploiement bleu/vert sans entraîner de répercussion sur votre base de données.

### Sections

- [Prerequisites \(p. 339\)](#)
- [Lancement d'une instance de base de données dans Amazon RDS \(p. 340\)](#)
- [Lancer un environnement Elastic Beanstalk \(p. 342\)](#)
- [Configuration des paramètres de sécurité et des propriétés de l'environnement \(p. 344\)](#)
- [Configuration et déploiement de votre application \(p. 347\)](#)
- [Installation de Drupal \(p. 349\)](#)
- [Mise à jour de la configuration de Drupal et retrait de restrictions d'accès \(p. 349\)](#)
- [Configuration de votre groupe Auto Scaling \(p. 352\)](#)
- [Cleanup \(p. 352\)](#)
- [Étapes suivantes \(p. 353\)](#)

## Prerequisites

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

Les procédures décrites dans ce tutoriel pour les tâches Amazon Relational Database Service (Amazon RDS) supposent que vous lancez des ressources dans un [Amazon Virtual Private Cloud](#) (Amazon VPC) par défaut. Tous les nouveaux comptes incluent un VPC par défaut dans chaque région. Si vous n'avez pas de VPC par défaut, les procédures seront différentes. Pour obtenir des instructions relatives aux plateformes EC2-Classic et VPC personnalisées, consultez [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#).

L'exemple d'application utilise Amazon EFS. Il fonctionne uniquement dans les régions AWS qui prennent en charge Amazon EFS. Pour de plus amples informations sur les régions AWS prises en charge, consultez [Points de terminaison et quotas Amazon Elastic File System](#) dans la Référence générale AWS.

Si la plateforme de votre environnement Elastic Beanstalk utilise PHP 7.4 ou version antérieure, nous vous recommandons d'utiliser la version 8.9.13 de Drupal pour ce tutoriel. Pour les plateformes installées avec PHP 8.0 ou version ultérieure, nous vous recommandons d'utiliser Drupal 9.1.5.

Pour plus d'informations sur les versions de Drupal et les versions de PHP qu'elles prennent en charge, consultez [Exigences relatives à PHP](#) sur le site Web de Drupal. Les versions principales recommandées par Drupal sont répertoriées sur le site <https://www.drupal.org/project/drupal>.

## Lancement d'une instance de base de données dans Amazon RDS

Pour utiliser une base de données externe avec une application exécutée dans Elastic Beanstalk, lancez d'abord une instance de base de données avec Amazon RDS. Lorsque vous lancez une instance avec Amazon RDS, elle est totalement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk et ne sera pas résiliée ou surveillée par Elastic Beanstalk.

Utilisez la console Amazon RDS pour lancer une instance de base de données DB MySQL Multi-AZ. Le choix d'un déploiement multi-AZ garantit que votre base de données basculera et qu'elle sera toujours disponible si l'instance de base de données source se retrouve hors service.

Pour lancer une instance DB RDS dans un VPC par défaut

1. Ouvrez la [console RDS](#).
2. Dans la panneau de navigation, choisissez Databases (Bases de données).
3. Choisissez Create database (Créer une base de données).
4. Choisissez Création standard.

### Important

Ne choisissez pas Création facile. Si vous choisissez cette option, vous ne pouvez pas configurer les paramètres nécessaires pour lancer cette base de données RDS.

5. Sous Configuration supplémentaire, pour Nom initial de la base de données, tapez **ebdb**.
6. Vérifiez les paramètres par défaut et ajustez ces paramètres en fonction de vos exigences spécifiques. Prêtez attention aux options suivantes :
  - DB instance class (Classe d'instance de base de données) : choisissez une taille d'instance avec un niveau approprié de puissance d'UC et de mémoire pour votre charge de travail.
  - Multi-AZ deployment (Déploiement multi-AZ) : pour une haute disponibilité, définissez cette option sur Create an Aurora Replica/Reader node in a different AZ (Créer un nœud de réplica/lecteur Aurora dans une autre zone de disponibilité).
  - Master username (Identifiant principal) et Master password (Mot de passe principal) : nom d'utilisateur et mot de passe de la base de données. Notez les valeurs de ces paramètres, car vous en aurez besoin par la suite.
7. Vérifiez les paramètres par défaut pour les autres options, puis cliquez sur Crée une base de données.

Ensuite, modifiez le groupe de sécurité associé à votre instance DB pour autoriser le trafic entrant sur le port approprié. Il s'agit du même groupe de sécurité que celui que vous associerez plus tard à votre environnement Elastic Beanstalk. Par conséquent, la règle que vous ajoutez accordera une autorisation d'entrée aux autres ressources de ce même groupe de sécurité.

Pour modifier les règles de trafic entrant sur le groupe de sécurité associé à votre instance RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.
4. Dans la section Connectivity (Connectivité), prenez note des Subnets (Sous-réseaux), des Security groups (Groupes de sécurité) et du Endpoint (Point de terminaison) affichés sur cette page. Ainsi, vous pourrez utiliser ces informations ultérieurement.
5. Sous Security (Sécurité), vous voyez le groupe de sécurité associé à l'instance de base de données. Ouvrez le lien pour afficher le groupe de sécurité dans la console Amazon EC2.

Connectivity	Monitoring	Logs & events	Configuration	Maintenance & backups	Tags
<b>Connectivity</b>					
<b>Endpoint &amp; port</b>					
Endpoint			Networking		Security
ebdb...us-east-1.rds.amazonaws.com			Availability zone		VPC security groups
Port			us-east-1b		rds-launch-wizard-4 (s
3306			VPC		( active )
			vpc-5732152e		
			Subnet group		Public accessibility
			default		Yes
			Subnets		Certificate authority
			subnet-778f5359		rds-ca-2015
			subnet-7cc75e73		Certificate authority d
			subnet-68432522		Mar 5th, 2020
			subnet-8762b1db		
			subnet-a7578ac0		
			subnet-a0e4069e		

6. Dans les détails du groupe de sécurité, choisissez l'onglet Inbound (Entrant).
7. Choisissez Modifier.
8. Choisissez Add Rule.
9. Pour Type, choisissez le moteur de base de données utilisé par votre application.
10. Pour Source, entrez **sg-** pour afficher la liste des groupes de sécurité disponibles. Choisissez le groupe de sécurité associé au groupe Auto Scaling utilisé avec votre environnement Elastic Beanstalk. Cela permet aux instances Amazon EC2 de l'environnement d'accéder à la base de données.

The screenshot shows the 'Edit inbound rules' page in the AWS Elastic Beanstalk console. There are two rows of rules. The first row has a 'Type' of 'MySQL/Aurora', 'Protocol' of 'TCP', and 'Port Range' of '3306'. The 'Source' dropdown is set to 'Custom' with the value '72.21.198.67/32'. The second row has a 'Type' of 'MySQL/Aurora', 'Protocol' of 'TCP', and 'Port Range' of '3306'. The 'Source' dropdown is set to 'Custom' with the value 'sg-07ecc7ed5b2c0c099 - rds-launch-wizard-4'. A yellow box highlights the second row's 'Source' field. A note at the bottom states: 'NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic to be dropped on that rule for a very brief period of time until the new rule can be created.'

11. Choisissez Enregistrer.

La création d'une instance DB prend environ 10 minutes. En attendant, lancez votre environnement Elastic Beanstalk.

## Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme PHP et acceptez les paramètres par défaut et l'exemple de code. Une fois l'environnement lancé, vous pouvez le configurer pour vous connecter à la base de données, puis déployer le code Drupal dans l'environnement.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créer une application.

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

## Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient. L'instance DB RDS que vous avez lancée se trouvant en dehors de votre environnement, vous êtes chargé de la gestion de son cycle de vie.

## Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Configuration des paramètres de sécurité et des propriétés de l'environnement

Ajoutez le groupe de sécurité de votre instance DB à votre environnement en cours d'exécution. Cette procédure conduit Elastic Beanstalk à remettre en service toutes les instances de votre environnement avec le groupe de sécurité supplémentaire associé.

Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    - a. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
    - b. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

- c. Dans le panneau de navigation, choisissez Configuration.
  - d. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).
  - e. Sous EC2 security groups (Groupes de sécurité EC2), choisissez les groupes de sécurité à attacher aux instances, en plus du groupe de sécurité de l'instance créé par Elastic Beanstalk.
  - f. Choisissez Apply.
  - g. Lisez l'avertissement, puis choisissez Confirmer.
- Pour ajouter un groupe de sécurité à l'aide d'un [fichier de configuration \(p. 737\)](#), utilisez l'exemple de fichier [securitygroup-addexisting.config](#).

Ensuite, utilisez les propriétés de l'environnement pour transmettre les informations de connexion à votre environnement. L'exemple d'application utilise un ensemble de propriétés par défaut qui correspondent à celles qui sont configurées par Elastic Beanstalk lorsque vous mettez en service une base de données dans votre environnement.

Pour configurer les propriétés d'environnement pour une instance de base de données Amazon RDS

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Dans la section Environment properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance DB RDS intégrée, utilisez les noms et valeurs suivants : Vous pouvez trouver toutes les valeurs, à l'exception de votre mot de passe, dans la [console RDS](#).

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

## Environment Properties

The following properties are passed into the application as environment variables. [Learn more.](#)

Property Name	Property Value
RDS_DB_NAME	ebdb
RDS_HOSTNAME	webapp-db.jxccb5mpan
RDS_PORT	5432
RDS_USERNAME	webapp-admin
RDS_PASSWORD	kUj5uKxmWDMYc403

[Cancel](#)

6. Choisissez Apply.

Une fois Drupal installé, vous devez vous connecter à l'instance avec SSH pour récupérer certains détails de configuration. Attribuez une clé SSH aux instances de votre environnement.

### Pour configurer SSH

1. Si vous n'avez pas déjà préalablement créé une paire de clés, ouvrez la page relative aux [paires de clés](#) de la console Amazon EC2 et suivez les instructions pour en créer une.
2. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
3. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

4. Dans le panneau de navigation, choisissez Configuration.
5. Sous Security (Sécurité), choisissez Edit (Modifier).
6. Pour Paire de clés EC2, choisissez votre paire de clés.
7. Choisissez Apply.

## Configuration et déploiement de votre application

Pour créer un projet Drupal pour Elastic Beanstalk, téléchargez le code source Drupal et associez-le aux fichiers dans le référentiel [aws-samples/eb-php-drupal](#) sur GitHub.

Pour créer un projet Drupal

1. Exécutez la commande suivante pour télécharger Drupal à partir de [www.drupal.org/download](http://www.drupal.org/download). Pour en savoir plus sur les téléchargements, consultez [le site Web de Drupal](#).

Si la plateforme de votre environnement Elastic Beanstalk utilise PHP 7.4 ou version antérieure, nous vous recommandons de télécharger la version 8.9.13 de Drupal pour ce tutoriel. Pour ce faire, exécutez la commande suivante.

```
~$ curl https://ftp.drupal.org/files/projects/drupal-8.9.13.tar.gz -o drupal.tar.gz
```

Si votre plateforme utilise PHP 8.0 ou version ultérieure, nous vous recommandons de télécharger Drupal 9.1.5. Pour ce faire, exécutez la commande suivante.

```
~$ curl https://ftp.drupal.org/files/projects/drupal-9.1.5.tar.gz -o drupal.tar.gz
```

Pour plus d'informations sur les versions de Drupal et les versions de PHP qu'elles prennent en charge, consultez [Exigences relatives à PHP](#) dans la documentation officielle de Drupal. Les versions principales recommandées par Drupal sont répertoriées sur [le site Web de Drupal](#).

2. Utilisez la commande suivante pour télécharger les fichiers de configuration à partir de l'exemple de référentiel :

```
~$ wget https://github.com/aws-samples/eb-php-drupal/releases/download/v1.1/eb-php-drupal-v1.zip
```

3. Extrayez Drupal et modifiez le nom du dossier.

Si vous avez téléchargé Drupal 8.9.13 :

```
~$ tar -xvf drupal.tar.gz
~$ mv drupal-8.9.13 drupal-beanstalk
~$ cd drupal-beanstalk
```

Si vous avez téléchargé Drupal 9.1.5 :

```
~$ tar -xvf drupal.tar.gz
~$ mv drupal-9.1.5 drupal-beanstalk
~$ cd drupal-beanstalk
```

4. Extrayez les fichiers de configuration sur l'installation de Drupal.

```
~/drupal-beanstalk$ unzip ../eb-php-drupal-v1.zip
creating: .ebextensions/
inflating: .ebextensions/dev.config
inflating: .ebextensions/drupal.config
inflating: .ebextensions/efs-create.config
inflating: .ebextensions/efs-filesystem.template
inflating: .ebextensions/efs-mount.config
inflating: .ebextensions/loadbalancer-sg.config
inflating: LICENSE
inflating: README.md
```

```
inflating: beanstalk-settings.php
```

Vérifiez que la structure de votre dossier drupal-beanstalk est correcte, comme illustré.

```
drupal-beanstalk$ tree -al 1
.
### autoload.php
### beanstalk-settings.php
### composer.json
### composer.lock
### core
### .csslintrc
### .ebextensions
### .ebextensions
### .editorconfig
### .eslintrc
### .eslintrc.json
### example.gitignore
### .gitattributes
### .htaccess
### .ht.router.php
### index.php
### LICENSE
### LICENSE.txt
### modules
### profiles
### README.md
### README.txt
### robots.txt
### sites
### themes
### update.php
### vendor
### web.config
```

Le fichier beanstalk-settings.php du référentiel de projet utilise les variables d'environnement que vous avez définies à l'étape précédente pour configurer la connexion de base de données. Le dossier .ebextensions contient des fichiers de configuration qui créent des ressources supplémentaires au sein de votre environnement Elastic Beanstalk.

Les fichiers de configuration doivent être modifiés pour être compatibles avec votre compte. Remplacez les valeurs d'espace réservé dans les fichiers par les ID appropriés et créez un bundle de fichiers source.

Pour mettre à jour les fichiers de configuration et créer une solution groupée source.

1. Modifiez les fichiers de configuration comme suit.
  - .ebextensions/dev.config – Restreint l'accès à votre environnement à votre adresse IP afin de le protéger lors du processus d'installation de Drupal. Remplacez l'adresse IP de l'espace réservé vers le haut du fichier par votre adresse IP publique.
  - .ebextensions/efs-create.config – Crée un système de fichiers EFS et des points de montage dans chaque zone de disponibilité/sous-réseau de votre VPC. Identifiez vos ID de sous-réseau et de VPC par défaut dans la console [Amazon VPC](#).
2. Créez un [bundle source \(p. 415\)](#) contenant les fichiers dans votre dossier de projet. La commande suivante permet de créer une solution groupée source nommée drupal-beanstalk.zip. Elle exclut les fichiers du dossier vendor, lesquels prennent beaucoup de place et ne sont pas nécessaires pour le déploiement de votre application dans Elastic Beanstalk.

```
~/eb-drupal$ zip ../drupal-beanstalk.zip -r *.[^.*] -x "vendor/*"
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer Drupal dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

## Installation de Drupal

Pour terminer l'installation de Drupal

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez l'URL d'environnement pour ouvrir votre site dans un navigateur. Vous êtes redirigé vers un assistant d'installation Drupal car le site n'a pas encore été configuré.
4. Effectuez une installation standard avec les paramètres suivants pour la base de données :
  - Nom de la base de données – Nom de la base de données qui apparaît sur la console Amazon RDS.
  - Nom d'utilisateur de base de données et mot de passe – Valeurs des champs Identifiant principal et Mot de passe principal que vous avez saisies lors de la création de votre base de données.
  - Options avancées > Hôte – Valeur du champ Point de terminaison correspondant à l'instance de base de données, affichée dans la console Amazon RDS.

L'installation prend environ une minute.

## Mise à jour de la configuration de Drupal et retrait de restrictions d'accès

Le processus d'installation de Drupal a créé un fichier nommé `settings.php` dans le dossier `sites/default` sur l'instance. Vous aurez besoin de ce fichier dans votre code source pour éviter de réinitialiser votre site lors des déploiements suivants, mais le fichier contient actuellement des codes secrets que vous ne souhaitez pas valider dans la source. Connectez-vous à l'instance d'application afin de récupérer les informations du fichier de paramètres.

Pour vous connecter à votre instance d'application avec SSH

1. Ouvrez la [page Instances](#) de la console Amazon EC2.
2. Choisissez l'instance d'application. C'est celui qui porte le nom de votre environnement Elastic Beanstalk.
3. Choisissez Connect (Connexion).
4. Suivez les instructions pour vous connecter à l'instance avec SSH. La commande ressemble à ce qui suit.

```
$ ssh -i ~/.ssh/mykey ec2-user@ec2-00-55-33-222.us-west-2.compute.amazonaws.com
```

Obtenez l'ID du répertoire de synchronisation dans la dernière ligne du fichier de paramètres.

```
[ec2-user ~]$ tail -n 1 /var/app/current/sites/default/settings.php
$config_directories['sync'] = 'sites/default/files/
config_4ccfx2spQm79p1mk5IbUq9S_FokcENO4mxyC-L18-4g_xKj_7j9ydn31kDOYOgnzMu071Tvc4Q/sync';
```

Le fichier contient également la clé de hachage actuelle des sites, mais vous pouvez ignorer la valeur actuelle et utiliser la vôtre.

Attribuez le chemin d'accès au répertoire et la clé de hachage aux propriétés de l'environnement. Le fichier de paramètres personnalisés du référentiel de projet lit ces propriétés pour configurer le site pendant le déploiement, en plus des propriétés de connexion de base de données que vous avez définies précédemment.

#### Propriétés de configuration de Drupal

- **SYNC\_DIR** – Chemin d'accès au répertoire de synchronisation.
- **HASH\_SALT** – Toute valeur de chaîne qui respecte les [exigences de propriété d'environnement \(p. 634\)](#).

Pour configurer les propriétés d'environnement dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Sous Propriétés de l'environnement, saisissez les paires clé/valeur.

6. Choisissez Apply.

Enfin l'exemple de projet inclut un fichier de configuration (`loadbalancer-sg.config`) qui crée un groupe de sécurité et l'affecte à l'équilibrer de charge de l'environnement, à l'aide de l'adresse IP que vous avez configurée dans `dev.config` afin de restreindre l'accès HTTP sur le port 80 aux connexions provenant de votre réseau. Sinon, un tiers pourrait se connecter éventuellement à votre site avant que vous ayez installé Drupal et configuré votre compte d'administrateur.

Pour mettre à jour la configuration de Drupal et retirer les restrictions d'accès

1. Supprimez le fichier `.ebextensions/loadbalancer-sg.config` de votre répertoire de projet.

```
~/drupal-beanstalk$ rm .ebextensions/loadbalancer-sg.config
```

2. Copiez le fichier `settings.php` personnalisé dans le dossier des sites.

```
~/drupal-beanstalk$ cp beanstalk-settings.php sites/default/settings.php
```

3. Créez une solution groupée source.

```
~/eb-drupal$ zip ../drupal-beanstalk-v2.zip -r * .[^.]* -x "vendor/*"
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer Drupal dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

## Configuration de votre groupe Auto Scaling

Enfin, configurez le groupe Auto Scaling de votre environnement avec un nombre minimum d'instances plus élevé. Exécutez au moins deux instances en permanence afin d'empêcher que les serveurs web de votre environnement constituent un point de défaillance unique et pour vous permettre de déployer des modifications sans mettre votre site hors service.

Pour configurer le groupe Auto Scaling de votre environnement pour une haute disponibilité

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling) définissez les Min instances (Instances min.) sur **2**.
6. Choisissez Apply.

Pour la prise en charge des chargements de contenu sur plusieurs instances, l'exemple de projet utilise Amazon Elastic File System pour créer un système de fichiers partagé. Créez une publication sur le site et téléchargez le contenu afin de le stocker sur le système de fichiers partagé. Affichez la publication et actualisez la page plusieurs fois pour atteindre les deux instances et vérifier que le système de fichiers partagé fonctionne.

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Vous pouvez également arrêter les ressources de base de données que vous avez créées hors de votre environnement Elastic Beanstalk. Lorsque vous résiliez une instance de base de données Amazon RDS, vous pouvez prendre un instantané et restaurer les données dans une autre instance ultérieurement.

Pour résilier votre instance DB RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Sélectionnez votre instance DB.
4. Choisissez Actions, puis choisissez Delete.
5. Choisissez si vous souhaitez créer un instantané, puis choisissez Supprimer.

## Étapes suivantes

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

L'exemple d'application utilise des fichiers de configuration pour configurer les paramètres PHP et pour créer une table dans la base de données si elle n'existe pas déjà. Vous pouvez également utiliser un fichier de configuration pour configurer les paramètres de groupe de sécurité de vos instances lors de la création de l'environnement afin d'éviter les mises à jour de configuration, qui prennent du temps. Pour plus d'informations, consultez [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

Pour le développement et les tests, vous pouvez utiliser la fonctionnalité Elastic Beanstalk permettant d'ajouter directement une instance de base de données gérée à votre environnement. Pour savoir comment configurer une base de données dans votre environnement, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

Si vous avez besoin d'une base de données hautes performances, envisagez d'utiliser [Amazon Aurora](#). Amazon Aurora est un moteur de base de données compatible MySQL qui offre des fonctionnalités de base de données commerciales à faible coût. Pour connecter votre application à une autre base de données, répétez la procédure de [configuration du groupe de sécurité \(p. 317\)](#) et [mettez à jour les propriétés d'environnement associées à RDS \(p. 320\)](#).

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez [configurer un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activer HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Ajout d'une instance de base de données Amazon RDS à votre environnement d'application PHP

Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être associée à votre environnement et gérée par Elastic Beanstalk, ou créée et gérée en externe.

Si vous utilisez Amazon RDS pour la première fois, [ajoutez une instance de base de données \(p. 354\)](#) à un environnement de test avec la console Elastic Beanstalk et assurez-vous que votre application peut s'y connecter.

Pour vous connecter à une base de données, [ajoutez le pilote \(p. 355\)](#) à votre application, chargez la classe de pilote dans votre code et [créez un objet de connexion \(p. 355\)](#) avec les propriétés d'environnement fournies par Elastic Beanstalk. La configuration et le code de connexion varient selon le moteur de base de données et l'infrastructure que vous utilisez.

#### Sections

- [Ajout d'une instance de base de données à votre environnement \(p. 354\)](#)
- [Téléchargement d'un pilote \(p. 355\)](#)
- [Connexion à une base de données avec un PDO ou MySQLi \(p. 355\)](#)
- [Connexion à une base de données avec Symfony \(p. 356\)](#)

## Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Choisissez Apply.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).

Nom de la propriété	Description	Valeur de la propriété
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour de plus amples informations sur la configuration d'une instance de base de données interne, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

## Téléchargement d'un pilote

Pour utiliser des objets de données PHP (PDO) pour la connexion à la base de données, installez le pilote correspondant au moteur de base de données que vous avez choisi.

- MySQL – [PDO\\_MYSQL](#)
- PostgreSQL – [PDO\\_PGSQLO](#)
- Oracle – [PDO\\_OCI](#)
- SQL Server – [PDO\\_SQLSRV](#)

Pour de plus amples informations, veuillez consulter <http://php.net/manual/en/pdo.installation.php>.

## Connexion à une base de données avec un PDO ou MySQLi

Vous pouvez utiliser `$_SERVER['VARIABLE']` afin de lire les informations de connexion depuis l'environnement.

Pour un PDO, créez un nom de source de données (DSN) à partir de l'hôte, du port et du nom. Transmettez le DSN au [constructeur du PDO](#) avec le nom d'utilisateur de base de données et le mot de passe.

Example Connexion à une base de données RDS avec PDO - MySQL

```
<?php
$dbhost = $_SERVER['RDS_HOSTNAME'];
$dbport = $_SERVER['RDS_PORT'];
$dbname = $_SERVER['RDS_DB_NAME'];
$charset = 'utf8';

$dsn = "mysql:host={$dbhost};port={$dbport};dbname={$dbname};charset={$charset}";
$username = $_SERVER['RDS_USERNAME'];
$password = $_SERVER['RDS_PASSWORD'];

$pdo = new PDO($dsn, $username, $password);
?>
```

Pour les autres pilotes, remplacez mysql par le nom de votre pilote – pgsql, oci ou sqlsrv.

Pour MySQLi, transmettez le nom d'hôte, le nom d'utilisateur, le mot de passe, le nom de base de données et le port au constructeur `mysql`.

Example Connexion à une base de données RDS avec mysqli\_connect()

```
$link = new mysqli($_SERVER['RDS_HOSTNAME'], $_SERVER['RDS_USERNAME'],  
$_SERVER['RDS_PASSWORD'], $_SERVER['RDS_DB_NAME'], $_SERVER['RDS_PORT']);
```

## Connexion à une base de données avec Symfony

Pour Symfony version 3.2 et versions ultérieures, vous pouvez utiliser `%env(PROPERTY_NAME)%` pour définir les paramètres de base de données dans un fichier de configuration en fonction des propriétés d'environnement définies par Elastic Beanstalk.

Example app/config/parameters.yml

```
parameters:  
    database_driver:    pdo_mysql  
    database_host:     '%env(RDS_HOSTNAME)%'  
    database_port:     '%env(RDS_PORT)%'  
    database_name:     '%env(RDS_DB_NAME)%'  
    database_user:     '%env(RDS_USERNAME)%'  
    database_password: '%env(RDS_PASSWORD)%'
```

Pour plus d'informations, consultez [External Parameters \(Symfony 3.4\)](#).

Pour les versions antérieures de Symfony, les variables d'environnement sont uniquement accessibles si elles commencent par `SYMFONY_`. Cela signifie que les propriétés d'environnement définies par Elastic Beanstalk ne sont pas accessibles, et vous devez définir vos propres propriétés d'environnement pour transmettre les informations de connexion à Symfony.

Pour vous connecter à une base de données avec Symfony 2, [créez une propriété d'environnement \(p. 295\)](#) pour chaque paramètre. Ensuite, utilisez `%property.name%` pour accéder à la variable transformée par Symfony dans un fichier de configuration. Par exemple, une propriété d'environnement nommée `SYMFONY__DATABASE__USER` est accessible en tant que `database.user`.

```
database_user:      "%database.user%"
```

Pour plus d'informations, consultez [External Parameters \(Symfony 2.8\)](#).

## Travail avec Python

Cette section fournit des didacticiels et des informations sur le déploiement d'applications Python à l'aide d'AWS Elastic Beanstalk.

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

### Rubriques

- [Configuration de votre environnement de développement Python \(p. 357\)](#)
- [Utilisation de la plateforme Python Elastic Beanstalk \(p. 359\)](#)
- [Déploiement d'une application Flask sur Elastic Beanstalk \(p. 365\)](#)
- [Déploiement d'une application Django sur Elastic Beanstalk \(p. 371\)](#)
- [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Python \(p. 382\)](#)
- [Outils et ressources Python \(p. 384\)](#)

# Configuration de votre environnement de développement Python

Configurez un environnement de développement Python pour tester votre application localement avant de la déployer dans AWS Elastic Beanstalk. Cette rubrique décrit les étapes de configuration de l'environnement de développement et des liens vers les pages d'installation pour des outils utiles.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command  
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

Pour accéder aux outils et aux étapes de configuration courants qui s'appliquent à toutes les langues, veuillez consulter [Configuration de votre machine de développement \(p. 1024\)](#).

## Sections

- [Installation de Python et de pip \(p. 357\)](#)
- [Utilisation d'un environnement virtuel \(p. 357\)](#)
- [Configuration d'un projet Python pour Elastic Beanstalk \(p. 358\)](#)

## Installation de Python et de pip

Pour toutes les applications Python que vous déploierez avec Elastic Beanstalk, ces conditions préalables sont courantes :

1. Une version Python correspondant à la version de la plateforme Elastic Beanstalk Python que votre application utilisera.
2. L'utilitaire `pip`, correspondant à votre version de Python. Il est utilisé pour installer et répertorier des dépendances pour votre projet, afin qu'Elastic Beanstalk sache comment mettre en place l'environnement de votre application.
3. Le package `virtualenv`. Il est utilisé pour créer un environnement utilisé pour développer et tester votre application, afin que l'environnement puisse être répliqué par Elastic Beanstalk sans avoir à installer des packages supplémentaires dont votre application n'a pas besoin.
4. Le package `awsebcli`. Il est utilisé pour initialiser votre application avec les fichiers nécessaires pour le déploiement avec Elastic Beanstalk.
5. Une installation `ssh` de travail. Elle est utilisée pour la connexion avec vos instances en cours d'exécution quand vous avez besoin d'examiner ou de déboguer un déploiement.

Pour de plus amples informations sur l'installation de Python, de pip et de l'interface de ligne de commande (CLI) EB, veuillez consulter [Installation de l'interface de ligne de commande EB \(p. 1028\)](#).

## Utilisation d'un environnement virtuel

Une fois les conditions préalables en place, configurez un environnement virtuel avec `virtualenv` pour installer les dépendances de votre application. En utilisant un environnement virtuel, vous pouvez

distinguer exactement de quels packages votre application a besoin afin que les packages requis soient installés sur les instances EC2 exécutant votre application.

Pour configurer un environnement virtuel

1. Ouvrez une fenêtre de ligne de commande et tapez :

```
$ virtualenv /tmp/eb_python_app
```

Remplacez `eb_python_app` par un nom logique pour votre application (utiliser votre nom de répertoire ou d'application est une bonne idée). La commande `virtualenv` crée un environnement virtuel pour vous dans le répertoire spécifié et imprime les résultats de ses actions :

```
Running virtualenv with interpreter /usr/bin/python
New python executable in /tmp/eb_python_app/bin/python3.7
Also creating executable in /tmp/eb_python_app/bin/python
Installing setuptools, pip...done.
```

2. Une fois que votre environnement virtuel est prêt, démarrez-le en exécutant le script `activate` situé dans le répertoire `bin` de l'environnement. Par exemple, pour démarrer l'environnement `eb_python_app` créé à l'étape précédente, tapez :

```
$ source /tmp/eb_python_app/bin/activate
```

L'environnement virtuel affiche son nom (par exemple : (`eb_python_app`) au début de chaque invite de commande, pour vous rappeler que vous êtes dans un environnement Python virtuel.

3. Pour arrêter d'utiliser votre environnement virtuel et revenir à l'interpréteur Python par défaut du système avec toutes les bibliothèques installées, exécutez la commande `deactivate`.

```
(eb_python_app) $ deactivate
```

#### Note

Une fois l'environnement virtuel créé, vous pouvez le redémarrer à tout moment en exécutant à nouveau son script `activate`.

## Configuration d'un projet Python pour Elastic Beanstalk

Vous pouvez utiliser l'interface de ligne de commande (CLI) Elastic Beanstalk pour préparer vos applications Python à déployer avec Elastic Beanstalk.

Pour configurer une application Python à déployer avec Elastic Beanstalk

1. Depuis votre [environnement virtuel \(p. 357\)](#), revenez vers le haut de l'arborescence de votre projet (`python_eb_app`) et saisissez :

```
pip freeze >requirements.txt
```

Cette commande copie les noms et les versions des packages qui sont installés dans votre environnement virtuel pour `requirements.txt`. Par exemple, si le package PyYAML, version 3.11 est installé dans votre environnement virtuel, le fichier contiendra la ligne :

```
PyYAML==3.11
```

Cela permet à Elastic Beanstalk de répliquer l'environnement Python de votre application à l'aide des mêmes packages et de la même version que ceux utilisés pour développer et tester votre application.

2. Configurez le référentiel de l'interface de ligne de commande (CLI) EB avec la commande eb init. Suivez les invites pour choisir une région, une plateforme et d'autres options. Pour obtenir des instructions complètes, veuillez consulter [Gestion des environnements Elastic Beanstalk avec l'interface de ligne de commande EB \(p. 1040\)](#).

Par défaut, Elastic Beanstalk recherche un fichier appelé `application.py` pour démarrer votre application. S'il n'existe pas dans le projet Python que vous avez créé, certains ajustements de l'environnement de votre application sont nécessaires. Vous devrez également définir des variables d'environnement afin que les modules de votre application puissent être chargés. Pour plus d'informations, consultez [Utilisation de la plateforme Python Elastic Beanstalk \(p. 359\)](#).

## Utilisation de la plateforme Python Elastic Beanstalk

### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [the section called “Mise à niveau vers Amazon Linux 2” \(p. 508\)](#).

La plateforme Python AWS Elastic Beanstalk est un ensemble de [versions de plateforme](#) pour les applications web Python qui peuvent s'exécuter derrière un serveur proxy avec WSGI. Chaque branche de plateforme correspond à une version de Python, telle que Python 3.8.

À partir des branches de plateforme Amazon Linux 2, Elastic Beanstalk fournit [Gunicorn](#) comme serveur WSGI par défaut.

Vous pouvez ajouter un [Procfile](#) à votre solution groupée source pour spécifier et configurer le serveur WSGI pour votre application. Pour plus d'informations, consultez [the section called “Procfile” \(p. 363\)](#).

Vous pouvez utiliser les fichiers [Pipfile](#) et [Pipfile.lock](#) créés par Pipenv pour spécifier les dépendances du package Python et d'autres exigences. Pour plus d'informations sur la spécification des dépendances, consultez [the section called “Spécification des dépendances” \(p. 363\)](#).

Elastic Beanstalk fournit des [options de configuration \(p. 658\)](#) que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute sur des instances EC2 dans votre environnement Elastic Beanstalk. Vous pouvez configurer des variables d'environnement nécessaires pour votre application, activer la rotation des journaux sur Amazon S3 et mapper des dossiers dans la source de votre application contenant des fichiers statiques vers des chemins desservis par le serveur proxy.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour [modifier la configuration d'un environnement en cours d'exécution \(p. 671\)](#). Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des [configurations enregistrées \(p. 779\)](#) pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des [fichiers de configuration \(p. 737\)](#). Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour

de plus amples informations sur la priorité et les autres méthodes de modification des paramètres, veuillez consulter [Options de configuration \(p. 658\)](#).

Pour les packages Python disponibles à partir de `pip`, vous pouvez également inclure un fichier d'exigences dans la racine du code source de votre application. Elastic Beanstalk installe tous les packages de dépendance spécifiés dans un fichier de configuration lors du déploiement. Pour plus d'informations, consultez [the section called “Spécification des dépendances” \(p. 363\)](#).

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter [the section called “Extension des plateformes Linux” \(p. 34\)](#).

## Configuration de votre environnement Python

Les paramètres de la plateforme Python vous permettent d'affiner le comportement de vos instances Amazon EC2. Vous pouvez modifier la configuration des instances Amazon EC2 de l'environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk.

Utilisez la console Elastic Beanstalk pour configurer les paramètres de processus Python, activer AWS X-Ray, activer la rotation des journaux vers Amazon S3 et configurer les variables que votre application peut lire depuis l'environnement.

Pour configurer votre environnement Python dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

## Paramètres Python

- Proxy server (Serveur proxy) – Serveur proxy à utiliser sur vos instances d'environnement. Le serveur nginx est utilisé par défaut.
- Chemin WSGI – Nom du chemin d'accès à votre fichier d'application principal. Par exemple, `application.py` ou `django/wsgi.py`.
- NumProcesses – Nombre de processus à exécuter sur chaque instance d'application.
- NumThreads – Nombre de threads à exécuter dans chaque processus.

## AWS X-RayParamètres

- Démon X-Ray – Exécutez le démon AWS X-Ray pour traiter les données de suivi à partir de [AWS X-Ray SDK for Python](#) .

## Options du journal

La section Options du journal a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.

- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

## Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour de plus amples informations sur la configuration des fichiers statiques à l'aide de la console Elastic Beanstalk, veuillez consulter [the section called “Fichiers statiques” \(p. 790\)](#).

Par défaut, le serveur proxy d'un environnement Python sert tous les fichiers dans un dossier nommé `static` sur le chemin d'accès `/static`. Par exemple, si votre code source d'application contient un fichier nommé `logo.png` dans un dossier nommé `static`, le serveur proxy le sert aux utilisateurs dans `subdomain.elasticbeanstalk.com/static/logo.png`. Vous pouvez configurer d'autres mappages comme expliqué dans cette section.

## Propriétés de l'environnement

Vous pouvez utiliser les propriétés de l'environnement afin de fournir des informations à votre application et de configurer des variables d'environnement. Par exemple, vous pouvez créer une propriété de l'environnement nommée `CONNECTION_STRING` qui spécifie une chaîne de connexion que votre application peut utiliser pour se connecter à une base de données.

A l'intérieur de l'environnement Python s'exécutant dans Elastic Beanstalk, ces valeurs sont accessibles à l'aide du dictionnaire `os.environ` Python. Pour plus d'informations, consultez <http://docs.python.org/library/os.html>.

Vous pouvez utiliser un code similaire au suivant pour accéder aux clés et aux paramètres :

```
import os
endpoint = os.environ['API_ENDPOINT']
```

Les propriétés de l'environnement peuvent également fournir des informations à une infrastructure. Par exemple, vous pouvez créer une propriété nommée `DJANGO_SETTINGS_MODULE` pour configurer Django pour utiliser un module de paramètres spécifique. Selon l'environnement, cette valeur peut être `development.settings`, `production.settings`, etc.

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Espaces de noms de la configuration Python

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

La plateforme Python définit des options dans les espaces de noms `aws:elasticbeanstalk:environment:proxy`, `aws:elasticbeanstalk:environment:proxy:staticfiles` et `aws:elasticbeanstalk:container:python`.

L'exemple suivant de fichier de configuration spécifie des paramètres d'option de configuration pour créer une propriété d'environnement nommée DJANGO\_SETTINGS\_MODULE, deux options de fichiers statiques qui mappent un répertoire nommé statichtml avec le chemin d'accès /html, un répertoire nommé staticimages avec le chemin d'accès /images et des paramètres supplémentaires dans l'espace de noms [aws:elasticbeanstalk:container:python \(p. 734\)](#). Cet espace de noms contient des options qui vous permettent de spécifier l'emplacement du script WSGI dans votre code source et le nombre de threads et de processus à exécuter dans WSGI.

```
option_settings:
  aws:elasticbeanstalk:application:environment:
    DJANGO_SETTINGS_MODULE: production.settings
  aws:elasticbeanstalk:environment:proxy:
    ProxyServer: apache
  aws:elasticbeanstalk:environment:proxy:staticfiles:
    /html: statichtml
    /images: staticimages
  aws:elasticbeanstalk:container:python:
    WSGIPath: ebdjango.wsgi:application
    NumProcesses: 3
    NumThreads: 20
```

## Notes

- Si vous utilisez une version de plateforme AMI Python Amazon Linux (antérieure à Amazon Linux 2), remplacez la valeur WSGIPath par ebdjango/wsgi.py. La valeur de l'exemple fonctionne avec le serveur WSGI Gunicorn, qui n'est pas pris en charge sur les versions de la plateforme AMI Amazon Linux.
- De plus, ces anciennes versions de plateforme utilisent un espace de noms différent pour configurer les fichiers statiques .—aws:elasticbeanstalk:container:python:staticfiles. Il a les mêmes noms d'option et la même sémantique que l'espace de noms de fichier statique standard.

Les fichiers de configuration prennent également en charge plusieurs clés permettant de [modifier davantage le logiciel sur les instances de votre environnement \(p. 740\)](#). Cet exemple utilise la clé de packages (p. 741) pour installer Memcached avec yum et des [commandes de conteneur \(p. 748\)](#) pour exécuter des commandes qui configurent le serveur durant le déploiement :

```
packages:
  yum:
    libmemcached-devel: '0.31'

container_commands:
  collectstatic:
    command: "django-admin.py collectstatic --noinput"
  01syncdb:
    command: "django-admin.py syncdb --noinput"
    leader_only: true
  02migrate:
    command: "django-admin.py migrate"
    leader_only: true
  03wsgipass:
    command: 'echo "WSGIPassAuthorization On" >> ../wsgi.conf'
  99customize:
    command: "scripts/customize.sh"
```

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Configuration du serveur WSGI avec un Procfile

### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [the section called “Mise à niveau vers Amazon Linux 2” \(p. 508\)](#).

Vous pouvez ajouter un `Procfile` à votre solution groupée source pour spécifier et configurer le serveur WSGI pour votre application. L'exemple suivant utilise un `Procfile` pour spécifier UWSGI comme serveur et le configurer.

### Example Procfile

```
web: uwsgi --http :8000 --wsgi-file application.py --master --processes 4 --threads 2
```

L'exemple suivant utilise un `Procfile` pour configurer Gunicorn, le serveur WSGI par défaut.

### Example Procfile

```
web: gunicorn --bind :8000 --workers 3 --threads 2 project.wsgi:application
```

### Notes

- Si vous configurez un serveur WSGI autre que Gunicorn, assurez-vous de le spécifier également en tant que dépendance de votre application, afin qu'il soit installé sur vos instances d'environnement. Pour de plus amples informations sur la spécification de dépendance, veuillez consulter [the section called “Spécification des dépendances” \(p. 363\)](#).
- Le port par défaut du serveur WSGI est 8 000. Si vous spécifiez un numéro de port différent dans votre commande `Procfile`, définissez également la [propriété d'environnement \(p. 632\)](#) `PORT` sur ce numéro de port.

Lorsque vous utilisez un `Procfile`, il remplace les options d'espace de noms `aws:elasticbeanstalk:container:python` que vous définissez à l'aide des fichiers de configuration.

Pour de plus amples informations sur l'utilisation de `Procfile`, veuillez développer la section `Buildfile` et `Procfile` dans [the section called “Extension des plateformes Linux” \(p. 34\)](#).

## Spécification des dépendances à l'aide d'un fichier Requirements

Une application Python classique comporte des dépendances par rapport à d'autres packages Python tiers. La plateforme Python Elastic Beanstalk vous permet de spécifier les paquets Python dont dépend votre application de plusieurs façons.

### Utilisation de pip et requirements.txt

L'outil standard pour installer les packages Python est `pip`. Il s'agit d'une fonctionnalité qui vous permet de spécifier tous les packages dont vous avez besoin (ainsi que leurs versions) dans un fichier Requirements unique. Pour plus d'informations sur le fichier Requirements, consultez [Requirements File Format](#).

Créez un fichier `requirements.txt` et placez-le dans le répertoire de niveau supérieur de votre solution groupée source. Voici un exemple de fichier `requirements.txt` pour Django.

```
Django==2.2
mysqlclient==2.0.3
```

Dans votre environnement de développement, vous pouvez utiliser la commande `pip freeze` pour générer votre fichier Requirements.

```
~/my-app$ pip freeze > requirements.txt
```

Pour vous assurer que votre fichier Requirements contient uniquement des packages qui sont réellement utilisés par votre application, utilisez un [environnement virtuel \(p. 357\)](#) sur lequel seuls ces packages sont installés. En dehors d'un environnement virtuel, le résultat de `pip freeze` comprendra tous les packages `pip` installés sur votre ordinateur de développement, y compris ceux livrés avec votre système d'exploitation.

#### Note

Sur les versions de la plateforme AMI Python Amazon Linux, Elastic Beanstalk ne prend pas en charge nativement Pipenv ou Pipfiles. Si vous utilisez Pipenv pour gérer les dépendances de votre application, exécutez la commande suivante pour générer un fichier `requirements.txt`.

```
~/my-app$ pipenv lock -r > requirements.txt
```

Pour en savoir plus, consultez [Generating a requirements.txt](#) dans la documentation.

## Utilisation de Pipenv et Pipfile

Pipenv est un outil d'emballage Python moderne. Il combine l'installation de packages avec la création et la gestion d'un fichier de dépendance et d'un environnement virtuel pour votre application. Pipenv gère deux fichiers : `Pipfile` contient différents types de dépendances et d'exigences, et `Pipfile.lock` est un instantané de version qui permet des builds déterministes. Pour de plus amples informations, veuillez consulter [Pipenv: Python Dev Workflow for Humans](#).

Les versions de la plateforme Python Amazon Linux 2 prennent en charge les fichiers d'exigences basés sur Pipenv. Créez-les dans votre environnement de développement et incluez-les dans le bundle de fichiers source que vous déployez sur Elastic Beanstalk.

#### Note

Les versions de plateforme AMI Python Amazon Linux (antérieure à Amazon Linux 2) ne prennent pas en charge Pipenv et `Pipfile`.

L'exemple suivant utilise Pipenv pour installer Django et le framework REST Django.

```
~/my-app$ pipenv install django
~/my-app$ pipenv install djangorestframework
```

Ces commandes créent les fichiers `Pipfile` et `Pipfile.lock`. Placez `Pipfile` dans le répertoire de niveau supérieur de votre solution groupée source pour obtenir les dernières versions des packages de dépendance installés sur vos instances d'environnement. Vous pouvez également inclure `Pipfile.lock` pour obtenir un ensemble constant de versions de package reflétant votre environnement de développement au moment de la création du fichier.

Si vous incluez plusieurs fichiers d'exigences décrits ici, Elastic Beanstalk n'en utilise qu'un. La liste suivante montre la priorité, par ordre décroissant.

1. `requirements.txt`
2. `Pipfile.lock`
3. `Pipfile`

## Déploiement d'une application Flask sur Elastic Beanstalk

Flask est une infrastructure d'application web open source pour Python. Ce didacticiel vous guide tout au long du processus de génération d'une application Flask et de son déploiement dans un environnement AWS Elastic Beanstalk.

Dans le cadre de ce didacticiel, vous effectuerez les tâches suivantes :

- Configuration d'un environnement virtuel Python avec Flask (p. 365)
- Création d'une application Flask (p. 366)
- Déploiement de votre site avec l'interface de ligne de commande (CLI) EB (p. 368)
- Cleanup (p. 371)

## Prerequisites

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

Flask nécessite Python 3.6 ou version ultérieure. Dans ce tutoriel, nous utilisons Python 3.6 et la version correspondante de la plateforme Elastic Beanstalk. Installez Python en suivant les instructions à l'adresse [Configuration de votre environnement de développement Python \(p. 357\)](#).

L'infrastructure [Flask](#) sera installée dans le cadre de ce didacticiel.

Ce didacticiel utilise également l'interface de ligne de commande (CLI) Elastic Beanstalk (EB). Pour de plus amples informations sur l'installation et la configuration de la CLI EB, veuillez consulter [Installation de l'interface de ligne de commande EB \(p. 1028\)](#) et [Configuration de l'interface de ligne de commande EB \(p. 1036\)](#).

## Configuration d'un environnement virtuel Python avec Flask

Créez un répertoire de projet et un environnement virtuel pour votre application, puis installez Flask.

Pour configurer votre environnement de projet

1. Créez un répertoire de projet.

```
~$ mkdir eb-flask
~$ cd eb-flask
```

2. Créez et activez un environnement virtuel nommé `virt`:

```
~/eb-flask$ virtualenv virt
~$ source virt/bin/activate
(virt) ~/eb-flask$
```

Vous verrez `(virt)` ajouté à votre invite de commande, ce qui indique que vous êtes dans un environnement virtuel. Utilisez l'environnement virtuel pour le reste du didacticiel.

3. Installez Flask avec `pip install`:

```
(virt)~/eb-flask$ pip install flask==1.1.2
```

4. Affichez les bibliothèques installées avec `pip freeze`:

```
(virt)~/eb-flask$ pip freeze
click==7.1.2
Flask==1.1.2
itsdangerous==1.1.0
Jinja2==2.11.3
MarkupSafe==1.1.1
Werkzeug==1.0.1
```

Cette commande répertorie tous les packages installés dans votre environnement virtuel. Étant donné que vous êtes dans un environnement virtuel, globalement les packages installés tels que l'interface de ligne de commande (CLI) EB ne sont pas affichés.

5. Enregistrez la sortie de `pip freeze` dans un fichier nommé `requirements.txt`.

```
(virt)~/eb-flask$ pip freeze > requirements.txt
```

Ce fichier indique à Elastic Beanstalk d'installer les bibliothèques pendant le déploiement. Pour plus amples informations, veuillez consulter [Spécification des dépendances à l'aide d'un fichier Requirements \(p. 363\)](#).

## Création d'une application Flask

Créez ensuite une application qui vous allez déployer à l'aide d'Elastic Beanstalk. Nous allons créer un service web RESTful « Hello World ».

Créez un nouveau fichier texte dans ce répertoire nommé `application.py` avec le contenu suivant :

Example `~/eb-flask/application.py`

```
from flask import Flask

# print a nice greeting.
def say_hello(username = "World"):
    return '<p>Hello %s!</p>\n' % username
```

```
# some bits of text for the page.
header_text = '''
    <html>\n<head> <title>EB Flask Test</title> </head>\n<body>'''
instructions = '''
    <p><em>Hint</em>: This is a RESTful web service! Append a username
    to the URL (for example: <code>/Thelonious</code>) to say hello to
    someone specific.</p>\n'''
home_link = '<p><a href="/">Back</a></p>\n'
footer_text = '</body>\n</html>'

# EB looks for an 'application' callable by default.
application = Flask(__name__)

# add a rule for the index page.
application.add_url_rule('/', 'index', (lambda: header_text +
    say_hello() + instructions + footer_text))

# add a rule when the page is accessed with a name appended to the site
# URL.
application.add_url_rule('/<username>', 'hello', (lambda username:
    header_text + say_hello(username) + home_link + footer_text))

# run the app.
if __name__ == "__main__":
    # Setting debug to True enables debug output. This line should be
    # removed before deploying a production app.
    application.debug = True
    application.run()
```

Cet exemple affiche un message d'accueil personnalisé qui varie selon le chemin d'accès utilisé pour accéder au service.

#### Note

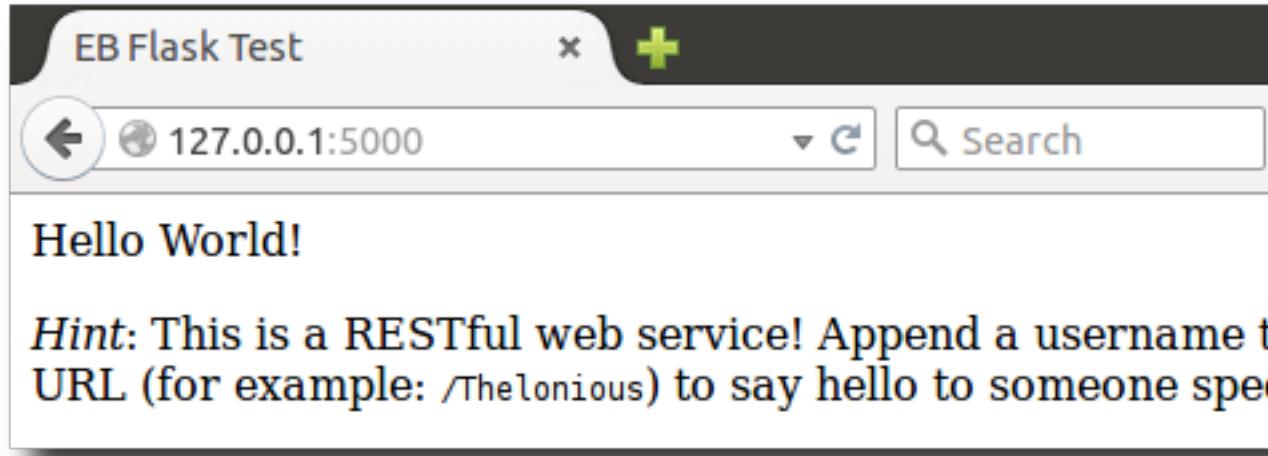
En ajoutant `application.debug = True` avant d'exécuter l'application, la sortie de débogage est activée pour parer à une éventuelle défaillance. C'est une bonne pratique pour le développement, mais vous devez supprimer les instructions de débogage dans le code de production, car la sortie de débogage peut révéler des aspects internes de votre application.

Utiliser `application.py` comme nom de fichier et fournir un objet `application` joignable (dans ce cas, l'objet Flask) permet à Elastic Beanstalk de trouver facilement le code de votre application.

Exécutez `application.py` avec Python :

```
(virt) ~/eb-flask$ python application.py
* Serving Flask app "application" (lazy loading)
* Environment: production
WARNING: Do not use the development server in a production environment.
Use a production WSGI server instead.
* Debug mode: on
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 313-155-123
```

Ouvrez `http://127.0.0.1:5000/` dans votre navigateur web. Vous devriez voir l'application en cours d'exécution, affichant la page d'index :



Consultez le journal du serveur pour voir le résultat de votre demande. Vous pouvez arrêter le serveur web et revenir à votre environnement virtuel en appuyant sur Ctrl+C.

Si vous avez obtenu la sortie de débogage à la place, corrigez les erreurs et veillez à ce que l'application s'exécute localement avant de procéder à la configuration pour Elastic Beanstalk.

## Déploiement de votre site avec l'interface de ligne de commande (CLI) EB

Vous avez ajouté tout ce dont vous avez besoin pour déployer votre application sur Elastic Beanstalk. Votre répertoire de projet devrait maintenant ressembler à ceci :

```
~/eb-flask/
|-- virt
|-- application.py
`-- requirements.txt
```

Le dossier `virt`, toutefois, n'est pas requis pour que l'application s'exécute sur Elastic Beanstalk. Au moment du déploiement, Elastic Beanstalk crée un environnement virtuel sur les instances de serveur et installe les bibliothèques répertoriées dans `requirements.txt`. Pour réduire la taille du bundle source que vous chargez pendant le déploiement, ajoutez un fichier [.ebignore \(p. 1038\)](#) qui demande à l'interface de ligne de commande (CLI) EB d'exclure le dossier `virt`.

Example `~/Eb-flask/.ebignore`

```
virt
```

Ensuite, vous allez créer votre environnement d'applications et déployer votre application configurée avec Elastic Beanstalk.

Pour créer un environnement et déployer votre application Flask

1. Initialisez votre référentiel d'interface de ligne de commande (CLI) EB avec la commande `eb init` :

```
~/eb-flask$ eb init -p python-3.6 flask-tutorial --region us-east-2
Application flask-tutorial has been created.
```

Cette commande crée une application appelée `flask-tutorial` et configure votre référentiel local afin de créer des environnements avec la dernière version de plateforme Python 3.6.

2. (facultatif) Exécutez `eb init` à nouveau pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l'instance EC2 exécutant votre application avec SSH :

```
~/eb-flask$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez `eb init -i`.

3. Créez un environnement et déployez-y votre application avec `eb create`:

```
~/eb-flask$ eb create flask-env
```

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

#### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande `sudo yum install` reprise dans l'article. Vous pouvez également utiliser la commande `sudo sed` reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.

- Groupe de sécurité de l'équilibrer de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrer de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

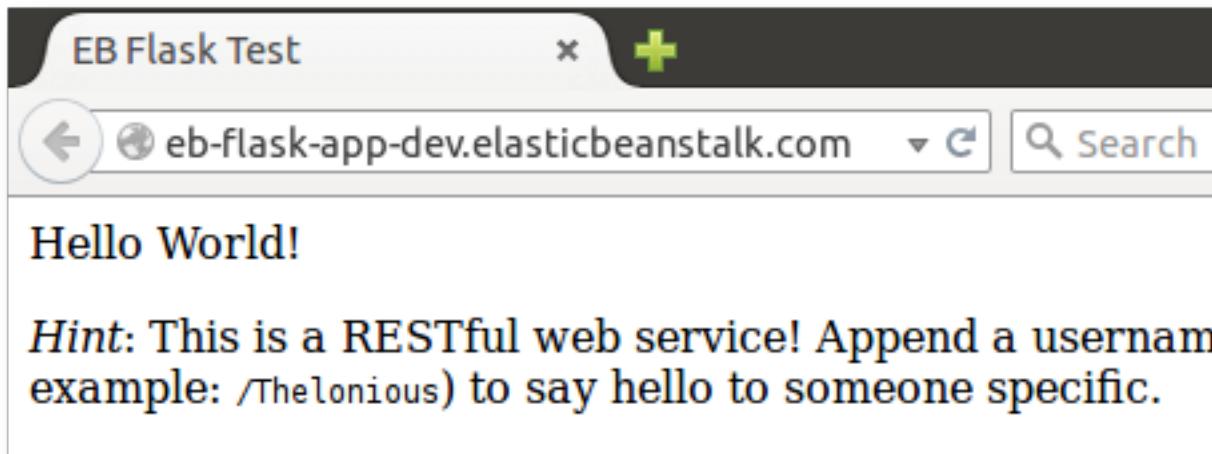
#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

Lorsque le processus de création de l'environnement est terminé, ouvrez votre site web avec `eb open`:

```
~/eb-flask$ eb open
```

Cela ouvre une fenêtre de navigateur à l'aide du nom de domaine créé pour votre application. Vous devez voir le même site web Flask que celui que vous avez créé et testé localement.



Si vous ne voyez pas votre application en cours d'exécution ou si vous obtenez un message d'erreur, consultez [Déploiements \(p. 1142\)](#) afin d'obtenir de l'aide concernant la façon de déterminer la cause de l'erreur.

Si vous voyez effectivement votre application en cours d'exécution, alors félicitations, vous avez déployé votre première application Flask avec Elastic Beanstalk !

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 371\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

Ou, avec l'interface de ligne de commande (CLI) EB :

```
~/eb-flask$ eb terminate flask-env
```

## Étapes suivantes

Pour plus d'informations sur Flask, visitez [flask.pocoo.org](http://flask.pocoo.org).

Si vous souhaitez essayer un autre cadre web Python, veuillez consulter [Déploiement d'une application Django sur Elastic Beanstalk \(p. 371\)](#).

## Déploiement d'une application Django sur Elastic Beanstalk

Ce didacticiel vous guide tout au long du déploiement d'un site web [Django](#) par défaut, généré automatiquement, sur un environnement AWS Elastic Beanstalk exécutant Python. Ce tutoriel vous montre comment héberger une application web Python dans le cloud à l'aide d'un environnement Elastic Beanstalk.

Dans le cadre de ce didacticiel, vous effectuerez les tâches suivantes :

- Configuration d'un environnement virtuel Python et installation de Django (p. 372)
- Création d'un projet Django (p. 373)
- Configurer votre application Django pour Elastic Beanstalk (p. 374)
- Déploiement de votre site avec l'interface de ligne de commande (CLI) EB (p. 376)
- Mise à jour de votre application (p. 378)
- Nettoyage (p. 381)

## Prerequisites

Pour utiliser n'importe quel service AWS, y compris Elastic Beanstalk, vous devez disposer d'un compte AWS et d'informations d'identification. Pour en savoir plus et vous inscrire, veuillez consulter <https://aws.amazon.com/>.

Pour suivre ce didacticiel, la [Configuration de votre environnement de développement Python \(p. 357\)](#) doit être terminée et les packages suivants installés :

- Python 3.6 ou version ultérieure
- pip
- virtualenv
- awsebcli

L'infrastructure [Django](#) est installée dans le cadre de ce tutoriel.

### Note

Créer des environnements avec l'interface de ligne de commande (CLI) EB nécessite un [rôle de service \(p. 21\)](#). Vous pouvez créer un rôle de service en créant un environnement dans la console Elastic Beanstalk. Si vous n'avez pas de rôle de service, l'interface de ligne de commande (CLI) EB essaie d'en créer un lorsque vous exécutez `eb create`.

## Configuration d'un environnement virtuel Python et installation de Django

Créez un environnement virtuel avec `virtualenv` et utilisez-le pour installer Django et ses dépendances. En utilisant un environnement virtuel, vous pouvez savoir exactement de quels packages votre application a besoin, afin que les packages requis soient installés sur les instances Amazon EC2 exécutant votre application.

Les étapes suivantes illustrent les commandes que vous devez entrer pour les systèmes Unix et Windows, affichées sur des onglets distincts.

Pour configurer votre environnement virtuel

1. Créez un environnement virtuel nommé `eb-virt`.

### Unix-based systems

```
~$ virtualenv ~/eb-virt
```

### Windows

```
C:\> virtualenv %HOMEPATH%\eb-virt
```

2. Activez l'environnement virtuel.

### Unix-based systems

```
~$ source ~/eb-virt/bin/activate  
(eb-virt) ~$
```

### Windows

```
C:\>%HOMEPATH%\eb-virt\Scripts\activate
```

```
(eb-virt) C:\>
```

Vous verrez (eb-virt) ajouté à votre invite de commande, ce qui indique que vous êtes dans un environnement virtuel.

#### Note

Les instructions restantes montrent l'invite de commande Linux dans votre répertoire de base ~\$. Sous Windows, il s'agit de C:\Users\*USERNAME*>, où *USERNAME* est votre nom de connexion Windows.

3. Utilisez pip pour installer Django.

```
(eb-virt)~$ pip install django==2.2
```

#### Note

La version Django que vous installez doit être compatible avec la version Python sur la configuration Elastic Beanstalk Python que vous choisissez pour déployer votre application. Pour plus d'informations sur le déploiement, consultez [??? \(p. 376\)](#) dans cette rubrique. Pour plus d'informations sur les versions actuelles de la plateforme Python, consultez [Python](#) dans le document Plateformes prises en charge par AWS Elastic Beanstalk. Pour plus d'informations sur la compatibilité de la version Django avec Python, consultez [Quelle version de Python puis-je utiliser avec Django ?](#)

4. Pour vérifier que Django est installé, entrez ce qui suit.

```
(eb-virt)~$ pip freeze
Django==2.2
...
```

Cette commande répertorie tous les packages installés dans votre environnement virtuel. Plus tard, vous utiliserez le résultat de cette commande pour configurer votre projet à utiliser avec Elastic Beanstalk.

## Création d'un projet Django

Vous êtes maintenant prêt à créer un projet Django et à l'exécuter sur votre ordinateur, à l'aide de l'environnement virtuel.

#### Note

Ce didacticiel utilise SQLite, qui est un moteur de base de données inclus dans Python. La base de données est déployée avec vos fichiers de projet. Pour les environnements de production, nous vous recommandons d'utiliser Amazon Relational Database Service (Amazon RDS) et de le séparer de votre environnement. Pour de plus amples informations, veuillez consulter [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Python \(p. 382\)](#).

Pour générer une application Django

1. Activez votre environnement virtuel.

Unix-based systems

```
-$ source ~/eb-virt/bin/activate
(eb-virt) ~$
```

## Windows

```
C:\>%HOMEPATH%\eb-virt\Scripts\activate  
(eb-virt) C:\>
```

Vous verrez le préfixe (eb-virt) ajouté à votre invite de commande, ce qui indique que vous êtes dans un environnement virtuel.

### Note

Les instructions restantes montrent l'invite de commande Linux ~\$ dans votre répertoire de base et le répertoire de base Linux ~/. Sous Windows, il s'agit de C:\Users\<USERNAME>, où **USERNAME** est votre nom de connexion Windows.

2. Utilisez la commande `django-admin startproject` pour créer un projet Django nommé ebdjango.

```
(eb-virt)~$ django-admin startproject ebdjango
```

Cette commande crée un site Django standard nommé ebdjango avec la structure de répertoires suivante.

```
~/ebdjango
| -- ebdjango
|   | -- __init__.py
|   | -- settings.py
|   | -- urls.py
|   | -- wsgi.py
`-- manage.py
```

3. Exécutez votre site Django localement avec `manage.py runserver`.

```
(eb-virt) ~$ cd ebdjango
```

```
(eb-virt) ~/ebdjango$ python manage.py runserver
```

4. Dans un navigateur web, ouvrez `http://127.0.0.1:8000/` pour afficher le site.
5. Consultez le journal du serveur pour voir le résultat de votre demande. Pour arrêter le serveur web et revenir à votre environnement virtuel, appuyez sur Ctrl+C.

```
Django version 2.2, using settings 'ebdjango.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CONTROL-C.
[07/Sep/2018 20:14:09] "GET / HTTP/1.1" 200 16348
Ctrl+C
```

## Configurer votre application Django pour Elastic Beanstalk

Maintenant que vous avez un site Django résidant sur votre ordinateur local, vous pouvez le configurer pour le déploiement avec Elastic Beanstalk.

Par défaut, Elastic Beanstalk recherche un fichier nommé `application.py` pour démarrer votre application. Comme il n'existe pas dans le projet Django que vous avez créé, vous devez apporter

quelques modifications à l'environnement de votre application. Vous devez également définir des variables d'environnement afin que les modules de votre application puissent être chargés.

Pour configurer votre site pour Elastic Beanstalk

1. Activez votre environnement virtuel.

Unix-based systems

```
~/ebdjango$ source ~/eb-virt/bin/activate
```

Windows

```
C:\Users\USERNAME\ebdjango>%HOMEPATH%\eb-virt\Scripts\activate
```

2. Exécutez `pip freeze`, puis enregistrez les données de sortie dans un fichier nommé `requirements.txt`.

```
(eb-virt) ~/ebdjango$ pip freeze > requirements.txt
```

Elastic Beanstalk utilise `requirements.txt` pour déterminer quel package installer sur les instances EC2 qui exécutent votre application.

3. Créez un répertoire nommé `.ebextensions`.

```
(eb-virt) ~/ebdjango$ mkdir .ebextensions
```

4. Dans le répertoire `.ebextensions`, ajoutez un [fichier de configuration \(p. 737\)](#) nommé `django.config` avec le texte suivant.

Example `~/ebdjango/.ebextensions/django.config`

```
option_settings:  
aws:elasticbeanstalk:container:python:  
    WSGIPath: ebdjango.wsgi:application
```

Ce paramètre, `WSGIPath`, spécifie l'emplacement du script WSGI qu'Elastic Beanstalk utilise pour lancer votre application.

#### Note

Si vous utilisez une version de plateforme AMI Python Amazon Linux (antérieure à Amazon Linux 2), remplacez la valeur `WSGIPath` par `ebdjango/wsgi.py`. La valeur de l'exemple fonctionne avec le serveur WSGI Gunicorn, qui n'est pas pris en charge sur les versions de la plateforme AMI Amazon Linux.

5. Désactivez votre environnement virtuel avec la commande `deactivate`.

```
(eb-virt) ~/ebdjango$ deactivate
```

Réactivez votre environnement virtuel chaque fois que vous devez ajouter des packages à votre application ou exécuter votre application localement.

## Déploiement de votre site avec l'interface de ligne de commande (CLI) EB

Vous avez ajouté tout ce dont vous avez besoin pour déployer votre application sur Elastic Beanstalk. Le répertoire de votre projet devrait maintenant se présenter comme suit.

```
~/ebdjango/
|-- .ebextensions
|   '-- django.config
|-- ebdjango
|   |-- __init__.py
|   |-- settings.py
|   |-- urls.py
|   '-- wsgi.py
|-- db.sqlite3
|-- manage.py
`-- requirements.txt
```

Ensuite, vous allez créer votre environnement d'applications et déployer votre application configurée avec Elastic Beanstalk.

Immédiatement après le déploiement, vous allez modifier la configuration de Django afin d'ajouter le nom de domaine attribué par Elastic Beanstalk à votre application au code de Django `ALLOWED_HOSTS`. Ensuite, vous redéployerez votre application. Il s'agit d'une exigence de sécurité de Django, conçue pour empêcher les attaques de l'en-tête `HTTP Host`. Pour plus d'informations, consultez la section [Validation de l'en-tête Host](#).

Pour créer un environnement et déployer votre application Django

### Note

Ce tutoriel utilise l'interface de ligne de commande (CLI) EB comme mécanisme de déploiement, mais vous pouvez également utiliser la console Elastic Beanstalk pour déployer un fichier ZIP où se trouve le contenu de votre projet.

1. Initialisez votre référentiel de la CLI EB avec la commande `eb init` :

```
~/ebdjango$ eb init -p python-3.6 django-tutorial
Application django-tutorial has been created.
```

Cette commande crée une application nommée `django-tutorial`. Elle configure également votre référentiel local pour créer des environnements avec la dernière version de la plateforme Python 3.6.

2. (facultatif) Exécutez à nouveau la commande `eb init` pour configurer une paire de clés par défaut afin de pouvoir vous connecter à l'instance EC2 qui exécute votre application.

```
~/ebdjango$ eb init
Do you want to set up SSH for your instances?
(y/n): y
Select a keypair.
1) my-keypair
2) [ Create new KeyPair ]
```

Sélectionnez une paire de clés si vous en avez déjà une, ou suivez les invites pour en créer une. Si vous ne voyez pas l'invite ou que vous avez besoin de modifier vos paramètres ultérieurement, exécutez `eb init -i`.

3. Créez un environnement et déployez-y votre application avec `eb create`.

```
~/ebdjango$ eb create django-env
```

#### Note

Si un message d'erreur « rôle de service requis » s'affiche, exécutez `eb create` de manière interactive (sans spécifier de nom d'environnement) et l'interface de ligne de commande (CLI) EB créera le rôle pour vous.

Cette commande crée un environnement Elastic Beanstalk équilibré en charge nommé `django-env`. La création d'un environnement prend environ 5 minutes. Lorsqu'Elastic Beanstalk crée les ressources nécessaires pour exécuter votre application, il génère des messages d'information que l'interface de ligne de commande (CLI) EB relaie à votre terminal.

4. Lorsque le processus de création de l'environnement est terminé, recherchez le nom de domaine de votre nouvel environnement en exécutant `eb status`.

```
~/ebdjango$ eb status
Environment details for: django-env
  Application name: django-tutorial
  ...
  CNAME: eb-django-app-dev.elasticbeanstalk.com
  ...
```

Le nom de domaine de votre environnement est la valeur de la propriété `CNAME`.

5. Ouvrez le fichier `settings.py` dans le répertoire `ebdjango`. Recherchez le paramètre `ALLOWED_HOSTS`, puis ajoutez le nom de domaine de votre application que vous avez trouvé à l'étape précédente à la valeur de ce paramètre. Si vous ne trouvez pas ce paramètre dans le fichier, ajoutez-le sur une nouvelle ligne.

```
...
ALLOWED_HOSTS = ['eb-django-app-dev.elasticbeanstalk.com']
```

6. Enregistrez le fichier, puis déployez votre application en exécutant `eb deploy`. Lorsque vous exécutez `eb deploy`, l'interface de ligne de commande (CLI) EB crée un bundle avec le contenu de votre répertoire de projet et le déploie dans votre environnement.

```
~/ebdjango$ eb deploy
```

#### Note

Si vous utilisez Git avec votre projet, veuillez consulter [Utilisation de l'interface de ligne de commande EB avec Git \(p. 1046\)](#).

7. Lorsque le processus de mise à jour de l'environnement est terminé, ouvrez votre site web avec la commande `eb open`.

```
~/ebdjango$ eb open
```

Celle-ci ouvre une fenêtre de navigation en utilisant le nom de domaine créé pour votre application. Vous devez voir le même site web Django que celui que vous avez créé et testé localement.

Si vous ne voyez pas votre application en cours d'exécution ou si vous obtenez un message d'erreur, consultez [Déploiements \(p. 1142\)](#) afin d'obtenir de l'aide concernant la façon de déterminer la cause de l'erreur.

Si vous voyez effectivement votre application en cours d'exécution, alors félicitations, vous avez déployé votre première application Django avec Elastic Beanstalk !

## Mise à jour de votre application

Maintenant que vous disposez d'une application en cours d'exécution sur Elastic Beanstalk, vous pouvez mettre à jour et redéployer votre application ou sa configuration, et Elastic Beanstalk va effectuer les tâches de mise à jour de vos instances et de démarrage de la nouvelle version de votre application.

Pour cet exemple, nous activerons la console d'administration Django et configurerons quelques autres paramètres.

### Modification des paramètres de votre site

Par défaut, votre site web Django utilise le fuseau horaire UTC pour afficher l'heure. Vous pouvez modifier cela en spécifiant un fuseau horaire dans `settings.py`.

Pour modifier le fuseau horaire de votre site

1. Modifiez le paramètre `TIME_ZONE` dans `settings.py`.

Example `~/ebdjango/ebdjango/settings.py`

```
...
# Internationalization
LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'US/Pacific'
USE_I18N = True
USE_L10N = True
USE_TZ = True
```

Pour obtenir une liste des fuseaux horaires, consultez [cette page](#).

2. Déployez l'application dans votre environnement Elastic Beanstalk.

```
~/ebdjango/$ eb deploy
```

### Création d'un administrateur de site

Vous pouvez créer un administrateur de site pour votre application Django afin d'accéder à la console d'administration directement à partir du site web. Les informations de connexion administrateur sont stockées en toute sécurité dans l'image de base de données locale incluse dans le projet par défaut que Django génère.

Pour créer un administrateur de site

1. Initialisez la base de données locale de votre application Django.

```
(eb-virt) ~/ebdjango$ python manage.py migrate
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
```

```
Applying contenttypes.0002_remove_content_type_name... OK
Applying auth.0002_alter_permission_name_max_length... OK
Applying auth.0003_alter_user_email_max_length... OK
Applying auth.0004_alter_user_username_opts... OK
Applying auth.0005_alter_user_last_login_null... OK
Applying auth.0006_require_contenttypes_0002... OK
Applying auth.0007_alter_validators_add_error_messages... OK
Applying auth.0008_alter_user_username_max_length... OK
Applying auth.0009_alter_user_last_name_max_length... OK
Applying sessions.0001_initial... OK
```

- Exédez `manage.py createsuperuser` pour créer un administrateur.

```
(eb-virt) ~/ebdjango$ python manage.py createsuperuser
Username: admin
Email address: me@mydomain.com
Password: *****
Password (again): *****
Superuser created successfully.
```

- Pour dire à Django où stocker les fichiers statiques, définissez `STATIC_ROOT` dans `settings.py`.

Example `~/ebdjango/ebdjango/settings.py`

```
# Static files (CSS, JavaScript, Images)
# https://docs.djangoproject.com/en/2.2/howto/static-files/
STATIC_URL = '/static/'
STATIC_ROOT = 'static'
```

- Exédez `manage.py collectstatic` pour remplir le répertoire `static` avec des ressources statiques (JavaScript, CSS et images) pour le site d'administration.

```
(eb-virt) ~/ebdjango$ python manage.py collectstatic
119 static files copied to ~/ebdjango/static
```

- Déployez votre application.

```
~/ebdjango$ eb deploy
```

- Affichez la console d'administration en ouvrant le site dans votre navigateur, en ajoutant `/admin/` à l'URL du site, comme suit.

```
http://djang-env.p33kq46sfh.us-west-2.elasticbeanstalk.com/admin/
```

The screenshot shows the Django administration login interface. It features a dark blue header bar with the text "Django administration". Below it is a white form area. The first field is labeled "Username:" with the value "admin" entered. The second field is labeled "Password:" with a series of black dots representing the password. At the bottom right of the form is a blue "Log in" button.

7. Connectez-vous avec le nom utilisateur et le mot de passe que vous avez configurés à l'étape 2.

The screenshot shows the Django administration site list page. The top navigation bar includes the text "Django administration", "WELCOME, ADMIN.", and links for "VIEW SITE / CHANGE PASSWORD / LOG OUT". Below this is a section titled "Site administration". A blue header bar labeled "AUTHENTICATION AND AUTHORIZATION" contains two entries: "Groups" with "+ Add" and "Change" buttons, and "Users" with "+ Add" and "Change" buttons. To the right, there is a sidebar titled "Recent actions" which is currently empty, and another titled "My actions" which also says "None available".

Vous pouvez utiliser une procédure similaire de mise à jour/test locale suivie de eb deploy. Elastic Beanstalk prend en charge les tâches de mise à jour de vos serveurs en direct, afin que vous puissiez vous concentrer sur le développement d'applications au lieu de l'administration du serveur !

### Ajout d'un fichier de configuration de migration de base de données

Vous pouvez ajouter des commandes à votre script `.ebextensions`, qui sont exécutées lors de la mise à jour de votre site. Cela vous permet de générer automatiquement des migrations de base de données.

Pour ajouter une étape de migration lorsque votre application est déployée

1. Créez un [fichier de configuration \(p. 737\)](#) nommé `db-migrate.config` avec le contenu suivant.

Example `~/ebdjango/.ebextensions/db-migrate.config`

```
container_commands:  
  01_migrate:  
    command: "django-admin.py migrate"  
    leader_only: true  
option_settings:  
  aws:elasticbeanstalk:application:environment:  
    DJANGO_SETTINGS_MODULE: ebdjango.settings
```

Ce fichier de configuration exécute la commande `django-admin.py migrate` lors du processus de déploiement, avant le démarrage de votre application. Étant donné que cette commande est exécutée avant le démarrage de l'application, vous devez également configurer la variable d'environnement `DJANGO_SETTINGS_MODULE` explicitement (généralement, `wsgi.py` s'occupe de cette tâche pour vous pendant le démarrage). La spécification de `leader_only: true` dans la commande garantit qu'elle est exécutée une seule fois lors d'un déploiement sur plusieurs instances.

2. Déployez votre application.

```
~/ebdjango$ eb deploy
```

## Nettoyage

Pour économiser les heures d'instance et d'autres ressources AWS entre les sessions de développement, résiliez votre environnement Elastic Beanstalk avec `eb terminate`.

```
~/ebdjango$ eb terminate django-env
```

Cette commande résilie l'environnement et toutes les ressources AWS qui s'y exécutent. Cependant, elle ne supprime pas l'application. Vous pouvez donc toujours créer d'autres environnements avec la même configuration en exécutant à nouveau `eb create`. Pour de plus amples informations sur les commandes de l'interface de ligne de commande (CLI) EB, veuillez consulter [Gestion des environnements Elastic Beanstalk avec l'interface de ligne de commande EB \(p. 1040\)](#).

Si vous en avez terminé avec l'exemple d'application, vous pouvez également supprimer le dossier du projet et l'environnement virtuel.

```
~$ rm -rf ~/eb-virt  
~$ rm -rf ~/ebdjango
```

## Étapes suivantes

Pour plus d'informations sur Django, y compris un didacticiel approfondi, consultez [la documentation officielle](#).

Si vous souhaitez essayer un autre cadre web Python, veuillez consulter [Déploiement d'une application Flask sur Elastic Beanstalk \(p. 365\)](#).

## Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Python

Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être associée à votre environnement et gérée par Elastic Beanstalk, ou créée et gérée en externe.

Si vous utilisez Amazon RDS pour la première fois, [ajoutez une instance de base de données \(p. 382\)](#) à un environnement de test avec la console de gestion Elastic Beanstalk et assurez-vous que votre application peut s'y connecter.

Pour vous connecter à une base de données, [ajoutez le pilote \(p. 383\)](#) à votre application, chargez-le dans votre code et [créez un objet de connexion \(p. 383\)](#) avec les propriétés d'environnement fournies par Elastic Beanstalk. La configuration et le code de connexion varient selon le moteur de base de données et l'infrastructure que vous utilisez.

### Sections

- [Ajout d'une instance de base de données à votre environnement \(p. 382\)](#)
- [Téléchargement d'un pilote \(p. 383\)](#)
- [Connexion à une base de données \(p. 383\)](#)

## Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Choisissez Apply.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des	Sous l'onglet Connectivity & security (Connectivité et sécurité)

Nom de la propriété	Description	Valeur de la propriété
	connexions. La valeur par défaut varie selon les moteurs de base de données.	de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour de plus amples informations sur la configuration d'une instance de base de données interne, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

## Téléchargement d'un pilote

Ajoutez le pilote de base de données au [fichier Requirements \(p. 363\)](#) de votre projet.

Example requirements.txt – Django avec MySQL

```
Django==2.2
mysqlclient==2.0.3
```

Packages de pilotes courants pour Python

- MySQL – mysqlclient
- PostgreSQL – psycopg2
- Oracle – cx\_Oracle
- SQL Server – adodbapi

Pour plus d'informations, consultez [Python Database Interfaces](#) et [Django 2.2 – bases de données prises en charge](#).

## Connexion à une base de données

Elastic Beanstalk fournit des informations de connexion pour les instances de base de données attachées dans les propriétés de l'environnement. Utilisez `os.environ['VARIABLE']` pour lire les propriétés et configurer une connexion de base de données.

Example Fichier de paramètres Django – Dictionnaire DATABASES

```
import os

if 'RDS_HOSTNAME' in os.environ:
    DATABASES = {
        'default': {
            'ENGINE': 'django.db.backends.mysql',
            'NAME': os.environ['RDS_DB_NAME'],
```

```
        'USER': os.environ['RDS_USERNAME'],
        'PASSWORD': os.environ['RDS_PASSWORD'],
        'HOST': os.environ['RDS_HOSTNAME'],
        'PORT': os.environ['RDS_PORT'],
    }
}
```

## Outils et ressources Python

Il existe plusieurs endroits auxquels vous pouvez accéder pour obtenir une aide supplémentaire lors du développement de vos applications Python :

Ressource	Description
<a href="#">Boto (le kit SDK AWS pour Python)</a>	Installez Boto à l'aide de GitHub.
<a href="#">Forum de développement Python</a>	Posez vos questions et obtenez des commentaires.
<a href="#">Centre pour développeurs Python</a>	Guichet unique pour l'exemple de code, la documentation, les outils et les ressources supplémentaires.

# Création et déploiement d'applications Ruby sur Elastic Beanstalk

## Rubriques

- [Configuration de votre environnement de développement Ruby \(p. 384\)](#)
- [Utilisation de la plateforme Elastic Beanstalk Ruby \(p. 387\)](#)
- [Déploiement d'une application rails sur Elastic Beanstalk \(p. 391\)](#)
- [Déploiement d'une application sinatra sur Elastic Beanstalk \(p. 397\)](#)
- [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Ruby \(p. 401\)](#)

AWS Elastic Beanstalk pour Ruby permet de déployer, de gérer et de dimensionner facilement vos applications web Ruby à l'aide d'Amazon Web Services. Elastic Beanstalk est accessible à toute personne développant ou hébergeant une application web à l'aide de Ruby. Cette section fournit des instructions détaillées permettant de déployer un exemple d'application dans Elastic Beanstalk à l'aide de l'interface de ligne de commande (CLI) Elastic Beanstalk, puis de mettre à jour l'application pour utiliser les infrastructures d'application web [Rails](#) et [Sinatra](#).

Les rubriques de ce chapitre supposent que vous avez une certaine connaissance des environnements Elastic Beanstalk. Si vous n'avez jamais utilisé Elastic Beanstalk, essayez le [tutoriel de mise en route \(p. 3\)](#) pour acquérir les bases.

## Configuration de votre environnement de développement Ruby

Configurez un environnement de développement Ruby pour tester votre application localement avant de le déployer dans AWS Elastic Beanstalk. Cette rubrique décrit les étapes de configuration de l'environnement de développement et des liens vers les pages d'installation pour des outils utiles.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command  
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

Pour accéder aux outils et aux étapes de configuration courants qui s'appliquent à toutes les langues, veuillez consulter [Configuration de votre machine de développement pour une utilisation avec Elastic Beanstalk \(p. 1024\)](#)

#### Sections

- [Installation de Ruby \(p. 385\)](#)
- [Installation du kit AWS SDK pour Ruby \(p. 386\)](#)
- [Installation d'un IDE ou d'un éditeur de texte \(p. 386\)](#)

## Installation de Ruby

Installez GCC si vous n'avez pas de compilateur C. Sur Ubuntu, utilisez apt.

```
~$ sudo apt install gcc
```

Sur Amazon Linux, utilisez yum.

```
~$ sudo yum install gcc
```

Installez RVM pour gérer les installations de langage Ruby sur votre machine. Utilisez les commandes sur [rvm.io](#) pour obtenir les clés du projet et exécuter le script d'installation.

```
~$ gpg --keyserver hkp://keys.gnupg.net --recv-keys key1 key2  
~$ curl -sSL https://get.rvm.io | bash -s stable
```

Ce script installe RVM dans un dossier nommé `.rvm` dans votre répertoire utilisateur et modifie votre profil shell pour charger un script de configuration chaque fois que vous ouvrez un nouveau terminal. Chargez le script manuellement pour commencer.

```
~$ source ~/.rvm/scripts/rvm
```

Utilisez `rvm get head` pour obtenir la version la plus récente.

```
~$ rvm get head
```

Affichez les versions disponibles de Ruby.

```
~$ rvm list known  
# MRI Rubies  
[ruby-]2.1[.10]
```

```
[ruby-]2.2[.10]
[ruby-]2.3[.7]
[ruby-]2.4[.4]
[ruby-]2.5[.1]
...
```

Consultez [Ruby](#) dans le document Plateformes AWS Elastic Beanstalk pour trouver la dernière version de Ruby disponible sur une plateforme Elastic Beanstalk. Installez cette version.

```
~$ rvm install 2.5.1
Searching for binary rubies, this might take some time.
Found remote file https://rvm_io.global.ssl.fastly.net/binaries/ubuntu/16.04/x86_64/
ruby-2.5.1.tar.bz2
Checking requirements for ubuntu.
Requirements installation successful.
ruby-2.5.1 - #configure
ruby-2.5.1 - #download
...
```

Testez votre installation Ruby.

```
~$ ruby --version
ruby 2.5.1p57 (2018-03-29 revision 63029) [x86_64-linux]
```

## Installation du kit AWS SDK pour Ruby

Si vous devez gérer les ressources AWS à partir de votre application, installez AWS SDK for Ruby. Par exemple, avec le kit SDK pour Ruby, vous pouvez utiliser Amazon DynamoDB (DynamoDB) pour stocker les informations utilisateur et session sans créer de base de données relationnelle.

Installez le kit SDK pour Ruby et ses dépendances avec la commande gem.

```
$ gem install aws-sdk
```

Consultez la [page d'accueil AWS SDK for Ruby](#) pour plus d'informations et pour connaître les instructions d'installation.

## Installation d'un IDE ou d'un éditeur de texte

Les environnements de développement intégré (IDE) offrent un large éventail de fonctionnalités qui facilitent le développement d'applications. Si vous n'avez pas utilisé d'IDE pour le développement Ruby, testez Aptana et RubyMine, puis évaluez ce qui vous convient le mieux.

- [Installez Aptana](#)
- [RubyMine](#)

### Note

Un IDE peut ajouter des fichiers dans votre dossier de projet que vous pouvez ne pas souhaiter engager sur le contrôle de code source. Pour empêcher la validation de ces fichiers de contrôle de code source, utilisez `.gitignore` ou l'équivalent de votre outil de contrôle de source.

Si vous souhaitez simplement commencer le codage et que vous n'avez pas besoin de toutes les fonctionnalités d'un IDE, pensez à [installer Sublime Text](#).

## Utilisation de la plateforme Elastic Beanstalk Ruby

### Important

Les versions de plateforme Amazon Linux 2 sont fondamentalement différentes des versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2). Ces différentes générations de plateformes sont incompatibles à plusieurs égards. Si vous migrez vers une version de plateforme Amazon Linux 2, veillez à lire les informations de la section [the section called “Mise à niveau vers Amazon Linux 2” \(p. 508\)](#).

La plateforme Ruby AWS Elastic Beanstalk est un ensemble de [configurations d'environnement](#) pour les applications web Ruby qui peuvent s'exécuter derrière un serveur proxy nginx sous un serveur d'applications Puma. Chaque branche de la plateforme correspond à une version de Ruby.

Si vous utilisez RubyGems, vous pouvez [inclure un fichier Gemfile \(p. 390\)](#) dans votre bundle source pour installer des packages durant le déploiement.

Votre application peut s'exécuter sous un autre serveur d'applications, par exemple Passenger. Vous pouvez utiliser un [Procfile](#) pour démarrer un autre serveur d'applications et un [Gemfile](#) pour l'installer. Pour plus d'informations, consultez [the section called “Procfile” \(p. 390\)](#).

### Note

Si vous utilisez une branche de plateforme AMI Ruby (anciennement Amazon Linux 2), sachez qu'Elastic Beanstalk fournit deux versions de ces branches - avec Puma et avec Passenger. Si votre application a besoin du serveur d'applications Passenger, vous pouvez utiliser la branche de plateforme Passenger appropriée et vous n'avez pas besoin d'effectuer de configuration supplémentaire.

Elastic Beanstalk fournit des [options de configuration \(p. 658\)](#) que vous pouvez utiliser pour personnaliser le logiciel qui s'exécute sur les instances Amazon Elastic Compute Cloud (Amazon EC2) dans votre environnement Elastic Beanstalk. Vous pouvez configurer des variables d'environnement nécessaires pour votre application, activer la rotation des journaux sur Amazon S3 et mapper des dossiers dans la source de votre application contenant des fichiers statiques vers des chemins desservis par le serveur proxy. La plateforme prédéfinit également certaines variables d'environnement courantes associées à Rails et Rack pour faciliter la découverte et l'utilisation.

Des options de configuration sont disponibles dans la console Elastic Beanstalk pour [modifier la configuration d'un environnement en cours d'exécution \(p. 671\)](#). Pour éviter de perdre la configuration de votre environnement en le résiliant, vous pouvez utiliser des [configurations enregistrées \(p. 779\)](#) pour enregistrer vos paramètres et les appliquer par la suite à un autre environnement.

Pour enregistrer les paramètres dans votre code source, vous pouvez inclure des [fichiers de configuration \(p. 737\)](#). Les paramètres des fichiers de configuration sont appliquées chaque fois que vous créez un environnement ou que vous déployez votre application. Vous pouvez également utiliser des fichiers de configuration pour installer des packages, exécuter des scripts ou effectuer d'autres opérations de personnalisation d'instance lors des déploiements.

Les paramètres appliqués dans la console Elastic Beanstalk remplacent les mêmes paramètres des fichiers de configuration, s'ils existent. Cela vous permet d'utiliser les paramètres par défaut dans les fichiers de configuration et de les remplacer par des paramètres spécifiques à l'environnement dans la console. Pour de plus amples informations sur la priorité et les autres méthodes de modification des paramètres, veuillez consulter [Options de configuration \(p. 658\)](#).

Pour de plus amples informations sur les différentes manières d'étendre une plateforme Elastic Beanstalk basée sur Linux, veuillez consulter [the section called “Extension des plateformes Linux” \(p. 34\)](#).

## Configuration de votre environnement Ruby

Vous pouvez utiliser la console Elastic Beanstalk pour activer la rotation de journal sur Amazon S3 et configurer les variables que votre application peut lire depuis l'environnement.

Pour accéder aux paramètres de configuration du logiciel pour votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).

## Options du journal

La section Log Options (Options du journal) a deux paramètres :

- Instance profile (Profil d'instance) – Spécifie le profil d'instance qui est autorisé à accéder au compartiment Amazon S3 associé à votre application.
- Enable log file rotation to Amazon S3 (Permettre la rotation du fichier journal sur Amazon S3) – Indique si les fichiers journaux des instances Amazon EC2 de votre application doivent être copiés dans le compartiment Amazon S3 associé à votre application.

## Fichiers statiques

Pour améliorer les performances, la section des Fichiers statiques vous permet de configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Pour chaque répertoire, vous définissez le chemin virtuel sur le mappage de répertoires. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Pour de plus amples informations sur la configuration des fichiers statiques à l'aide de la console Elastic Beanstalk, veuillez consulter [the section called “Fichiers statiques” \(p. 790\)](#).

Par défaut, le serveur proxy dans un environnement Ruby sert tous les fichiers du dossier `public` sur le chemin d'accès `/public`, et tous les fichiers du sous-dossier `public/assets` sur le chemin d'accès `/assets`. Par exemple, si votre code source d'application contient un fichier nommé `logo.png` dans un dossier nommé `public`, le serveur proxy le sert aux utilisateurs dans `subdomain.elasticbeanstalk.com/public/logo.png`. Si `logo.png` se trouve dans un dossier nommé `assets` à l'intérieur du dossier `public`, le serveur proxy le sert à `subdomain.elasticbeanstalk.com/assets/logo.png`. Vous pouvez configurer d'autres mappages comme expliqué dans cette section.

## Propriétés de l'environnement

La section Propriétés de l'environnement vous permet de spécifier des paramètres de configuration de l'environnement sur les instances Amazon EC2 exécutant votre application. Les propriétés de l'environnement sont passées en tant que paires clé-valeur à l'application.

La plateforme Ruby définit les propriétés suivantes pour la configuration de l'environnement :

- `BUNDLE_WITHOUT` – Une liste séparée par des deux-points de groupes à ignorer lors de l'[installation de dépendances](#) à partir d'un `Gemfile`.
- `BUNDLER_DEPLOYMENT_MODE` – Définissez la valeur sur `true` (valeur par défaut) pour installer les dépendances en [mode de déploiement](#) à l'aide de Bundler. Définissez la valeur sur `false` pour exécuter `bundle install` en mode de développement.

### Note

Cette propriété d'environnement n'est pas définie sur les branches de la plateforme Amazon Linux AMI Ruby (auparavant Amazon Linux 2).

- RAILS\_SKIP\_ASSET\_COMPILATION – Définissez ce paramètre sur `true` pour ignorer l'exécution de `rake assets:precompile` durant le déploiement.
- RAILS\_SKIP.Migrations – Définissez ce paramètre sur `true` pour ignorer l'exécution de `rake db:migrate` durant le déploiement.
- RACK\_ENV – Spécifiez l'étape de l'environnement pour Rack. Par exemple, `development`, `production` ou `test`.

Dans l'environnement Ruby en cours d'exécution dans Elastic Beanstalk, les variables d'environnement sont accessibles à l'aide de l'objet `ENV`. Par exemple, vous pouvez lire une propriété nommée `API_ENDPOINT` sur une variable avec le code suivant :

```
endpoint = ENV['API_ENDPOINT']
```

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Espaces de noms de configuration Ruby

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

Vous pouvez utiliser l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles` pour configurer le proxy d'environnement afin de traiter les fichiers statiques. Vous définissez les mappages des chemins virtuels vers les répertoires d'applications.

La plate-forme Ruby ne définit aucun espace de noms spécifique à la plate-forme. Au lieu de cela, elle définit les propriétés d'environnement pour les options Rails et Rack communes.

Le fichier de configuration suivant spécifie une option de fichiers statiques qui mappe un répertoire nommé `staticimages` au chemin d'accès `/images`, définit chacune des propriétés d'environnement définies par la plate-forme et définit une propriété d'environnement supplémentaire nommée `LOGGING`.

Example `.ebextensions/ruby-settings.config`

```
option_settings:  
  aws:elasticbeanstalk:environment:proxy:staticfiles:  
    /images: staticimages  
  aws:elasticbeanstalk:application:environment:  
    BUNDLE_WITHOUT: test  
    BUNDLER_DEPLOYMENT_MODE: true  
    RACK_ENV: development  
    RAILS_SKIP_ASSET_COMPILATION: true  
    RAILS_SKIP.Migrations: true  
    LOGGING: debug
```

### Note

La propriété d'environnement `BUNDLER_DEPLOYMENT_MODE` et l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles` ne sont pas définis sur les branches de la plateforme Amazon Linux AMI Ruby (anciennement Amazon Linux 2).

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Installation de packages avec un Gemfile

Utilisez un fichier `Gemfile` à la racine de votre source de projet pour utiliser RubyGems pour installer des packages dont votre application a besoin.

Example `Gemfile`

```
source "https://rubygems.org"
gem 'sinatra'
gem 'json'
gem 'rack-parser'
```

Lorsqu'un fichier `Gemfile` est présent, Elastic Beanstalk exécute `bundle install` pour installer les dépendances.

## Configuration du processus de l'application avec un Procfile

Pour spécifier la commande qui démarre votre application Ruby, incluez un fichier appelé `Procfile` à la racine de votre solution groupée source.

Note

Elastic Beanstalk ne prend pas en charge cette fonctionnalité sur les branches de plateforme Amazon Linux AMI Ruby (auparavant Amazon Linux 2). Les branches de plateforme dont les noms contiennent Puma ou Passenger, quelle que soit leur version Ruby, précèdent Amazon Linux 2 et ne prennent pas en charge la fonctionnalité `Procfile`.

Pour plus d'informations sur l'écriture et l'utilisation d'un `Procfile`, développez la section `Buildfile` et `Procfile` dans [the section called “Extension des plateformes Linux” \(p. 34\)](#).

Lorsque vous ne fournissez pas de `Procfile`, Elastic Beanstalk génère le fichier par défaut suivant, qui suppose que vous utilisez le serveur d'applications Puma préinstallé.

```
web: puma -C /opt/elasticbeanstalk/config/private/pumaconf.rb
```

Si vous souhaitez utiliser votre propre serveur Puma fourni, vous pouvez l'installer à l'aide d'un [Gemfile \(p. 390\)](#). L'exemple suivant de `Procfile` indique comment démarrer :

Example `Procfile`

```
web: bundle exec puma -C /opt/elasticbeanstalk/config/private/pumaconf.rb
```

Si vous souhaitez utiliser le serveur d'applications Passenger, utilisez les exemples de fichiers suivants pour configurer votre environnement Ruby pour installer et utiliser Passenger.

1. Utilisez ce fichier d'exemple pour installer Passenger.

Example `Gemfile`

```
source 'https://rubygems.org'
gem 'passenger'
```

2. Utilisez cet exemple de fichier pour demander à Elastic Beanstalk de démarrer Passenger.

#### Example Procfile

```
web: bundle exec passenger start /var/app/current --socket /var/run/puma/my_app.sock
```

#### Note

Aucun changement ne doit être apporté à la configuration du serveur proxy nginx pour utiliser Passenger. Pour utiliser d'autres serveurs d'applications, vous devrez peut-être personnaliser la configuration nginx pour transférer correctement les demandes à votre application.

## Déploiement d'une application rails sur Elastic Beanstalk

Rails est une infrastructure open source, modèle-vue-contrôleur (MVC) pour Ruby. Ce didacticiel vous guide tout au long du processus de génération d'une application Rails et de son déploiement dans un environnement AWS Elastic Beanstalk.

#### Sections

- [Prerequisites \(p. 391\)](#)
- [Lancer un environnement Elastic Beanstalk \(p. 392\)](#)
- [Installation de Rails et génération d'un site web \(p. 393\)](#)
- [Configuration des paramètres Rails \(p. 395\)](#)
- [Déploiement de votre application \(p. 396\)](#)
- [Cleanup \(p. 396\)](#)
- [Étapes suivantes \(p. 397\)](#)

## Prerequisites

### Connaissances de base sur Elastic Beanstalk

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

### Ligne de commande

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

### Dépendances Rails

L'infrastructure Rails a les dépendances suivantes. Veillez à ce qu'elles soient toutes installées.

- Ruby 2.2.2 ou version ultérieure – Pour obtenir les instructions d'installation, consultez [Configuration de votre environnement de développement Ruby \(p. 384\)](#).

Dans ce tutoriel, nous utilisons Ruby 2.5.1 et la version correspondante de la plateforme Elastic Beanstalk.

- Node.js – Pour obtenir des instructions d'installation, consultez [Installation de Node.js via le gestionnaire de package](#).
- Yarn – Pour obtenir des instructions d'installation, veuillez consulter [Installation](#) sur le site web Yarn.

## Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme Ruby et acceptez les paramètres par défaut et l'exemple de code.

### Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créer une application.

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibrEUR de charge à

atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.

- Équilibrer de charge – Équilibrer de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrer de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrer de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrer de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Installation de Rails et génération d'un site web

Installez Rails et ses dépendances avec la commande `gem`.

```
~$ gem install rails
Fetching: concurrent-ruby-1.0.5.gem (100%)
Successfully installed concurrent-ruby-1.0.5
Fetching: i18n-1.0.1.gem (100%)
Successfully installed i18n-1.0.1
...
```

Testez votre installation Rails.

```
~$ rails --version
Rails 5.2.0
```

Utilisez `rails new` avec le nom de l'application pour créer un projet Rails.

```
~$ rails new ~/eb-rails
```

Rails crée un répertoire avec le nom spécifié, génère tous les fichiers nécessaires pour exécuter un exemple de projet localement et exécute ensuite Bundler pour installer toutes les dépendances (Gems) définies dans le Gemfile du projet.

Testez votre installation Rails en exécutant le projet par défaut localement.

```
~$ cd eb-rails
eb-rails $ rails server
=> Booting Puma
=> Rails 5.2.0 application starting in development
=> Run `rails server -h` for more startup options
Puma starting in single mode...
* Version 3.11.4 (ruby 2.5.1-p57), codename: Love Song
* Min threads: 5, max threads: 5
* Environment: development
* Listening on tcp://0.0.0.0:3000
Use Ctrl+C to stop
...
```

Ouvrez <http://localhost:3000> dans un navigateur web pour afficher le projet par défaut en action.



Cette page est uniquement visible en mode développement. Ajoutez du contenu à la première page de l'application afin de prendre en charge le déploiement de production sur Elastic Beanstalk. Utilisez `rails generate` pour créer un contrôleur, un itinéraire et une vue pour votre page d'accueil.

```
~/eb-rails$ rails generate controller WelcomePage welcome
      create  app/controllers/welcome_page_controller.rb
      route   get 'welcome_page/welcome'
      invoke   erb
      create    app/views/welcome_page
      create    app/views/welcome_page/welcome.html.erb
      invoke   test_unit
```

```
create  test/controllers/welcome_page_controller_test.rb
invoke  helper
create  app/helpers/welcome_page_helper.rb
invoke  test_unit
invoke  assets
invoke  coffee
create    app/assets/javascripts/welcome_page.coffee
invoke  scss
create    app/assets/stylesheets/welcome_page.scss.
```

Cela vous permet de disposer de tout ce dont vous avez besoin pour accéder à la page à l'adresse /welcome\_page/welcome. Avant de publier les modifications, toutefois, modifiez le contenu dans la vue et ajoutez un itinéraire pour que cette page s'affiche au niveau supérieur du site.

Utilisez un éditeur de texte pour modifier le contenu dans app/views/welcome\_page/welcome.html.erb. Pour cet exemple, vous utiliserez cat pour remplacer simplement le contenu du fichier existant.

Example app/views/welcome\_page/welcome.html.erb

```
<h1>Welcome!</h1>
<p>This is the front page of my first Rails application on Elastic Beanstalk.</p>
```

Enfin, ajoutez l'itinéraire suivant à config/routes.rb:

Example config/routes.rb

```
Rails.application.routes.draw do
  get 'welcome_page/welcome'
  root 'welcome_page#welcome'
```

Cela indique à Rails d'acheminer les demandes à la racine du site web à la méthode de bienvenue du contrôleur de la page d'accueil, qui affiche le contenu dans la vue d'accueil (welcome.html.erb).

## Configuration des paramètres Rails

Utilisez la console Elastic Beanstalk pour configurer Rails avec les propriétés d'environnement. Définissez la propriété d'environnement SECRET\_KEY\_BASE sous la forme d'une chaîne de 256 caractères alphanumériques au plus.

Rails utilise cette propriété pour créer des clés. Par conséquent, vous devez la garder secrète et ne pas la stocker dans le contrôle de code source en texte brut. Au lieu de cela, vous la fournissez au code Rails dans votre environnement via une propriété d'environnement.

Pour configurer les propriétés d'environnement dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Sous Propriétés de l'environnement, saisissez les paires clé/valeur.

6. Choisissez Apply.

Maintenant, vous êtes prêt à déployer le site dans votre environnement.

## Déploiement de votre application

Créez un [bundle source](#) (p. 415) contenant les fichiers créés par Rails. La commande suivante permet de créer une solution groupée source nommée `rails-default.zip`.

```
~/eb-rails$ zip ..../rails-default.zip -r * .[^.]*
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer Rails dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances](#)

Amazon EC2 (p. 538), les instances de base de données (p. 617), les équilibreurs de charge (p. 562), les groupes de sécurité et les alarmes (p. 562).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

## Étapes suivantes

Pour plus d'informations sur Rails, visitez [rubyonrails.org](#).

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez [configurer un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activer HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Déploiement d'une application Sinatra sur Elastic Beanstalk

Cette procédure détaillée explique comment déployer une application web simple [Sinatra](#) sur AWS Elastic Beanstalk.

### Prerequisites

Ce tutoriel suppose que vous connaissez les opérations de base Elastic Beanstalk et la console Elastic Beanstalk. Si ce n'est pas déjà fait, suivez les instructions dans [Mise en route avec Elastic Beanstalk \(p. 3\)](#) pour lancer votre premier environnement Elastic Beanstalk.

Pour suivre les procédures décrites dans ce manuel, vous aurez besoin d'un shell ou d'un terminal de ligne de commande pour exécuter des commandes. Dans les listes, les commandes sont précédées d'un symbole d'invite (\$) et du nom du répertoire actuel, le cas échéant.

```
~/eb-project$ this is a command
this is output
```

Sous Linux et macOS, vous pouvez utiliser le shell et le gestionnaire de package de votre choix. Sur Windows 10, vous pouvez [installer le sous-système Windows pour Linux](#) afin d'obtenir une version intégrée à Windows d'Ubuntu et Bash.

Sinatra nécessite Ruby 1.9 ou version postérieure. Dans ce tutoriel, nous utilisons Ruby 2.5.1 et la version correspondante de la plateforme Elastic Beanstalk. Installez Ruby en suivant les instructions de la section [Configuration de votre environnement de développement Ruby \(p. 384\)](#).

## Lancer un environnement Elastic Beanstalk

Utilisez la console Elastic Beanstalk pour créer un environnement Elastic Beanstalk. Choisissez la plateforme Ruby et acceptez les paramètres par défaut et l'exemple de code.

Pour lancer un environnement (console)

1. Ouvrez la console Elastic Beanstalk en utilisant le lien préconfiguré suivant :  
[console.aws.amazon.com/elasticbeanstalk/home#/newApplication?  
applicationName=tutorials&environmentType=LoadBalanced](https://console.aws.amazon.com/elasticbeanstalk/home#/newApplication?applicationName=tutorials&environmentType=LoadBalanced)
2. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.
3. Pour l'option Code de l'application, choisissez Exemple d'application.
4. Choisissez Vérifier et lancer.
5. Passez en revue les options disponibles. Choisissez l'option disponible que vous souhaitez utiliser et, une fois que vous êtes prêt, choisissez Créez une application.

La création d'un environnement; prend environ 5 minutes et crée les ressources suivantes :

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibrEUR de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.

- Équilibrer de charge – Équilibrer de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrer de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrer de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrer de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.
- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Toutes ces ressources sont gérées par Elastic Beanstalk. Lorsque vous arrêtez votre environnement, Elastic Beanstalk arrête toutes les ressources qu'il contient.

#### Note

Le compartiment Amazon S3 créé par Elastic Beanstalk est partagé entre les environnements et n'est pas supprimé lors de l'arrêt de l'environnement. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#).

## Écriture d'un site web Sinatra

Pour créer et déployer une application Sinatra

1. Créez un fichier de configuration nommé config.ru avec le contenu suivant.

#### Example config.ru

```
require './helloworld'  
run Sinatra::Application
```

2. Créez un fichier de code Ruby nommé helloworld.rb avec le contenu suivant.

#### Example helloworld.rb

```
require 'sinatra'  
get '/' do  
  "Hello World!"  
end
```

3. Créez un Gemfile avec le contenu suivant.

#### Example Gemfile

```
source 'https://rubygems.org'  
gem 'sinatra'
```

## Déploiement de votre application

Créez un [bundle source \(p. 415\)](#) contenant vos fichiers source. La commande suivante permet de créer une solution groupée source nommée `sinatra-default.zip`.

```
~/eb-sinatra$ zip .. /sinatra-default.zip -r * .[^.]*
```

Téléchargez le bundle source sur Elastic Beanstalk pour déployer Sinatra dans votre environnement.

Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

## Cleanup

Lorsque vous avez fini d'utiliser Elastic Beanstalk, vous pouvez arrêter votre environnement. Elastic Beanstalk arrête toutes les ressources AWS associées à votre environnement, telles que les [instances Amazon EC2 \(p. 538\)](#), les [instances de base de données \(p. 617\)](#), les [équilibreurs de charge \(p. 562\)](#), les groupes de sécurité et les [alarmes \(p. 562\)](#).

Pour arrêter votre environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Avec Elastic Beanstalk, vous pouvez facilement créer un nouvel environnement pour votre application à tout moment.

## Étapes suivantes

Pour plus d'informations sur Sinatra, consultez [sinatrarb.com](http://sinatrarb.com).

À mesure que vous continuez à développer votre application, vous souhaiterez probablement disposer d'une solution pour gérer des environnements et déployer votre application sans devoir créer un fichier .zip et le télécharger manuellement sur la console Elastic Beanstalk. L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) fournit des commandes faciles à utiliser pour la création, la configuration et le déploiement d'applications dans des environnements Elastic Beanstalk à partir de la ligne de commande.

Enfin, si vous prévoyez d'utiliser votre application dans un environnement de production, vous devez [configurer un nom de domaine personnalisé \(p. 656\)](#) pour votre environnement et [activer HTTPS \(p. 792\)](#) pour des connexions sécurisées.

## Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Ruby

Vous pouvez utiliser une instance de base de données Amazon Relational Database Service (Amazon RDS) pour stocker les données collectées et modifiées par votre application. La base de données peut être associée à votre environnement et gérée par Elastic Beanstalk, ou créée et gérée en externe.

Si vous utilisez Amazon RDS pour la première fois, [ajoutez une instance de base de données \(p. 401\)](#) à un environnement de test avec la console de gestion Elastic Beanstalk et assurez-vous que votre application peut s'y connecter.

Pour vous connecter à une base de données, [ajoutez la carte \(p. 402\)](#) à votre application et [configurez une connexion \(p. 402\)](#) avec les propriétés d'environnement fournies par Elastic Beanstalk. La configuration et le code de connexion varient selon le moteur de base de données et l'infrastructure que vous utilisez.

### Sections

- [Ajout d'une instance de base de données à votre environnement \(p. 401\)](#)
- [Téléchargement d'une carte \(p. 402\)](#)
- [Connexion à une base de données \(p. 402\)](#)

## Ajout d'une instance de base de données à votre environnement

Pour ajouter une instance DB à votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Choisissez Apply.

L'ajout d'une instance DB prend environ 10 minutes. Une fois la mise à jour de l'environnement terminée, le nom d'hôte de l'instance DB et les autres informations de connexion sont disponibles dans votre application, via les propriétés d'environnement suivantes :

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

Pour de plus amples informations sur la configuration d'une instance de base de données interne, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

## Téléchargement d'une carte

Ajoutez la carte de base de données au [Gemfile \(p. 390\)](#) de votre projet.

Example Gemfile – Rails avec MySQL

```
source 'https://rubygems.org'
gem 'puma'
gem 'rails', '4.1.8'
gem 'mysql2'
```

Fichiers Gem de carte courants pour Ruby

- MySQL – mysql2
- PostgreSQL – pg
- Oracle – activerecord-oracle\_enhanced-adapter
- SQL Server – sql\_server

## Connexion à une base de données

Elastic Beanstalk fournit des informations de connexion pour les instances de base de données attachées dans les propriétés de l'environnement. Utilisez `ENV['VARIABLE']` pour lire les propriétés et configurer une connexion de base de données.

Example config/database.yml – Ruby sur une configuration de base de données Rails (MySQL)

```
production:
```

```
adapter: mysql2
encoding: utf8
database: <%= ENV['RDS_DB_NAME'] %>
username: <%= ENV['RDS_USERNAME'] %>
password: <%= ENV['RDS_PASSWORD'] %>
host: <%= ENV['RDS_HOSTNAME'] %>
port: <%= ENV['RDS_PORT'] %>
```

# Didacticiels et exemples

Des didacticiels sont proposés tout au long du Guide du développeur AWS Elastic Beanstalk, pour différents langages et différentes infrastructures. Nous actualisons cette liste en permanence en y ajoutant les didacticiels que nous créons ou mettons à jour. Les mises à jour les plus récentes sont affichées en premier.

Ces didacticiels s'adressent aux utilisateurs de niveau intermédiaire et peuvent ne pas contenir d'instructions pour des procédures de base, telles que l'inscription à AWS. Si vous utilisez AWS ou Elastic Beanstalk pour la première fois, consultez notre [procédure de démarrage \(p. 3\)](#) pour suivre les étapes permettant de rendre votre premier environnement Elastic Beanstalk opérationnel.

- Ruby on Rails - [Déploiement d'une application rails sur Elastic Beanstalk \(p. 391\)](#)
- Ruby et Sinatra - [Déploiement d'une application sinatra sur Elastic Beanstalk \(p. 397\)](#)
- Configuration haute disponibilité PHP et MySQL - [Déploiement d'une application PHP haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk \(p. 316\)](#)
- PHP et Laravel - [Déploiement d'une application Laravel sur Elastic Beanstalk \(p. 298\)](#)
- PHP et CakePHP - [Déploiement d'une application CakePHP sur Elastic Beanstalk \(p. 305\)](#)
- Configuration haute disponibilité PHP et Drupal - [Déploiement d'un site web Drupal haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk \(p. 339\)](#)
- Configuration haute disponibilité PHP et WordPress - [Déploiement d'un site Web WordPress haute disponibilité avec une base de données Amazon RDS externe vers Elastic Beanstalk \(p. 325\)](#)
- Configuration haute disponibilité Node.js avec DynamoDB - [Déploiement d'une application Node.js avec DynamoDB vers Elastic Beanstalk \(p. 278\)](#)
- ASP.NET Core - [Tutorial : Déploiement d'une application ASP.NET Core avec Elastic Beanstalk \(p. 212\)](#)
- Python et Flask - [Déploiement d'une application Flask sur Elastic Beanstalk \(p. 365\)](#)
- Python et Django - [Déploiement d'une application Django sur Elastic Beanstalk \(p. 371\)](#)
- Node.js et Express - [Déploiement d'une application Express sur Elastic Beanstalk \(p. 265\)](#)
- Docker, PHP et nginx - [Environnements Docker multiconteneurs avec la console Elastic Beanstalk \(p. 73\)](#)
- .NET Framework (IIS et ASP.NET) - [Tutorial : Apprenez à déployer un exemple d'application .NET à l'aide d'Elastic Beanstalk \(p. 206\)](#)

Vous pouvez utiliser les liens suivants afin de télécharger les exemples d'application utilisés par Elastic Beanstalk lorsque vous créez un environnement sans fournir de bundle de fichiers source :

- Docker – [docker.zip](#)
- Docker multi-conteneurs – [docker-multicontainer-v2.zip](#)
- Docker avec plateforme préconfigurée (Glassfish) – [docker-glassfish-v1.zip](#)
- Go – [go.zip](#)
- Corretto – [corretto.zip](#)
- Tomcat – [tomcat.zip](#)
- .NET Core sous Linux – [dotnet-core-linux.zip](#)
- .NET – [dotnet-asp-v1.zip](#)
- Node.js – [nodejs.zip](#)
- PHP – [php.zip](#)
- Python – [python.zip](#)

- Ruby – [ruby.zip](#)

Des exemples d'application plus complexes, qui illustrent l'utilisation d'autres infrastructures web, bibliothèques et outils, sont disponibles sur GitHub sous forme de projets open source :

- [WordPress à charge équilibrée \(tutoriel \(p. 325\)\)](#) – Fichiers de configuration pour l'installation de WordPress en toute sécurité et son exécution dans un environnement Elastic Beanstalk à charge équilibrée.
- [Drupal à charge équilibrée \(tutoriel \(p. 339\)\)](#) – Fichiers de configuration et instructions pour l'installation de Drupal en toute sécurité et son exécution dans un environnement Elastic Beanstalk à charge équilibrée.
- [Scorekeep](#) – API web RESTful qui utilise l'infrastructure Spring et le kit AWS SDK for Java pour fournir une interface permettant de créer et gérer des utilisateurs, des sessions et des jeux. Cette API est regroupée avec une application web Angular 1.5 qui utilise l'API sur HTTP. Elle inclut les branches qui montrent l'intégration à Amazon Cognito, AWS X-Ray et Amazon Relational Database Service.

L'application utilise des fonctionnalités de la plateforme Java SE pour télécharger des dépendances et des instances intégrées, ce qui réduit la taille du bundle de fichiers source. L'application inclut également des fichiers de configuration nginx qui remplacent la configuration par défaut pour servir l'application web frontale de manière statique sur le port 80 via le proxy et acheminer les requêtes vers des chemins sous /api vers l'API s'exécutant sur localhost:5000.

- [Does it Have Snakes?](#) - Application Tomcat qui illustre l'utilisation de RDS au sein d'une application web Java EE dans Elastic Beanstalk. Ce projet décrit comment utiliser les fichiers de configuration Servlets, JSP, Simple Tag Support, Tag Files, JDBC, SQL, Log4J, Bootstrap, Jackson et Elastic Beanstalk.
- [Locust Load Generator](#) - Ce projet décrit comment utiliser les fonctionnalités de la plateforme Java SE pour installer et exécuter Locust, un outil de test de charge développé en Python. Ce projet inclut des fichiers de configuration qui installent et configurent Locust, un script de construction qui configure une table DynamoDB et un Procfile qui exécute Locust.
- [Share Your Thoughts \(tutoriel \(p. 316\)\)](#) - Application PHP qui décrit l'utilisation de MySQL sur Amazon RDS, Composer et les fichiers de configuration.
- [A New Startup \(tutoriel \(p. 278\)\)](#) - Exemple d'application Node.js qui décrit l'utilisation de DynamoDB, du kit SDK AWS pour JavaScript dans Node.js, de la gestion des packages npm et des fichiers de configuration.

# Gestion et configuration d'applications Elastic Beanstalk

Lorsque vous commencez à utiliser AWS Elastic Beanstalk, la première étape consiste à créer une application, qui représente votre application web dans AWS. Dans Elastic Beanstalk, une application sert de conteneur pour les environnements qui exécutent votre application web, et les versions du code source de votre application web, les configurations enregistrées, les journaux et autres artefacts que vous créez lorsque vous utilisez Elastic Beanstalk.

Pour créer une application

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le volet de navigation, choisissez Applications, puis Create a new application (Créer une nouvelle application).
3. Utilisez le formulaire à l'écran pour fournir un nom d'application.
4. Le cas échéant, fournissez une description et ajoutez les clés et valeurs de balise.
5. Sélectionnez Créer.

The screenshot shows the 'Create new application' wizard in the AWS Elastic Beanstalk console. It consists of two main sections: 'Application information' and 'Tags'.  
**Application information:** This section contains fields for 'Application Name' (a required input) and 'Description' (an optional input). A note below the name field specifies a maximum length of 100 characters, not including forward slash (/).  
**Tags:** This section allows adding up to 50 tags. It includes fields for 'Key' and 'Value', an 'Add tag' button, and a note indicating 50 remaining tags. A 'Remove' button is also visible.  
At the bottom right of the wizard, there is a 'Create application' button.

Une fois que vous avez créé une application, la console vous invite à créer un environnement pour cette application. Pour obtenir des informations détaillées sur toutes les options disponibles, veuillez consulter [Création d'un environnement Elastic Beanstalk \(p. 439\)](#).

Si vous n'avez plus besoin d'une application, vous pouvez la supprimer.

#### Warning

La suppression d'une application entraîne la mise hors service de tous les environnements associés, ainsi que la suppression de toutes les versions d'application et des configurations enregistrées appartenant à l'application.

### Pour supprimer une application

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le volet de navigation, choisissez Applications, puis sélectionnez votre application dans la liste.
3. Choisissez Actions, puis Delete application (Supprimer l'application).

### Rubriques

- [Console de gestion d'application Elastic Beanstalk \(p. 408\)](#)
- [Gestion des versions d'application \(p. 409\)](#)
- [Création d'une solution groupée d'application \(p. 415\)](#)
- [Balisage des ressources d'application Elastic Beanstalk \(p. 423\)](#)

## Console de gestion d'application Elastic Beanstalk

Vous pouvez utiliser la console AWS Elastic Beanstalk pour gérer des applications, des versions d'application et des configurations enregistrées.

### Pour accéder à la console de gestion des applications

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

La page de présentation de l'application affiche une liste avec une présentation de tous les environnements associés à l'application.

The screenshot shows the AWS Elastic Beanstalk Application Management interface. On the left, there's a sidebar with 'Environments' and 'Applications' sections, and a collapsed 'getting-started-app' section containing 'Application versions' and 'Saved configurations'. Below that is a 'Recent environments' section with 'GettingStartedApp-env' and 'GettingStartedApp-Windows'. The main area has a breadcrumb trail 'Elastic Beanstalk > Applications > getting-started-app'. The title is 'Application 'getting-started-app' environments'. A search bar says 'Filter results matching the display values'. A table lists two environments:

Environment name	Health	Date created	Last modified	URL	Platform
GettingStartedApp-env	OK	2020-01-28 12:06:50 UTC-0800	2020-01-30 15:02:35 UTC-0800	GettingStartedApp-env.bn7dx222kw.us-east-2.elasticbeanstalk.com	Tomcat running Linux
GettingStartedApp-Windows	OK	2020-01-28 16:34:29 UTC-0800	2020-01-28 16:38:20 UTC-0800	GettingStartedApp-Windows.bn7dx222kw.us-east-2.elasticbeanstalk.com	IIS 10.0 Windows

3. Vous disposez de plusieurs façons de continuer :

- a. Choisissez le menu déroulant Actions, puis choisissez l'une des actions de gestion de l'application. Pour lancer un environnement dans cette application, vous pouvez directement choisir Create a new environment (Créer un nouvel environnement). Pour plus d'informations, consultez [the section called "Création d'environnements" \(p. 439\)](#).

- b. Choisissez un nom d'environnement pour accéder à la [console de gestion des environnements \(p. 429\)](#) pour cet environnement, où vous pouvez configurer, surveiller ou gérer ce dernier.
- c. Choisissez Application versions (Versions d'application) en suivant le nom de l'application dans le volet de navigation pour afficher et gérer les versions d'application de votre application.

Une version d'application est une version téléchargée de votre code d'application. Vous pouvez télécharger de nouvelles versions, déployer une version existante sur n'importe quel environnement de l'application, ou supprimer d'anciennes versions. Pour de plus amples informations, veuillez consulter [Gestion des versions d'application \(p. 409\)](#).

- d. Choisissez Saved configurations (Configurations enregistrées) en suivant le nom de l'application dans le volet de navigation pour afficher et gérer les configurations enregistrées à partir d'environnements en cours d'exécution.

Une configuration enregistrée est un ensemble de paramètres que vous pouvez utiliser pour restaurer les paramètres d'un environnement à un état antérieur ou créer un environnement avec les mêmes paramètres. Pour de plus amples informations, veuillez consulter [Utilisation des configurations enregistrées par Elastic Beanstalk \(p. 779\)](#).

## Gestion des versions d'application

Elastic Beanstalk crée une version d'application chaque fois que vous chargez le code source. Généralement, cette situation se produit lorsque vous créez un environnement, ou lorsque vous chargez et déployez le code via la [console de gestion de l'environnement \(p. 429\)](#) ou l'[interface de ligne de commande EB \(p. 1027\)](#). Elastic Beanstalk supprime ces versions de l'application en fonction de la stratégie de cycle de vie de l'application et lorsque vous supprimez l'application. Pour plus d'informations sur la stratégie de cycle de vie de l'application, consultez [Configuration des paramètres du cycle de vie des versions d'application \(p. 411\)](#).

Vous pouvez également télécharger un bundle de fichiers source sans le déployer depuis la [console de gestion des applications \(p. 408\)](#) ou avec la commande de l'interface de ligne de commande EB `eb appversion` ([p. 1064](#)). Elastic Beanstalk stocke les lots source dans Amazon Simple Storage Service (Amazon S3) et ne les supprime pas automatiquement.

Vous pouvez appliquer des balises à une version d'application lorsque vous la créez, et modifier les balises de versions d'applications existantes. Pour plus d'informations, consultez [Balisage des versions d'application \(p. 413\)](#).

### Pour créer une version d'application

Vous pouvez également créer une nouvelle version d'application à l'aide de l'interface de ligne de commande EB. Pour plus d'informations, veuillez consulter [eb appversion \(p. 1064\)](#) au chapitre Commandes de l'interface de ligne de commande EB.

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

#### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).

4. Choisissez Upload (Charger). Utilisez le formulaire à l'écran pour télécharger le [bundle source de votre application \(p. 415\)](#).

Note

La taille de fichier du bundle de fichiers source ne doit pas dépasser 512 Mo.

5. Le cas échéant, fournissez une brève description, et ajouter les clés et valeurs de balise.
6. Choisissez Upload (Charger).

Le fichier spécifié est associé à votre application. Vous pouvez déployer la version d'application dans un environnement nouveau ou existant.

Au fil du temps, votre application peut accumuler un grand nombre de versions de l'application. Pour économiser de l'espace de stockage et éviter d'atteindre le [quota de versions de l'application](#), il est conseillé de supprimer les versions dont vous n'avez plus besoin.

Note

La suppression d'une version d'application n'a aucune incidence sur les environnements qui exécutent actuellement cette version.

#### Pour supprimer une version d'application

Vous pouvez également supprimer une version d'application à l'aide de l'interface de ligne de commande EB. Pour plus d'informations, veuillez consulter [eb appversion \(p. 1064\)](#) au chapitre Commandes de l'interface de ligne de commande EB.

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
4. Sélectionnez une ou plusieurs versions de l'application à supprimer.

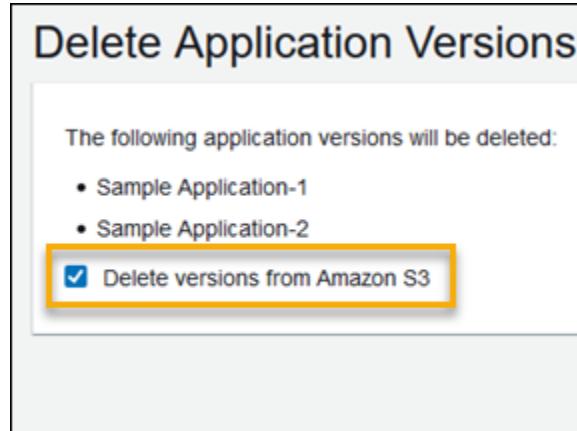
The screenshot shows the 'Application versions' section of the AWS Elastic Beanstalk console. It lists five application versions in a table:

Version label	Description	Date created	Source
Sample Application-3		2020-01-30T15:02:09-08:00	2020030Wv-java-tomcat-v3.zip
Sample Application-2		2020-01-28T15:16:47-08:00	2020028J18-java-tomcat-v3.zip
Sample Application-1		2019-12-06T14:56:36-08:00	2019340FRL-eb-demo-php-simple-app-v1.3.zip
Sample Application		2019-11-01T15:12:51-07:00	Sample Application

Two rows are highlighted with a yellow box and have checkboxes checked in the first column. The 'Actions' menu in the top right corner is open, showing 'Delete' with a cursor icon over it, and other options like 'Deploy' and 'Manage tags'.

5. Choisissez Actions, puis Delete (Supprimer).

6. (Facultatif) Pour laisser le bundle des fichiers source d'application pour ces versions d'application dans votre compartiment Amazon Simple Storage Service (Amazon S3), désactivez la case Delete versions from Amazon S3 (Supprimer les versions d'Amazon S3).



7. Sélectionnez Supprimer.

Vous pouvez également configurer Elastic Beanstalk pour supprimer automatiquement les anciennes versions en configurant les paramètres du cycle de vie de la version d'application. Si vous configurez ces paramètres de cycle de vie, ils sont appliqués lorsque vous créez de nouvelles versions de l'application. Par exemple, si vous configurez un maximum de 25 versions d'application, Elastic Beanstalk supprime la version la plus ancienne lorsque vous chargez une 26e version. Si vous définissez une ancienneté maximale de 90 jours, toute version dont l'ancienneté dépasse 90 jours est supprimée lorsque vous chargez une nouvelle version. Pour plus d'informations, consultez [the section called "Cycle de vie des versions" \(p. 411\)](#).

Si vous ne choisissez pas de supprimer le lot source d'Amazon S3, Elastic Beanstalk supprime toujours la version de ses enregistrements. Toutefois, le bundle source est laissé dans votre [compartiment de stockage Elastic Beanstalk \(p. 1003\)](#). Le quota de versions d'application s'applique uniquement aux versions des pistes Elastic Beanstalk. Vous pouvez donc supprimer des versions pour respecter le quota, mais conserver tous les bundles de fichiers source dans Amazon S3.

#### Note

Le quota de versions d'application ne s'applique pas aux bundles de fichiers source, mais vous pouvez cependant encourir des frais Amazon S3 et conserver des informations personnelles même si vous n'en avez plus besoin. Elastic Beanstalk ne supprime jamais les bundles de fichiers source automatiquement. Vous devez supprimer les bundles de fichiers source lorsque vous n'en avez plus besoin.

## Configuration des paramètres du cycle de vie des versions d'application

Chaque fois que vous téléchargez une nouvelle version de votre application avec la console Elastic Beanstalk ou l'interface de ligne de commande EB, Elastic Beanstalk crée une [version d'application \(p. 409\)](#). Si vous ne supprimez pas les versions que vous n'utilisez plus, vous finirez par atteindre le [quota des versions d'application](#) et vous ne pourrez pas créer de nouvelles versions de cette application.

Vous pouvez éviter d'atteindre le quota en appliquant une stratégie de cycle de vie des versions d'application à vos applications. Une stratégie de cycle demande à Elastic Beanstalk de supprimer les versions d'application qui sont obsolètes, ou de supprimer des versions d'application lorsque le nombre total de versions pour une application dépasse un nombre spécifié.

Elastic Beanstalk applique la stratégie de cycle de vie d'une application chaque fois que vous créez une nouvelle version d'application, et supprime jusqu'à 100 versions chaque fois que la stratégie de cycle de vie est appliquée. Elastic Beanstalk supprime les anciennes versions après avoir créé la nouvelle version et ne comptabilise pas la nouvelle version dans le nombre de versions maximum défini dans la stratégie.

Elastic Beanstalk ne supprime pas les versions d'application en cours d'utilisation par un environnement, ou celles déployées dans des environnements qui ont été arrêtés moins de dix semaines avant le déclenchement de la stratégie.

Le quota de versions d'application s'applique à toutes les applications dans une région. Si vous avez plusieurs applications, configurez chacune d'entre elles avec une stratégie de cycle de vie appropriée pour éviter d'atteindre le quota. Par exemple, si vous avez 10 applications dans une région et que le quota est de 1 000 versions d'application, envisagez de définir une stratégie de cycle de vie avec un quota de 99 versions d'application pour toutes les applications, ou définissez d'autres valeurs dans chaque application tant que le total est inférieur à 1 000 versions d'application. Elastic Beanstalk applique uniquement la stratégie si la création de la version d'application réussit. Si vous avez déjà atteint le quota, vous devez donc supprimer certaines versions manuellement avant de créer une nouvelle version.

Par défaut, Elastic Beanstalk quitte le [bundle de fichiers source \(p. 415\)](#) de la version d'application dans Amazon S3 pour éviter la perte de données. Vous pouvez supprimer le bundle de fichiers source pour économiser de l'espace.

Vous pouvez définir les paramètres de cycle de vie via l'interface de ligne de commande Elastic Beanstalk et les API. Pour plus amples informations, veuillez consulter [eb appversion \(p. 1064\)](#), [CreateApplication](#) (à l'aide du paramètre `ResourceLifecycleConfig`) et [UpdateApplicationResourceLifecycle](#).

## Configuration des paramètres du cycle de vie d'une application dans la console

Vous pouvez spécifier les paramètres de cycle de vie dans la console Elastic Beanstalk.

Pour spécifier les paramètres de cycle de vie de votre application

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
4. Sélectionnez Paramètres.
5. Utilisez le formulaire à l'écran pour configurer les paramètres de cycle de vie de l'application.
6. Choisissez Enregistrer.

## Application version lifecycle settings

Configure a lifecycle policy to limit the number of application versions to retain for future deployments. This policy will not delete application versions that are currently deployed or are in the process of being created. [Learn more](#)

Lifecycle policy

Enable

Lifecycle rule

Set the application versions limit by total count  
200  Application Versions

Set the application versions limit by age  
180  days

Retention

Delete source bundle from S3

Service role

**Cancel** **Save**

La page des paramètres vous permet d'effectuer les actions suivantes.

- Configurer les paramètres de cycle de vie en fonction du nombre total de versions d'application ou de l'âge des versions d'application.
- Spécifier si le bundle de fichiers source doit être supprimé dans S3 lorsque la version de l'application est supprimée.
- Spécifier le rôle sous lequel la version de l'application est supprimée. Pour inclure toutes les autorisations requises pour la suppression de la version, choisissez le rôle de service Elastic Beanstalk par défaut, nommé `aws-elasticbeanstalk-service-role`, ou un autre rôle de service à l'aide des stratégies de service gérées Elastic Beanstalk. Pour de plus amples informations, veuillez consulter [Gestion des rôles de service Elastic Beanstalk \(p. 926\)](#).

## Balisage des versions d'application

Vous pouvez appliquer des balises à vos versions d'application AWS Elastic Beanstalk. Les balises sont des paires clé-valeur associées aux ressources AWS. Pour de plus amples informations sur le balisage

des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur de balise, et les types de ressources pris en charge, veuillez consulter [Balisage des ressources d'application Elastic Beanstalk \(p. 423\)](#).

Vous pouvez spécifier des balises lorsque vous créez une version d'application. Dans une version d'application, vous pouvez ajouter ou supprimer des balises, ainsi que mettre à jour les valeurs des balises existantes. Vous pouvez ajouter jusqu'à 50 balises par version de l'application.

## Ajout de balises lors de la création de la version de l'application

Lorsque vous utilisez la console Elastic Beanstalk pour [créer un environnement \(p. 442\)](#), et que vous choisissez de charger une version de votre code d'application, vous pouvez spécifier des clés et valeurs de balise à associer à la nouvelle version d'application.

Vous pouvez également utiliser la console Elastic Beanstalk pour [charger une version d'application \(p. 409\)](#) sans immédiatement l'utiliser dans un environnement. Vous pouvez spécifier des clés et des valeurs de balise lorsque vous chargez une version d'application.

Avec l'AWS CLI ou les autres clients basés sur l'API, ajoutez des balises en utilisant le paramètre `--tags` sur la commande `create-application-version`.

```
$ aws elasticbeanstalk create-application-version \
  --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
  --application-name my-app --version-label v1
```

Lorsque vous utilisez l'interface de ligne de commande EB pour créer ou mettre à jour un environnement, une version d'application est créée à partir du code que vous déployez. Il n'y a pas de moyen direct de baliser une version d'application lors de sa création via l'interface de ligne de commande EB. Consultez la section suivante pour en savoir plus sur l'ajout de balises à une version d'application existante.

## Gestion des balises de la version d'une application existante

Vous pouvez ajouter, mettre à jour et supprimer des balises dans une version d'application Elastic Beanstalk existante.

Pour gérer les balises d'une version d'application à l'aide de la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
4. Sélectionnez la version d'application que vous souhaitez gérer.
5. Choisissez Actions, puis Manage Tags (Gérer les balises).
6. Utilisez le formulaire à l'écran pour ajouter, mettre à jour ou supprimer des balises.
7. Choisissez Apply (Appliquer).

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour la version de votre application, utilisez [eb tags \(p. 1118\)](#) pour ajouter, mettre à jour, supprimer ou répertorier des balises.

Par exemple, la commande suivante répertorie les balises d'une version d'application.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

La commande suivante met à jour la balise mytag1 et supprime la balise mytag2.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \  
--resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Pour obtenir une liste complète des options et d'autres exemples, consultez [eb tags \(p. 1118\)](#).

Avec l'AWS CLI ou d'autres clients basés sur l'API, utilisez la commande [list-tags-for-resource](#) pour afficher les balises d'une version de l'application.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Utilisez la commande [update-tags-for-resource](#) pour ajouter, mettre à jour ou supprimer des balises dans une version d'application.

```
$ aws elasticbeanstalk update-tags-for-resource \  
--tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \  
--resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:applicationversion/my-app/my-version"
```

Spécifiez les balises à ajouter et les balises à mettre à jour dans le paramètre --tags-to-add de update-tags-for-resource. Une balise inexistante est ajoutée et la valeur d'une balise existante est mise à jour.

#### Note

Pour utiliser certaines commandes de l'interface de ligne de commande EB et AWS CLI avec une version d'application Elastic Beanstalk, vous avez besoin de l'ARN de la version d'application. Vous pouvez extraire l'ARN à l'aide de la commande suivante.

```
$ aws elasticbeanstalk describe-application-versions --application-name my-app --  
version-label my-version
```

## Création d'une solution groupée d'application

Lorsque vous utilisez la console AWS Elastic Beanstalk pour déployer une nouvelle application ou une version de l'application, vous avez besoin de télécharger un bundle source. Votre groupe source doit répondre aux critères suivants :

- Se compose d'un seul fichier ZIP ou fichier WAR (vous pouvez inclure plusieurs fichiers WAR à l'intérieur de votre fichier ZIP)
- Ne dépasse pas 512 Mo
- N'inclut pas un dossier parent ou un répertoire de niveau supérieur (sous-répertoires acceptés)

Si vous souhaitez déployer une application de travail qui traite des tâches en arrière-plan de façon périodique, votre groupe source d'application doit également inclure un fichier cron.yaml. Pour de plus amples informations, veuillez consulter [Tâches périodiques \(p. 525\)](#).

Si vous déployez votre application avec l'interface ligne de commande Elastic Beanstalk, AWS Toolkit for Eclipse ou AWS Toolkit for Visual Studio, le fichier ZIP ou WAR sera automatiquement structuré correctement. Pour de plus amples informations, veuillez consulter [Utilisation de l'interface de ligne de](#)

commande Elastic Beanstalk (EB) (p. 1027), Création et déploiement d'applications Java sur Elastic Beanstalk (p. 110) et . AWS Toolkit for Visual Studio (p. 224).

#### Sections

- [Création d'une solution groupée source à partir de la ligne de commande \(p. 416\)](#)
- [Création d'une solution groupée source avec Git \(p. 416\)](#)
- [Compression de fichiers dans le Finder de Mac OS X ou l'Explorateur Windows \(p. 416\)](#)
- [Création d'une solution groupée source pour une application .NET \(p. 421\)](#)
- [Test de votre solution groupée source \(p. 422\)](#)

## Création d'une solution groupée source à partir de la ligne de commande

Créer un bundle de fichiers source à l'aide de la commande `zip`. Pour inclure les fichiers et les dossiers cachés, utilisez un schéma semblable au suivant.

```
~/myapp$ zip ../myapp.zip -r *.[^.]*
adding: app.js (deflated 63%)
adding: index.js (deflated 44%)
adding: manual.js (deflated 64%)
adding: package.json (deflated 40%)
adding: restify.js (deflated 85%)
adding: .ebextensions/ (stored 0%)
adding: .ebextensions/xray.config (stored 0%)
```

Cela permet de garantir que les [fichiers de configuration \(p. 737\)](#) Elastic Beanstalk et d'autres fichiers et dossiers qui commencent par un point sont inclus dans l'archive.

Pour les applications web Tomcat, utilisez `jar` pour créer une archive web.

```
~/myapp$ jar -cvf myapp.war .
```

Les commandes ci-dessus incluent des fichiers masqués qui peuvent augmenter la taille du bundle de fichiers source inutilement. Pour plus de contrôle, utilisez un modèle de fichier plus détaillé ou [créez votre bundle source avec Git \(p. 416\)](#).

## Création d'une solution groupée source avec Git

Si vous utilisez Git pour gérer le code source de votre application, utilisez la commande `git archive` pour créer votre groupe source.

```
$ git archive -v -o myapp.zip --format=zip HEAD
```

`git archive` comprend uniquement les fichiers qui sont stockés dans Git, et exclut les fichiers ignorés et Git. Cela vous aide à conserver un bundle de fichiers source le plus petit possible. Pour plus d'informations, consultez la [page de manuel git-archive](#).

## Compression de fichiers dans le Finder de Mac OS X ou l'Explorateur Windows

Lorsque vous créez un fichier ZIP dans le Finder de Mac OS X ou l'Explorateur Windows, assurez-vous de compresser les fichiers et les sous-dossiers eux-mêmes, plutôt que de compresser le dossier parent.

### Note

L'interface utilisateur graphique (GUI) sur les systèmes d'exploitation basés sur Mac OS X et sur Linux n'affiche pas les fichiers et les dossiers dont les noms commencent par un point (.). Utilisez la ligne de commande au lieu de l'interface utilisateur graphique pour compresser votre application si le fichier ZIP doit inclure un dossier masqué, comme .ebextensions. Pour que les procédures de ligne de commande créent un fichier ZIP sur un système d'exploitation basé sur Linux ou sur Mac OS X, consultez [Création d'une solution groupée source à partir de la ligne de commande \(p. 416\)](#).

### Example

Supposons que vous avez un dossier de projet Python intitulé myapp qui inclut les fichiers et les sous-dossiers suivants :

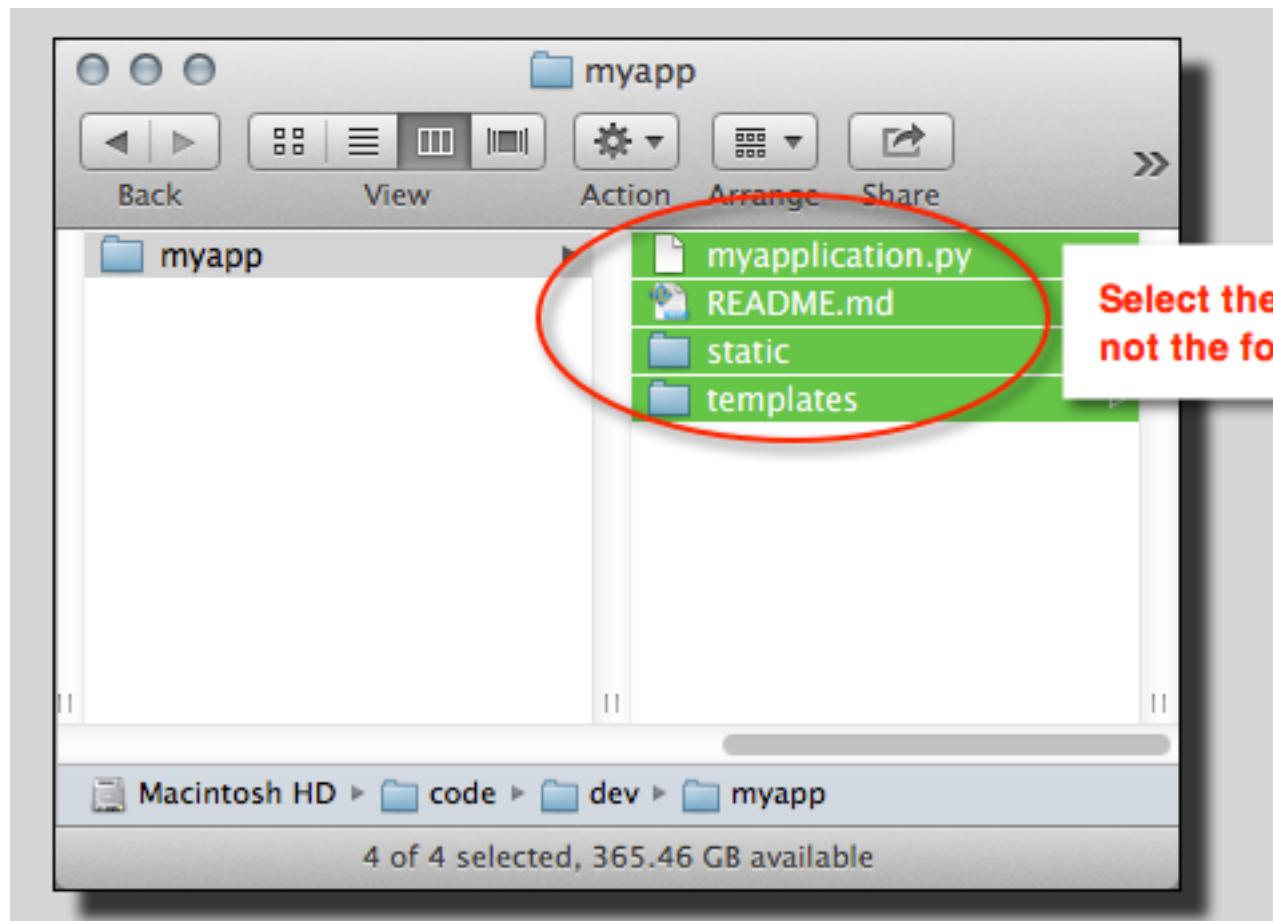
```
myapplication.py
README.md
static/
static/css
static/css/styles.css
static/img
static/img/favicon.ico
static/img/logo.png
templates/
templates/base.html
templates/index.html
```

Comme indiqué dans la liste des exigences ci-dessus, votre groupe source doit être compressé sans un dossier parent, afin que sa structure décompressée n'inclue pas un répertoire supplémentaire de niveau supérieur. Dans cet exemple, aucun dossier myapp ne doit être créé lorsque les fichiers sont décompressés (ou, dans la ligne de commande, aucun segment myapp ne doit être ajouté aux chemins d'accès aux fichiers).

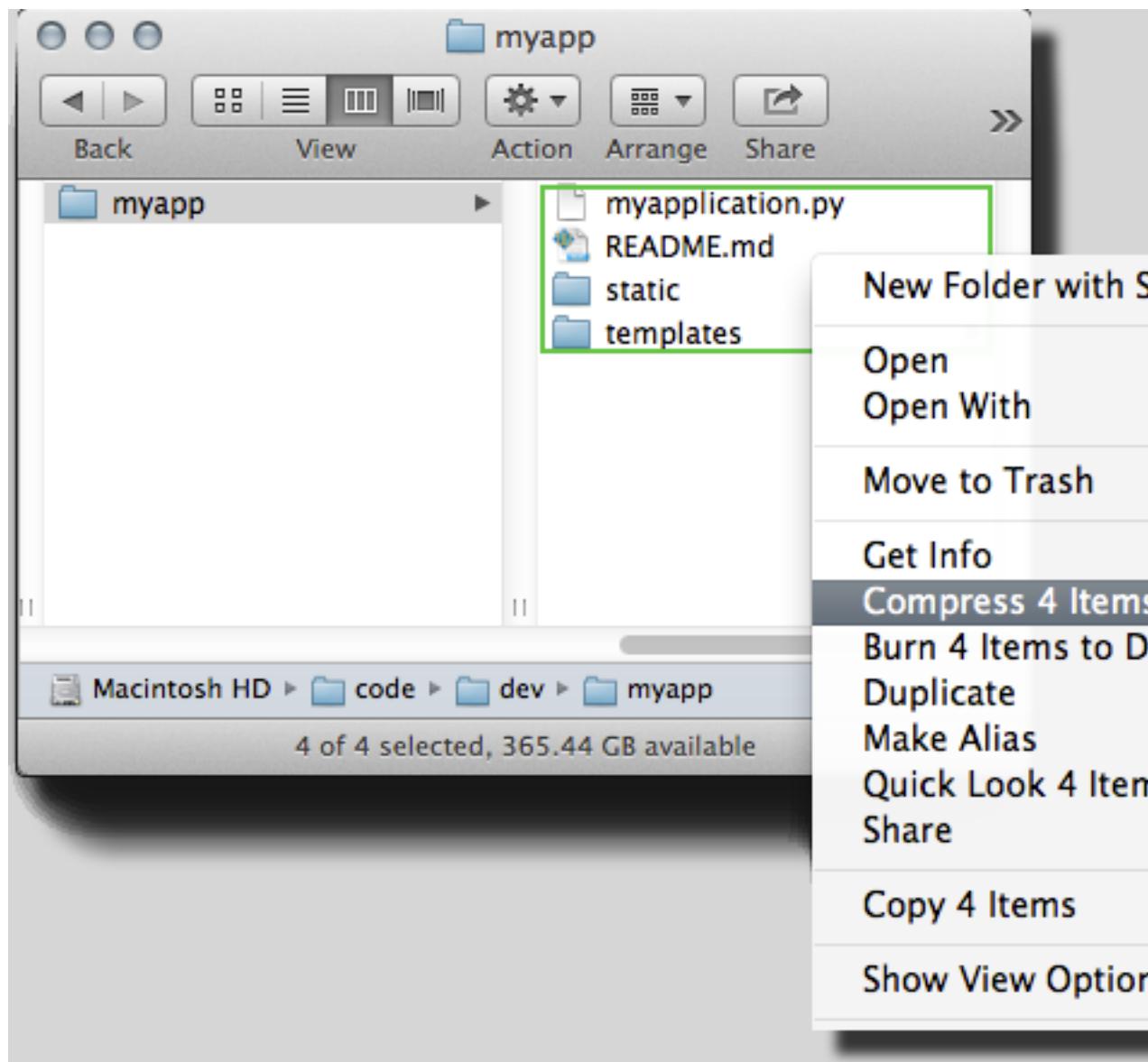
Cet exemple de structure de fichiers est utilisé dans cette rubrique pour illustrer comment compresser des fichiers.

Pour compresser des fichiers dans le Finder de Mac OS X

1. Ouvrez votre dossier de projet de niveau supérieur et sélectionnez tous les fichiers et sous-dossier qui s'y trouvent. Ne sélectionnez pas le dossier de niveau supérieur lui-même.

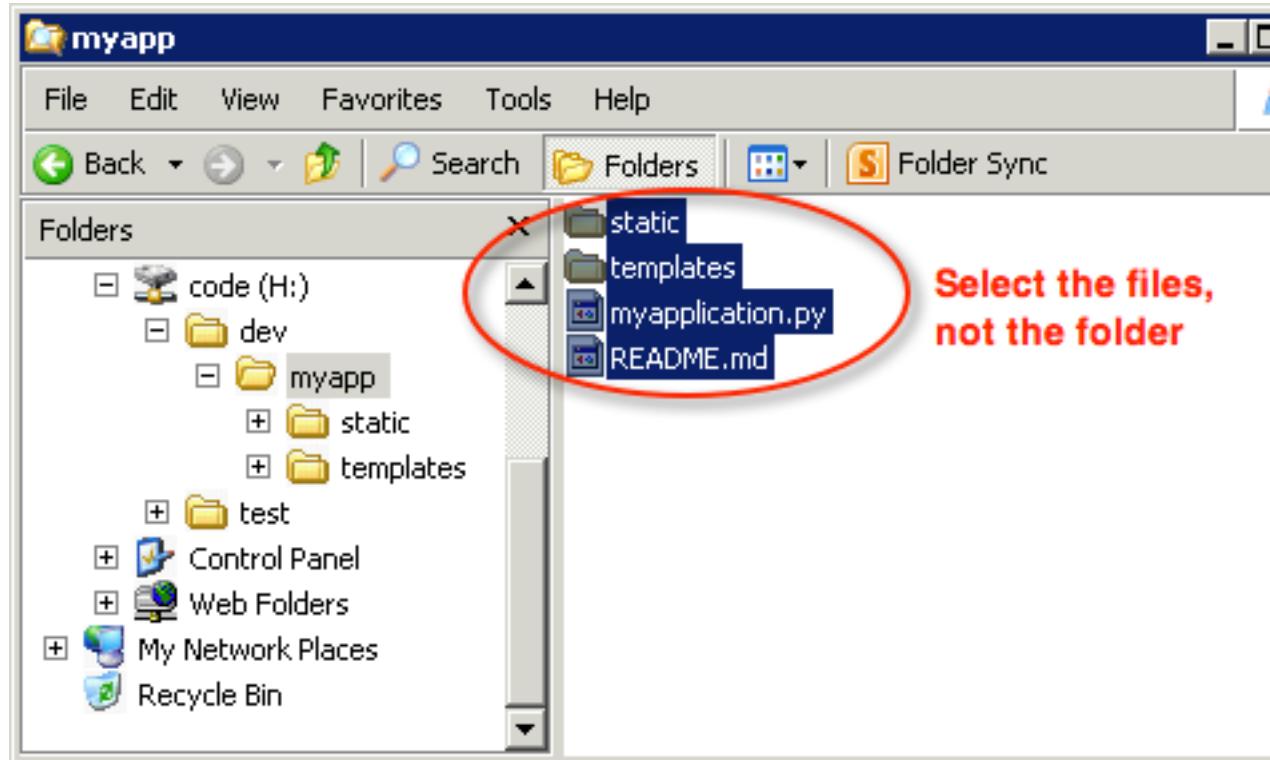


- Effectuez un clic droit sur les fichiers sélectionnés, puis choisissez Compresser X items, où X correspond au nombre de fichiers et de sous-dossiers que vous avez sélectionnés.

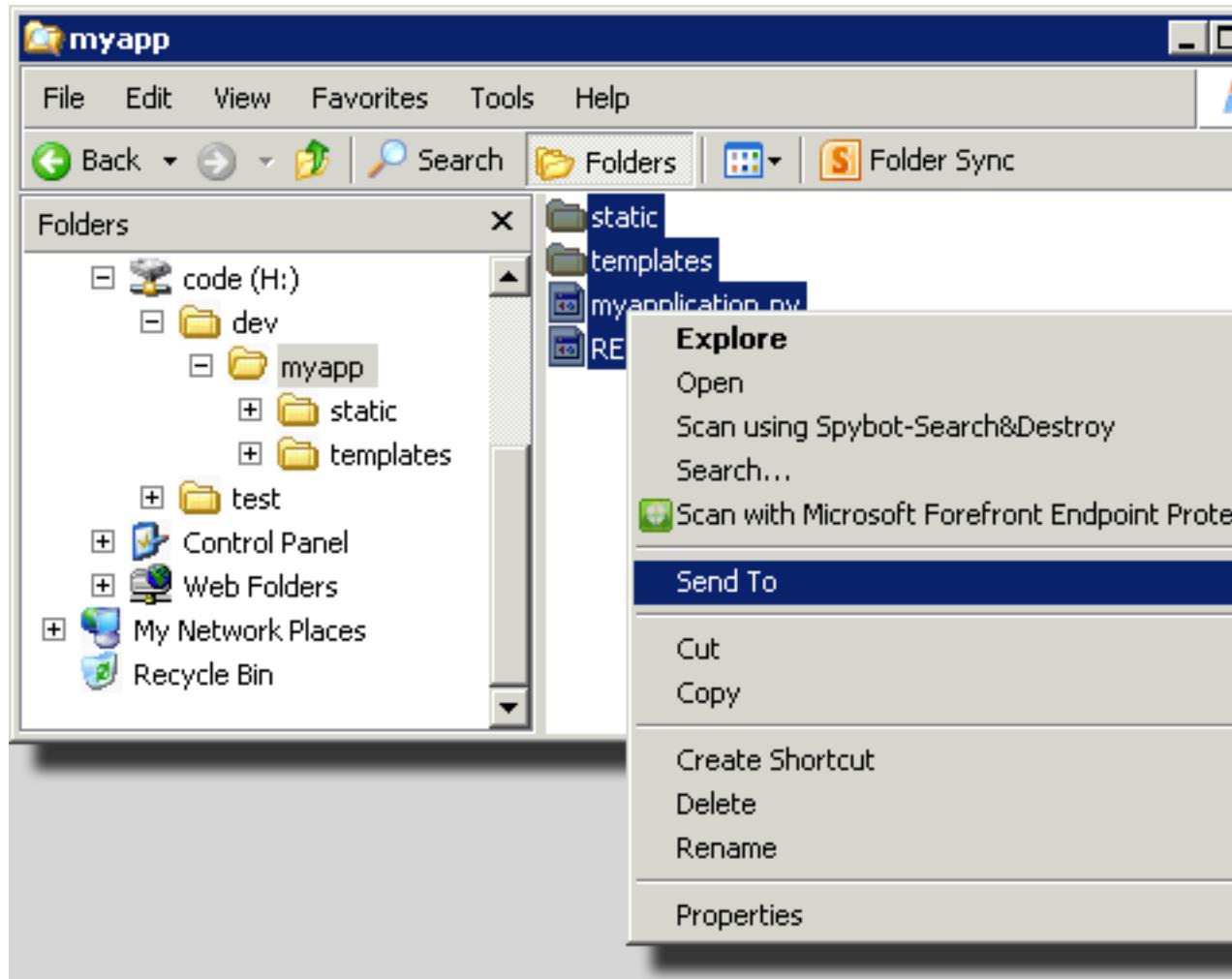


Pour compresser des fichiers dans l'Explorateur Windows

1. Ouvrez votre dossier de projet de niveau supérieur et sélectionnez tous les fichiers et sous-dossier qui s'y trouvent. Ne sélectionnez pas le dossier de niveau supérieur lui-même.



2. Effectuez un clic droit sur les fichiers sélectionnés, choisissez Envoyer vers, puis Compressed (zipped) folder (Dossier compressé).



## Création d'une solution groupée source pour une application .NET

Si vous utilisez Visual Studio, vous pouvez utiliser l'outil de déploiement inclus dans AWS Toolkit for Visual Studio pour déployer votre application .NET sur Elastic Beanstalk. Pour de plus amples informations, veuillez consulter [Déploiement d'applications Elastic Beanstalk dans .NET à l'aide de l'outil de déploiement \(p. 249\)](#).

Si vous avez besoin de créer manuellement un groupe source pour votre application .NET, vous ne pouvez pas créer simplement un fichier ZIP contenant le répertoire du projet. Vous devez créer un package de déploiement web pour votre projet convenant pour le déploiement sur Elastic Beanstalk. Il existe plusieurs méthodes que vous pouvez utiliser pour créer un package de déploiement :

- Créez le package de déploiement à l'aide de l'assistant Publier le site Web dans Visual Studio. Pour de plus amples informations, veuillez consulter [Comment : créer un Package de déploiement Web dans Visual Studio](#).

## Important

Lorsque vous créez le package de déploiement web, le nom du site doit commencer par Default Web Site.

- Si vous avez un projet .NET, vous pouvez créer le package de déploiement à l'aide de la commande msbuild, comme illustré dans l'exemple suivant.

## Important

Le paramètre DeployIisAppPath doit commencer par Default Web Site.

```
C:/> msbuild <web_app>.csproj /t:Package /p:DeployIisAppPath="Default Web Site"
```

- Si vous avez un projet de site web, vous pouvez utiliser l'outil Web Deploy IIS pour créer le package de déploiement. Pour plus d'informations, consultez [Packaging and Restoring a Web site](#).

## Important

Le paramètre apphostconfig doit commencer par Default Web Site.

Si vous déployez plusieurs applications ou une application ASP.NET Core, placez votre dossier .ebextensions à la racine de votre bundle de fichiers source, côté à côté avec les ensembles d'applications et le fichier manifeste :

```
~/workspace/source-bundle/
|-- .ebextensions
|   |-- environmentvariables.config
|   '-- healthcheckurl.config
|-- AspNetCore101HelloWorld.zip
|-- AspNetCoreHelloWorld.zip
|-- aws-windows-deployment-manifest.json
`-- VS2015AspNetWebApiApp.zip
```

## Test de votre solution groupée source

Vous pouvez souhaiter tester votre groupe source localement avant de le télécharger dans Elastic Beanstalk. Étant donné qu'Elastic Beanstalk utilise essentiellement la ligne de commande pour extraire les fichiers, il est préférable de réaliser vos tests à partir de la ligne de commande, plutôt qu'avec un outil GUI.

Pour tester l'extraction du fichier dans Mac OS X ou Linux

1. Ouvrez une fenêtre de terminal (Mac OS X) ou connectez-vous au serveur Linux. Accédez au répertoire qui contient votre groupe source.
2. A l'aide de la commande `unzip` ou `tar xf`, décompressez l'archive.
3. Assurez-vous que les fichiers décompressés s'affichent dans le même dossier que l'archive elle-même, plutôt que dans un nouveau répertoire ou dossier de niveau supérieur.

## Note

Si vous utilisez le Finder de Mac OS X pour décompresser l'archive, un dossier de niveau supérieur sera créé, quelle que soit la manière dont vous avez structuré l'archive elle-même. Pour obtenir de meilleurs résultats, utilisez la ligne de commande.

Pour tester l'extraction de fichier sous Windows

1. Téléchargez ou installez un programme qui vous permet d'extraire des fichiers compressés via la ligne de commande. Par exemple, vous pouvez télécharger le programme gratuit unzip.exe depuis <http://stahlforce.com/dev/index.php?tool=zipunzip>.
2. Si nécessaire, copiez le fichier exécutable dans le répertoire qui contient votre groupe source. Si vous avez installé un outil à l'échelle du système, vous pouvez ignorer cette étape.
3. A l'aide de la commande adéquate, décompressez l'archive. Si vous avez téléchargé unzip.exe à l'aide du lien à l'étape 1, la commande est `unzip <archive-name>`.
4. Assurez-vous que les fichiers décompressés s'affichent dans le même dossier que l'archive elle-même, plutôt que dans un nouveau répertoire ou dossier de niveau supérieur.

## Balisage des ressources d'application Elastic Beanstalk

Vous pouvez appliquer des balises aux ressources de vos applications AWS Elastic Beanstalk. Les balises sont des paires clé-valeur associées aux ressources AWS. Les balises peuvent vous aider à classer les ressources. C'est particulièrement utile si vous gérez un grand nombre de ressources dans le cadre de plusieurs applications AWS.

Voici plusieurs méthodes d'utilisation du balisage avec les ressources Elastic Beanstalk :

- Étapes du déploiement – Identifiez les ressources associées aux différentes étapes de votre application, comme développement, bêta et production.
- Allocation des coûts – Utilisez les rapports d'allocation des coûts pour suivre votre consommation des ressources AWS à l'aide de différents comptes de dépenses. Ces rapports incluent les ressources balisées et non balisées, et regroupent les coûts en fonction des balises. Pour savoir comment les rapports de répartition des coûts utilisent les balises, veuillez consulter [Utilisation des balises de répartition des coûts pour les rapports de facturation personnalisés](#) dans le Guide de l'utilisateur de AWS Billing and Cost Management.
- Contrôle d'accès – Utilisez les balises pour gérer les autorisations concernant les demandes et les ressources. Par exemple, un utilisateur qui ne peut créer et gérer que des environnements de test bêta ne doit avoir accès qu'aux ressources de l'étape bêta. Pour plus d'informations, consultez [Utilisation de balises pour contrôler l'accès aux ressources Elastic Beanstalk \(p. 978\)](#).

Vous pouvez ajouter jusqu'à 50 balises à chaque ressource. Les environnements sont légèrement différents : Elastic Beanstalk ajoute par défaut trois balises système aux environnements, et vous ne pouvez pas modifier ou supprimer ces balises. Outre les balises par défaut, vous pouvez ajouter jusqu'à 47 balises supplémentaires à chaque environnement.

Les contraintes suivantes s'appliquent aux clés et valeurs de balise :

- Les clés et les valeurs peuvent contenir uniquement des lettres, des chiffres, des espaces et les symboles suivants : \_ . : / = + - @
- Les clés peuvent contenir jusqu'à 127 caractères. Les valeurs peuvent contenir jusqu'à 255 caractères.

### Note

Ces limites s'appliquent aux caractères Unicode en UTF-8. Pour les autres encodages multioctets, la limite peut être inférieure.

- Les clés sont sensibles à la casse.
- Les clés ne peuvent pas commencer par `aws:` ou `elasticbeanstalk::`.

## Ressources que vous pouvez baliser

Vous trouverez ci-dessous les types de ressources Elastic Beanstalk que vous pouvez baliser, ainsi que des liens vers des rubriques relatives à la gestion des balises pour chacune d'entre elles :

- [Applications \(p. 424\)](#)
- [Environnements \(p. 629\)](#)
- [Versions de l'application \(p. 413\)](#)
- [Configurations enregistrées \(p. 783\)](#)
- [Versions de plateforme personnalisées \(p. 1157\)](#)

## Balisage des applications

Vous pouvez appliquer les balises à vos applications AWS Elastic Beanstalk. Les balises sont des paires clé-valeur associées aux ressources AWS. Pour de plus amples informations sur le balisage des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur de balise, et les types de ressources pris en charge, veuillez consulter [Balisage des ressources d'application Elastic Beanstalk \(p. 423\)](#).

Vous pouvez spécifier des balises lorsque vous créez une application. Dans une application existante, vous pouvez ajouter ou supprimer des balises, ainsi que mettre à jour les valeurs des balises existantes. Vous pouvez ajouter jusqu'à 50 balises à chaque application.

### Ajout de balises lors de la création de l'application

Lorsque vous utilisez la console Elastic Beanstalk pour [créer une application \(p. 406\)](#), vous pouvez spécifier des clés et valeurs de balise dans la boîte de dialogue Create New Application (Créer une nouvelle application).

Elastic Beanstalk

## Create new application

**Application information**

**Application Name**

Maximum length of 100 characters, not including forward slash (/).

**Description**

**Tags**

Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be a resource and is case-sensitive. [Learn more](#)

Key	Value	Remove
<input type="text"/>	<input type="text"/>	<input type="button" value="Remove"/>

**Add tag**

50 remaining

Ca

Si vous utilisez l'interface de ligne de commande EB pour créer une application, utilisez l'option `--tags` avec [eb init \(p. 1093\)](#) pour ajouter des balises.

```
~/workspace/my-app$ eb init --tags mytag1=value1,mytag2=value2
```

Avec l'AWS CLI ou les autres clients basés sur l'API, ajoutez des balises en utilisant le paramètre `--tags` sur la commande [create-application](#).

```
$ aws elasticbeanstalk create-application \
--tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
```

```
--application-name my-app --version-label v1
```

## Gestion des balises d'une application existante

Vous pouvez ajouter, mettre à jour et supprimer des balises dans une application Elastic Beanstalk existante.

Pour gérer les balises d'une application dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Choisissez Actions, puis Manage Tags (Gérer les balises).
4. Utilisez le formulaire à l'écran pour ajouter, mettre à jour ou supprimer des balises.
5. Choisissez Apply (Appliquer).

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour votre application, utilisez [eb tags \(p. 1118\)](#) pour ajouter, mettre à jour, supprimer ou répertorier des balises.

Par exemple, la commande suivante répertorie les balises dans une application.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

La commande suivante met à jour la balise mytag1 et supprime la balise mytag2.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
--resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Pour obtenir une liste complète des options et d'autres exemples, consultez [eb tags \(p. 1118\)](#).

Avec l'AWS CLI ou d'autres clients basés sur l'API, utilisez la commande [list-tags-for-resource](#) pour afficher les balises d'une application.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Utilisez la commande [update-tags-for-resource](#) pour ajouter, mettre à jour ou supprimer des balises dans une application.

```
$ aws elasticbeanstalk update-tags-for-resource \
--tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
--resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:application/my-app"
```

Spécifiez les balises à ajouter et les balises à mettre à jour dans le paramètre --tags-to-add de update-tags-for-resource. Une balise inexistante est ajoutée et la valeur d'une balise existante est mise à jour.

#### Note

Pour utiliser certaines commandes de l'interface de ligne de commande EB et AWS CLI avec une application Elastic Beanstalk, vous avez besoin de l'ARN de l'application. Vous pouvez extraire l'ARN à l'aide de la commande suivante.

```
$ aws elasticbeanstalk describe-applications --application-names my-app
```

# Gestion des environnements

AWS Elastic Beanstalk permet de créer facilement des environnements pour votre application. Vous pouvez créer et gérer des environnements distincts pour le développement, les tests et l'utilisation en production, et vous pouvez [déployer n'importe quelle version \(p. 476\)](#) de votre application dans n'importe quel environnement. Les environnements peuvent être de longue durée ou temporaires. Lorsque vous mettez un environnement hors service, vous pouvez sauvegarder sa configuration pour le recréer ultérieurement.

Au cours du développement de votre application, vous allez la déployer souvent, probablement dans plusieurs environnements différents pour répondre à différents besoins. Elastic Beanstalk vous permet de [configurer la façon dont les déploiements sont effectués \(p. 479\)](#). Vous pouvez effectuer le déploiement dans toutes les instances de votre environnement simultanément, ou fractionner le déploiement en lots via la propagation des déploiements.

Les [changements de configuration \(p. 488\)](#) sont traités séparément des déploiements et ont leur propre champ d'application. Par exemple, si vous modifiez le type des instances EC2 qui exécutent votre application, toutes les instances doivent être remplacées. En revanche, si vous modifiez la configuration de l'équilibrage de charge de l'environnement, ce changement peut être effectué sur place, sans interrompre le service ni réduire la capacité. Vous pouvez également appliquer des changements de configuration qui modifient les instances de votre environnement par lots, via les [mises à jour de configuration par propagation \(p. 489\)](#).

## Note

Modifiez les ressources de votre environnement en utilisant seulement Elastic Beanstalk. Si vous modifiez des ressources via la console d'un autre service, les commandes d'une interface de ligne de commande ou des kits SDK, Elastic Beanstalk ne pourra pas surveiller avec précision l'état de ces ressources, et vous ne pourrez pas enregistrer la configuration ni recréer l'environnement en toute sécurité. De plus, les modifications hors-bande peuvent provoquer des problèmes lors de la mise à jour ou de la mise hors service d'un environnement.

Lorsque vous lancez un environnement, vous sélectionnez une version de plateforme. Nous mettons régulièrement à jour les plateformes vers de nouvelles versions de plateforme dans le but d'améliorer les performances et de lancer de nouvelles fonctionnalités. Vous pouvez [mettre à jour votre environnement afin d'utiliser la dernière version de plateforme \(p. 496\)](#) à tout moment.

Lorsque votre application devient plus complexe, vous pouvez la diviser en plusieurs composants, chacun d'entre eux s'exécutant dans un environnement distinct. Pour les charges de travail longues, vous pouvez lancer des [environnements de travail \(p. 521\)](#) qui traitent des tâches à partir d'une file d'attente Amazon Simple Queue Service (Amazon SQS).

## Rubriques

- [Utilisation de la console de gestion de l'environnement Elastic Beanstalk \(p. 429\)](#)
- [Création d'un environnement Elastic Beanstalk \(p. 439\)](#)
- [Déploiement d'applications dans des environnements Elastic Beanstalk \(p. 476\)](#)
- [Configuration changes \(p. 488\)](#)
- [Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk \(p. 496\)](#)
- [Annulation de mises à jour de configuration d'environnement et de déploiements d'application \(p. 516\)](#)
- [Reconstruction d'environnements Elastic Beanstalk \(p. 517\)](#)

- Types d'environnement (p. 519)
- Environnements de travail Elastic Beanstalk (p. 521)
- Création de liens entre les environnements Elastic Beanstalk (p. 530)

## Utilisation de la console de gestion de l'environnement Elastic Beanstalk

La console Elastic Beanstalk fournit une page de gestion pour chacun de vos environnements AWS Elastic Beanstalk. Depuis une page de gestion, vous pouvez gérer la configuration de votre environnement et effectuer des actions courantes. Celles-ci incluent le redémarrage des serveurs Web exécutés dans votre environnement, le clonage de votre environnement et la reconstruction de votre environnement à partir de zéro.

The screenshot shows the AWS Elastic Beanstalk Management Console interface. On the left, a sidebar menu lists 'Environments', 'Applications', 'Change history', and sections for the current environment ('GettingStartedApp' and 'GettingStartedApp-env'). Under 'GettingStartedApp-env', options like 'Go to environment', 'Configuration', 'Logs', 'Health', 'Monitoring', 'Alarms', 'Managed updates', 'Events', and 'Tags' are visible. To the right, the main content area displays the environment details for 'GettingStartedApp-env'. It includes the application name 'GettingStartedApp', the URL 'GettingStartedApp-env.eba-nvfhyh93.us-west-2.elasticbeanstalk.com' with a copy link, and the status 'Ok' with a green checkmark icon. Below this is a 'Recent events' section showing a table of log entries:

Time	Type	Details
2020-10-15 21:24:42 UTC-0400	INFO	Successfully launched environment
2020-10-15 21:24:41 UTC-0400	INFO	Application available at
2020-10-15 21:24:41 UTC-0400	INFO	Added instance [i-0efc51]
2020-10-15 21:24:41 UTC-0400	INFO	Environment health has changed to
2020-10-15 21:24:10 UTC-0400	INFO	Instance deployment completed

## Accéder à la console de gestion de l'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

La page de présentation de l'environnement s'affiche. Le volet de navigation de la console affiche le nom de l'application à laquelle appartient l'environnement, avec les pages de gestion des applications associées et le nom de l'environnement, avec les pages de gestion de l'environnement.

### Rubriques

- [Présentation de l'environnement \(p. 430\)](#)
- [Actions dans l'environnement \(p. 432\)](#)
- [Configuration \(p. 433\)](#)
- [Logs \(p. 435\)](#)
- [Health \(p. 435\)](#)
- [Monitoring \(p. 436\)](#)
- [Alarms \(p. 436\)](#)
- [Mises à jour gérées \(p. 437\)](#)
- [Events \(p. 437\)](#)
- [Tags \(p. 438\)](#)

## Présentation de l'environnement

Pour afficher la page de présentation de l'environnement, choisissez le nom de l'environnement dans le volet de navigation, s'il s'agit de l'environnement actuel. Vous pouvez également accéder à l'environnement à partir de la page de l'application ou de la liste d'environnement principale de la page Environments (Environnements).

Le volet supérieur de la page de présentation de l'environnement affiche des informations de premier niveau sur votre environnement. Celles-ci incluent son nom, son URL, son état de santé actuel, le nom de la version de l'application actuellement déployée et la version de la plateforme sur laquelle l'application s'exécute. Vous pouvez voir les cinq événements d'environnement les plus récents sous le volet de présentation.

Sélectionnez Actualiser pour mettre à jour les informations affichées. La page de présentation inclut les options et informations suivantes.

### URL

L'URL de l'environnement se trouve en haut de la présentation et sous le nom de l'environnement. Il s'agit de l'URL de l'application web que l'environnement exécute.

### Health

État général de l'environnement. Si la fonction [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#) est activée, l'état de l'environnement s'affiche avec un bouton Causes que vous pouvez sélectionner pour consulter des informations supplémentaires sur l'état actuel de l'environnement.

Pour les environnements [Création de rapports d'intégrité de base \(p. 834\)](#), un lien vers la [Console de surveillance \(p. 830\)](#) s'affiche.

## Version en cours d'exécution

Nom de la version de l'application déployée et exécutée sur votre environnement. Sélectionnez Upload and deploy (Charger et déployer) pour charger un [bundle source \(p. 415\)](#) et le déployer dans votre environnement. Cette option permet de créer une nouvelle version d'application.

## Platform

Nom de la version de plateforme en cours d'exécution sur votre environnement. En général, elle comprend l'architecture, le système d'exploitation (OS), le langage et le serveur d'applications (collectivement, la branche de la plateforme), avec un numéro de version de plateforme spécifique. Choisissez Change (Changer) pour sélectionner une autre version de plateforme. Cette option s'affiche seulement si une autre version compatible de la branche de la plateforme est disponible.

La mise à jour de la version de la plateforme via cette option entraîne le remplacement des instances exécutées dans votre environnement par de nouvelles instances.

**Update platform version**

**Availability warning**

This operation replaces your instances; your application is unavailable during the update. To keep one instance in service during the update, enable rolling updates. Another option is to clone the environment, which creates a newer version of the platform, and then swap the CNAME of the clone with the original environment when you are ready to deploy the clone. Learn more at [Updating AWS Elastic Beanstalk Environments with Rolling Updates and Deploying Version with Zero Downtime](#).

Platform branch  
Tomcat 8.5 with Java 8 running on 64bit Amazon Linux

Current platform version  
3.3.1

New platform version  
3.3.2 (Recommended) ▾

Canc

### Note

Lorsque vous utilisez Elastic Beanstalk pour la première fois, seule la dernière version (recommandée) de chaque branche de plateforme est disponible pour son utilisation. Le Change (Changer) n'est pas disponible.

(Changement) devient d'abord disponible lorsqu'une nouvelle version de plateforme est publiée pour la branche. Une fois la mise à niveau vers la dernière version de branche de plateforme effectuée, vous avez la possibilité de revenir à une version précédente.

## Événements récents

La section Recent Events (Événements récents) de la plage de présentation de l'environnement inclut les événements les plus récents émis par votre environnement. Cette liste est actualisée en temps réel en même temps que votre environnement est mis à jour.

Sélectionnez Show all (Afficher tout) pour ouvrir la page Events (Événements).

## Actions dans l'environnement

La page de présentation de l'environnement contient un menu Actions que vous pouvez utiliser pour effectuer des opérations courantes sur votre environnement. Ce menu apparaît à droite de l'en-tête de l'environnement, en regard de l'option Create New Environment (Créer un environnement).

### Note

Certaines actions ne sont disponibles que dans certaines conditions et restent désactivées jusqu'à ce que ces conditions soient réunies.

## Charger la configuration

Chargez une configuration précédemment enregistrée. Les configurations sont enregistrées dans votre application et peuvent être chargées par tout environnement associé. Si vous avez modifié la configuration de votre environnement, vous pouvez charger une configuration enregistrée afin d'annuler ces modifications. Vous pouvez également charger une configuration que vous avez enregistrée à partir d'un autre environnement exécutant la même application afin de propager les modifications de configuration dans ces environnements.

## Enregistrer la configuration

Enregistrez la configuration actuelle de votre environnement dans votre application. Avant de modifier la configuration de votre environnement, enregistrez la configuration actuelle de façon à pouvoir la restaurer ultérieurement, si nécessaire. Vous pouvez également appliquer une configuration enregistrée lorsque vous lancez un nouvel environnement.

## Échanger l'URL d'un environnement

Echangez le CNAME de l'environnement actuel avec celui d'un nouvel environnement. Après un échange CNAME, l'ensemble du trafic vers l'application utilisant l'URL de l'environnement est acheminé vers le nouvel environnement. Lorsque vous êtes prêt à déployer une nouvelle version de votre application, vous pouvez lancer un environnement distinct dans le cadre de la nouvelle version. Lorsque le nouvel environnement est prêt à recevoir des demandes, effectuez un échange CNAME pour commencer à acheminer le trafic vers le nouvel environnement. Cette action n'interrompt pas vos services. Pour de plus amples informations, veuillez consulter [Déploiements bleu/vert avec Elastic Beanstalk \(p. 486\)](#).

## Cloner un environnement

Lancez un nouvel environnement avec la même configuration que celui en cours d'exécution.

## Clone with latest platform (Cloner avec la plateforme la plus récente)

Clonez votre environnement actuel avec la dernière version de la plateforme Elastic Beanstalk en cours d'utilisation. Cette option n'est disponible que lorsqu'une version plus récente de la plateforme de l'environnement actuel est disponible.

## Abort Current Operation (Annuler l'opération en cours)

Arrêtez une mise à jour d'environnement en cours. Suite à l'arrêt d'une opération, l'état de certaines instances de votre environnement peut être différent des autres, en fonction du niveau d'avancement de l'opération au moment où elle a été arrêtée. Cette option n'est disponible que lorsque votre environnement est en cours de mise à jour.

## Restart App Servers (Redémarrer les serveurs d'applications)

Redémarrez le serveur Web qui est en cours d'exécution sur les instances de votre environnement. Cette option n'entraîne pas la mise hors service ni le redémarrage des ressources AWS. Si votre environnement se comporte de façon étrange suite à l'envoi de demandes incorrectes, le redémarrage du serveur d'application peut restaurer temporairement la fonctionnalité, le temps d'identifier la cause première et de résoudre le problème.

## Rebuild Environment (Reconstruire l'environnement)

Suspendez toutes les ressources de l'environnement en cours d'exécution et créez un nouvel environnement avec les mêmes paramètres. Cette opération prend plusieurs minutes, soit à peu près le temps nécessaire au déploiement d'un nouvel environnement à partir de zéro. Lors d'une reconstruction, toutes les instances Amazon RDS qui sont exécutées dans la couche Données de votre environnement sont supprimées. Si vous avez besoin des données, créez un instantané. Vous pouvez créer un instantané manuellement [dans la console RDS](#) ou configurer la stratégie de suppression de votre couche Données afin de créer un instantané automatiquement avant de supprimer l'instance. Il s'agit du paramètre par défaut lorsque vous créez une couche Données.

## Terminate Environment (Supprimer l'environnement)

Résiliez toutes les ressources de l'environnement en cours d'exécution et supprimez l'environnement de l'application. Si une instance RDS est exécutée dans une couche Données et que vous devez conserver ses données, vérifiez que la stratégie de suppression de base de données est définie sur `Snapshot` ou sur `Retain`. Pour de plus amples informations, consultez [Cycle de vie de base de données \(p. 618\)](#) dans le chapitre Configuration des environnements de ce guide.

que vous preniez un instantané avant de résilier votre environnement. Vous pouvez créer un instantané manuellement [dans la console RDS](#) ou configurer la stratégie de suppression de votre couche Données afin de créer un instantané automatiquement avant de supprimer l'instance. Il s'agit du paramètre par défaut lorsque vous créez une couche Données.

## Restore environment (Restaurer l'environnement)

Si l'environnement a été suspendu au cours de l'heure précédente, vous pouvez le restaurer à partir de cette page. Après une heure, vous pouvez [le restaurer à partir de la page de présentation des applications \(p. 517\)](#).

## Configuration

La page Configuration overview (Vue d'ensemble de la configuration) affiche la configuration actuelle de votre environnement et de ses ressources, dont les instances Amazon EC2, un équilibriseur de charge,

les notifications et les paramètres de surveillance de l'état. Utilisez les paramètres figurant sur cette page pour personnaliser le comportement de votre environnement au cours des déploiements, activer des fonctionnalités supplémentaires, et modifier le type d'instance et d'autres paramètres que vous avez choisis lors de la création de l'environnement.

## Configuration overview

Search for an option name or value

<h3>Software</h3> <p>Environment properties: GRADLE_HOME, JAVA_HOME, M2, M2_HOME, PORT Log streaming: disabled Rotate logs: disabled X-Ray daemon: disabled</p> <p><a href="#">Edit</a></p>	<h3>Instances</h3> <p>EC2 security groups: awseb-e-m5yvre5nuj-stack-AWSEBSecurityGroup-12122MOSKFTC4 IMDSv1: disabled IOPS: container default Monitoring interval: 5 minute Root volume type: container default Size: container default Throughput: container default</p> <p><a href="#">Edit</a></p>	<h3>Load balancer</h3> <p>Listeners: 1 Load balancer type: application Processes: 1 Rules: 0 Shared: false Store logs: disabled</p> <p><a href="#">Edit</a></p>	<h3>Rolling updates and deployments</h3> <p>Batch size: 100% Command timeout: 600 Deployment policy: All at once Healthy threshold: Ok Ignore health check: disabled Rolling updates: disabled</p> <p><a href="#">Edit</a></p>
---	---	---	--

Pour de plus amples informations, veuillez consulter [Configuration d'environnements Elastic Beanstalk \(p. 532\)](#).

## Logs

La page Journaux vous permet d'extraire les journaux des instances EC2 de votre environnement. Lorsque vous demandez des journaux, Elastic Beanstalk envoie une commande aux instances, qui les chargent ensuite dans votre compartiment de stockage Elastic Beanstalk dans Amazon S3. Lorsque vous demandez des journaux sur cette page, Elastic Beanstalk les supprime automatiquement d'Amazon S3 après 15 minutes.

Vous pouvez également configurer les instances de votre environnement afin de charger les journaux dans Amazon S3 pour un stockage permanent une fois qu'ils ont fait l'objet d'une rotation au niveau local.

The screenshot shows the AWS Elastic Beanstalk interface for viewing logs. At the top, there's a breadcrumb navigation: Elastic Beanstalk > Environments > GettingStartedApp-env > Logs. Below this, a section titled "Logs" contains the instruction "Click Request Logs to retrieve the last 100 lines of logs or the entire set of logs from each EC2 instance." A "Learn more" link is provided. To the right, a vertical menu titled "Request Logs" offers two options: "Last 100 Lines" and "Full Logs", with "Last 100 Lines" currently selected and highlighted with a yellow box. At the bottom of the logs section, there's another call-to-action: "Click Request Logs to request and review log files for all your servers."

Pour de plus amples informations, veuillez consulter [Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk \(p. 884\)](#).

## Health

Si la surveillance améliorée de l'état a été activée, la page Enhanced health overview (Présentation améliorée de l'état) affiche des informations en direct sur l'état de chaque instance de votre environnement. Cette fonctionnalité permet à Elastic Beanstalk de surveiller attentivement les ressources de votre environnement, afin d'évaluer plus précisément l'état de votre application.

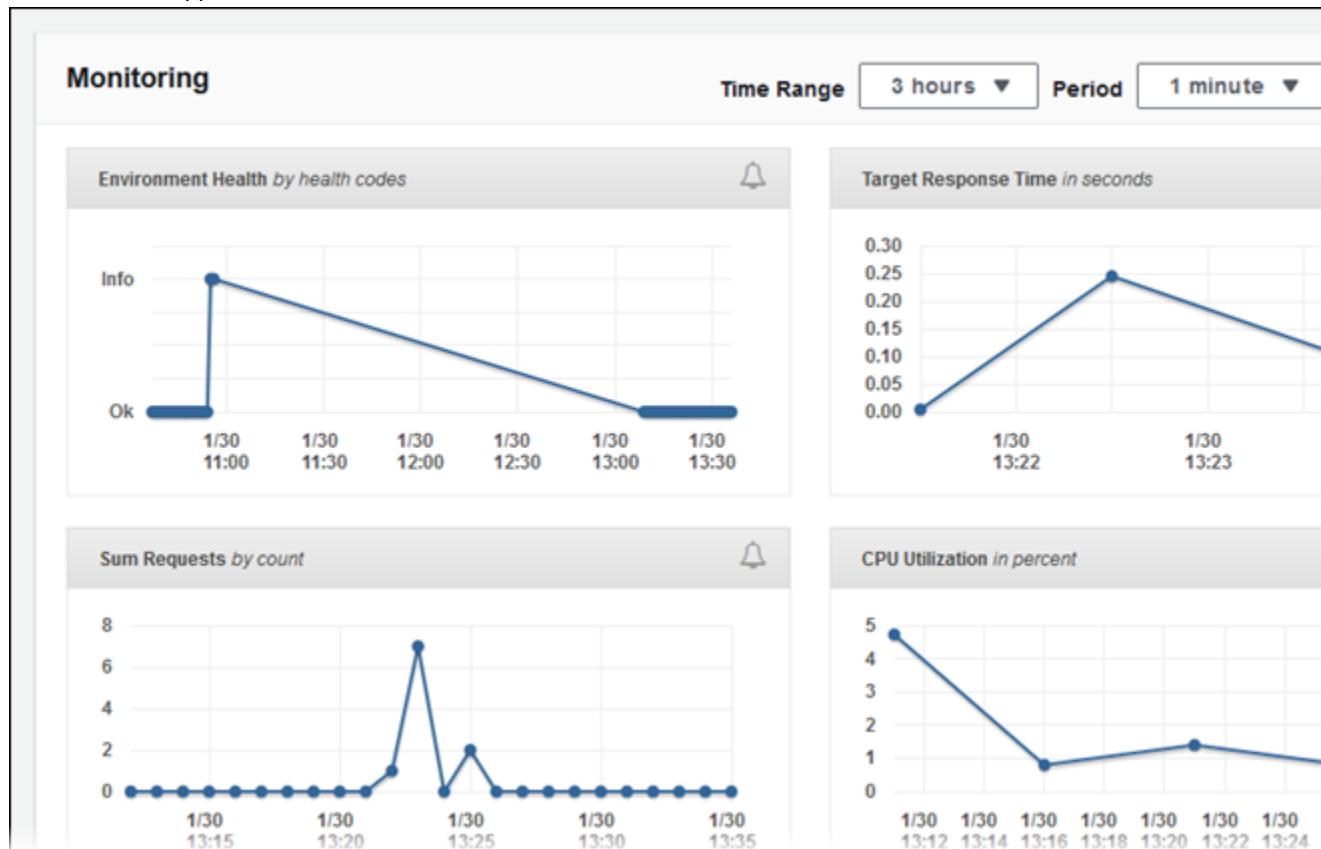
Lorsque la surveillance améliorée de l'état est activée, cette page affiche des informations sur les demandes envoyées par les instances de votre environnement et sur les métriques provenant du système d'exploitation, dont la latence, le chargement et l'utilisation de l'UC.

The screenshot shows the Enhanced health overview page. At the top, it displays "Instances: 2 Total, 2 Ok" and a "Learn more" link about enhanced health. Below this is a table with columns: Instance ID, Status, Running, Deployment ID, Requests/sec, 2xx Responses, 3xx Responses, 4xx Responses, 5xx Responses, P99 Latency, and P90 Latency. The table lists three rows: "Overall" (Status: Ok, Running: N/A, Requests/sec: 0.4, 2xx Responses: 100%, 3xx Responses: 0.0%, 4xx Responses: 0.0%, 5xx Responses: 0.0%, P99 Latency: 0.002, P90 Latency: 0.00), "i-00227807c4c4a1334" (Status: Ok, Running: 2 hours, Deployment ID: 3, Requests/sec: 0.2, 2xx Responses: 2, 3xx Responses: 0, 4xx Responses: 0, 5xx Responses: 0, P99 Latency: 0.002, P90 Latency: 0.00), and "i-03280193ba1ba4171" (Status: Ok, Running: 19 days, Deployment ID: 3, Requests/sec: 0.2, 2xx Responses: 2, 3xx Responses: 0, 4xx Responses: 0, 5xx Responses: 0, P99 Latency: 0.001, P90 Latency: 0.00).

Pour de plus amples informations, veuillez consulter [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#).

## Monitoring

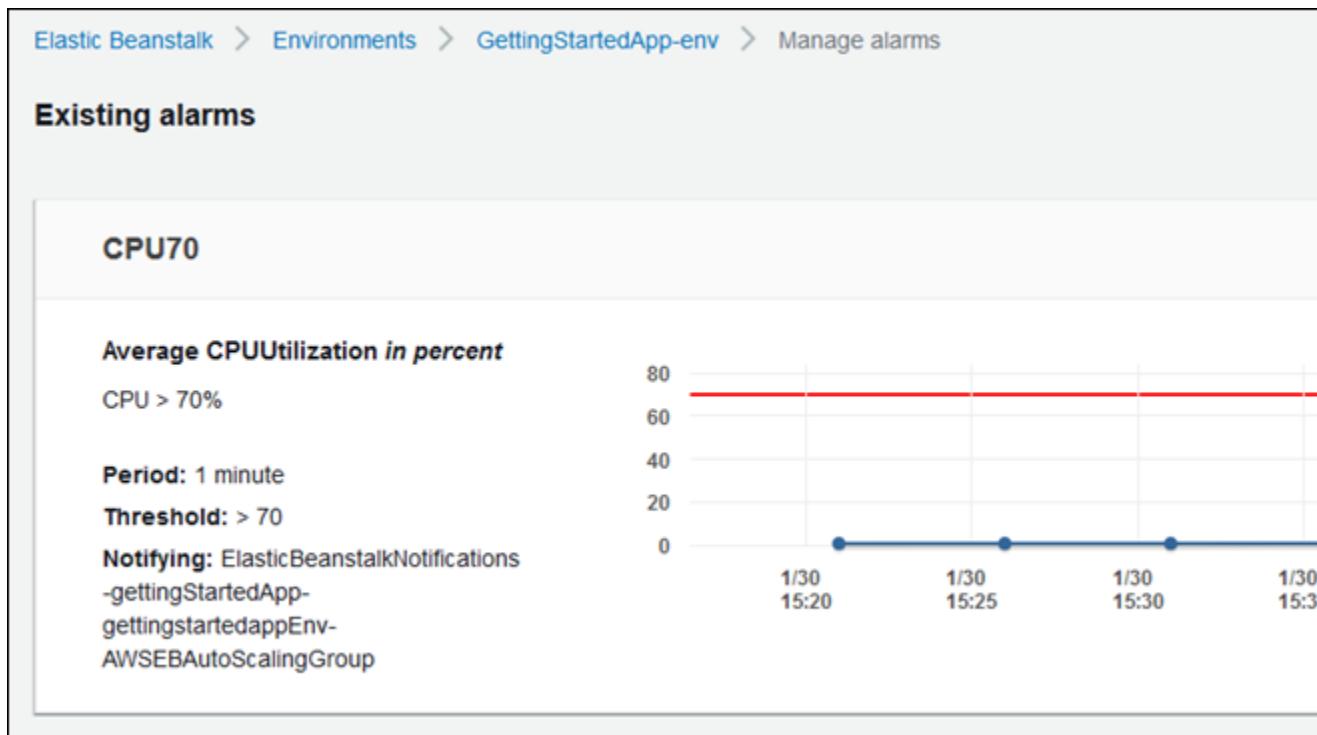
La page Surveillance présente l'ensemble des informations relatives à l'état de votre environnement. Cela inclut l'ensemble des métriques par défaut fournies par Elastic Load Balancing et Amazon EC2, et les graphiques qui montrent comment l'état de l'environnement a changé au fil du temps. Les options figurant sur cette page vous permettent de configurer d'autres graphiques relatifs à des métriques propres aux ressources et d'ajouter des alarmes pour toutes les métriques prises en charge par le système utilisé pour la création de rapports sur l'état.



Pour de plus amples informations, veuillez consulter [Surveillance de l'état de l'environnement dans la console de gestionAWS \(p. 830\)](#).

## Alarms

La page Existing alarms (Alarmes existantes) affiche des informations relatives aux alarmes que vous avez configurées pour votre environnement. Vous pouvez utiliser les options figurant sur cette page pour modifier ou supprimer des alarmes.



Pour de plus amples informations, veuillez consulter [Gestion des alarmes \(p. 874\)](#).

## Mises à jour gérées

La page Managed updates overview (Présentation des mises à jour gérées) inclut des informations sur les mises à jour gérées de la plateforme et les remplacements d'instance, en indiquant aussi bien les opérations terminées que celles à venir. Ces fonctionnalités vous permettent de configurer votre environnement afin de le mettre à jour automatiquement vers la dernière version de plateforme, au cours de la fenêtre de maintenance hebdomadaire de votre choix.

Entre les lancements des versions de plateforme, vous pouvez faire en sorte que votre environnement remplace la totalité de ses instances Amazon EC2 pendant la fenêtre de maintenance. Cela peut remédier aux problèmes qui se produisent lorsque votre application est exécutée pendant de longues périodes.

Pour de plus amples informations, veuillez consulter [Mises à jour gérées de la plateforme \(p. 501\)](#).

## Events

La page Events (Événements) affiche le flux d'événements relatif à votre environnement. Elastic Beanstalk génère des messages d'événement dès que vous interagissez avec l'environnement, et lorsque des ressources de votre environnement sont créées ou modifiées en conséquence de cette interaction.

Elastic Beanstalk > Environments > GettingStartedApp-env > Events

**Click the link to be routed to the previous Beanstalk Console**  
[Switch to the previous console](#)

## Events

Severity **TRACE** ▾

< 1 2 3 4 5 6 7 .

Time	Type	Details
2020-03-09 17:14:06 UTC-0700	INFO	createConfigurationTemplate completed successfully.
2020-03-09 17:14:06 UTC-0700	INFO	createConfigurationTemplate is starting.
2020-03-03 04:16:55 UTC-0800	INFO	Environment health has transitioned from Info to update completed 85 seconds ago and took 15
2020-03-03 04:16:07 UTC-0800	INFO	Environment update completed successfully.
2020-03-03 04:16:07 UTC-0800	INFO	Successfully deployed new configuration to envi

Pour de plus amples informations, veuillez consulter [Affichage du flux d'événements d'un environnement Elastic Beanstalk \(p. 879\)](#).

## Tags

La page Tags (Balises) affiche les balises appliquées par Elastic Beanstalk à l'environnement lors de sa création, ainsi que toutes les balises ajoutées. Vous pouvez ajouter, modifier et supprimer des balises personnalisées. Vous ne pouvez pas modifier ou supprimer les balises appliquées par Elastic Beanstalk.

Des balises d'environnement sont appliquées à toutes les ressources créées par Elastic Beanstalk pour prendre en charge votre application.

The screenshot shows the 'Tags for GettingStartedApp-env' page. At the top, there's a breadcrumb navigation: Elastic Beanstalk > Environments > GettingStartedApp-env > Tags. Below the title, a note says: 'Apply up to 47 tags in addition to the default tags to the resources in your environment. You can use tags to group and filter your environments by key-value pair. The key must be unique within the environment and is case-sensitive.' A 'Learn more' link is provided. The main area is a table with two columns: 'Key' and 'Value'. It contains three rows: one with 'elasticbeanstalk:environment-id' and 'e-cubmdjm6ga'; another with 'elasticbeanstalk:environment-name' and 'GettingStartedApp-env'; and a third with 'Name' and 'GettingStartedApp-env'. There are empty input fields for adding new tags. A 'Remove' button is visible on the right. At the bottom left, it says '47 remaining'.

Pour de plus amples informations, veuillez consulter [Balisage des ressources dans vos environnements Elastic Beanstalk \(p. 629\)](#).

## Création d'un environnement Elastic Beanstalk

Un AWS Elastic Beanstalk environnement est une collection de ressources AWS qui exécutent une version de l'application. Vous pouvez déployer plusieurs environnements lorsque vous avez besoin d'exécuter plusieurs versions d'une application. Par exemple, vous pouvez avoir des environnements de développement, d'intégration et de production.

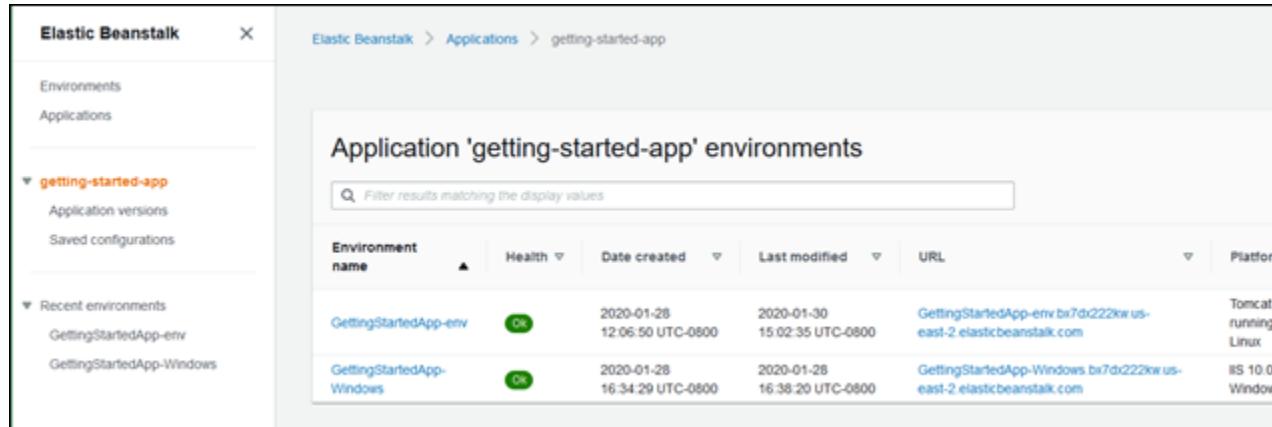
La procédure suivante lance un nouvel environnement exécutant l'application par défaut. Ces étapes sont simplifiées afin que votre environnement soit rapidement opérationnel, à l'aide de valeurs d'option par défaut. Pour obtenir des instructions détaillées décrivant les nombreuses options que vous pouvez utiliser pour configurer les ressources déployées par Elastic Beanstalk en votre nom, veuillez consulter [Assistant de création d'un environnement \(p. 442\)](#).

### Notes

- Pour de plus amples informations sur la création et la gestion d'environnements avec l'interface de ligne de commande (CLI) Elastic Beanstalk, veuillez consulter [Gestion des environnements Elastic Beanstalk avec l'interface de ligne de commande EB \(p. 1040\)](#).
- La création d'un environnement nécessite les autorisations de la stratégie gérée d'accès complet d'Elastic Beanstalk. Consultez [Stratégie utilisateur Elastic Beanstalk \(p. 24\)](#) pour plus de détails.

Pour lancer un environnement avec un exemple d'application (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le volet de navigation, choisissez Applications, puis le nom d'une application existante dans la liste ou [créez-en un \(p. 406\)](#).
3. Sur la page de présentation de l'application, choisissez Create a new environment (Créer un nouvel environnement).



The screenshot shows the AWS Elastic Beanstalk console interface. On the left, there's a sidebar with 'Environments' and 'Applications' sections. Under 'Applications', 'getting-started-app' is expanded, showing 'Application versions' and 'Saved configurations'. Below that is a 'Recent environments' section with 'GettingStartedApp-env' and 'GettingStartedApp-Windows'. The main area is titled 'Application 'getting-started-app' environments'. It contains a table with two rows:

Environment name	Health	Date created	Last modified	URL	Platform
GettingStartedApp-env	<span>OK</span>	2020-01-28 12:06:50 UTC-0800	2020-01-30 15:02:35 UTC-0800	GettingStartedApp-env.bn7dx222kw.us-east-2.elasticbeanstalk.com	Tomcat running Linux
GettingStartedApp-Windows	<span>OK</span>	2020-01-28 16:34:29 UTC-0800	2020-01-28 16:38:20 UTC-0800	GettingStartedApp-Windows.bn7dx222kw.us-east-2.elasticbeanstalk.com	IIS 10.0 Windows

4. Ensuite, pour le niveau d'environnement, choisissez l'environnement de serveur web ou le [niveau d'environnement de travail \(p. 14\)](#). Vous ne pouvez pas modifier le niveau d'un environnement après sa création.

#### Note

La plateforme .NET sur Windows Server (p. 192) ne prend pas en charge le niveau d'environnement worker.

The screenshot shows the 'Select environment tier' step in the AWS Elastic Beanstalk console. It has two options: 'Web server environment' (selected) and 'Worker environment'. A note below explains the difference: 'AWS Elastic Beanstalk has two types of environment tiers to support different types of web applications. Web server environments run a website, web application, or web API that serves HTTP requests. Worker environments run a background processing task that listens for messages on an Amazon SQS queue. Worker applications post the results of their processing back to the application by using HTTP.' There are 'Learn more' links for each.

5. Pour Platform (Plateforme), sélectionnez la plateforme et la branche de plateforme qui correspondent au langage utilisé par votre application.

Note

Elastic Beanstalk prend en charge plusieurs [versions \(p. 31\)](#) pour la plupart des plateformes répertoriées. Par défaut, la console sélectionne la version recommandée pour la plateforme et la branche de plateforme que vous choisissez. Si votre application nécessite une version différente, vous pouvez la sélectionner ici, ou choisir Configurer plus d'options, selon les instructions de l'étape 7. Pour plus d'informations sur les versions de plateforme prises en charge, consultez [the section called "Plateformes prises en charge" \(p. 31\)](#).

6. Pour l'option Code de l'application, choisissez Exemple d'application.
7. Pour personnaliser davantage votre environnement, choisissez Configurer plus d'options. Les options suivantes peuvent être définies uniquement lors de la création de l'environnement :
  - Nom de l'environnement
  - Nom de domaine
  - Version de plateforme
  - VPC
  - Palier

Vous pouvez modifier les paramètres suivants après la création de l'environnement, mais ils requièrent la mise en œuvre de nouvelles instances ou d'autres ressources et leur application peut prendre du temps :

- Type d'instance, volume racine, paire de clés et rôle AWS Identity and Access Management (IAM)
- Base de données interne Amazon RDS
- Equilibreur de charge

Pour de plus amples informations sur tous les paramètres disponibles, veuillez consulter [Assistant de création d'un environnement \(p. 442\)](#).

8. Choisissez Create environment.

Alors qu'Elastic Beanstalk crée votre environnement, vous êtes redirigé vers la [console Elastic Beanstalk \(p. 429\)](#). Une fois que l'intégrité de l'environnement devient verte, choisissez l'URL à côté du nom de l'environnement pour afficher l'application en cours d'exécution. Cette URL est généralement accessible depuis Internet, sauf si vous configurez votre environnement afin d'utiliser un [VPC personnalisé avec un équilibrage de charge interne \(p. 457\)](#).

#### Rubriques

- [Assistant de création d'un environnement \(p. 442\)](#)
- [Clonage d'un environnement Elastic Beanstalk \(p. 461\)](#)
- [Arrêt d'un environnement Elastic Beanstalk \(p. 464\)](#)
- [Création d'environnements Elastic Beanstalk avec l'interface de ligne de commande \(CLI\) AWS \(p. 465\)](#)
- [Création d'environnements Elastic Beanstalk avec l'API \(p. 466\)](#)
- [Construction d'une URL Launch Now \(p. 469\)](#)
- [Création et mise à jour de groupes d'environnements Elastic Beanstalk \(p. 474\)](#)

## Assistant de création d'un environnement

Dans [Création d'un environnement Elastic Beanstalk \(p. 439\)](#), nous présentons la façon d'ouvrir l'assistant Create a new environment (Créer un nouvel environnement) et de créer rapidement un environnement. Choisissez Créer un environnement pour démarrer un environnement avec un nom par défaut, un domaine généré automatiquement, un exemple de code d'application et des paramètres recommandés.

Cette rubrique décrit l'assistant Create a new environment (Créer un nouvel environnement) ainsi que l'ensemble des méthodes auxquelles vous pouvez avoir recours pour configurer l'environnement que vous souhaitez créer.

## Page principale de l'Assistant

La page principale de l'assistant Crée un nouvel environnement commence par présenter les informations de nommage du nouvel environnement. Définissez le nom et le sous-domaine de l'environnement et créez une description pour l'environnement. N'oubliez pas que les paramètres de cet environnement ne peuvent plus être modifiés une fois l'environnement créé.

### Environment information

Choose the name, subdomain, and description for your environment. These cannot be changed later.

**Application name**  
getting-started-app

**Environment name**  
GettingStartedApp-env-1

**Domain**  
Leave blank for autogenerated value .us-east-2.elasticbeanstalk  
**Check availability**

**Description**

- Name (Nom) : entrez le nom de l'environnement. Le formulaire fournit un nom généré.
- Domain (Domaine) : (environnements de serveur web) entrez un nom de domaine unique pour votre environnement. Le nom par défaut est le nom de l'environnement. Vous pouvez entrer un nom de domaine différent. Elastic Beanstalk utilise ce nom pour créer un CNAME unique pour l'environnement. Pour vérifier si le nom de domaine souhaité est disponible, choisissez Vérifier la disponibilité.
- Description : entrez la description de cet environnement.

## Sélection d'une plateforme pour le nouvel environnement

Vous pouvez créer un nouvel environnement à partir de deux types de plateformes :

- Plateforme gérée
- Plateforme personnalisée

### Plateforme gérée

Dans la plupart des cas, vous utilisez une plateforme gérée Elastic Beanstalk pour votre nouvel environnement. Lorsque l'assistant de création d'un environnement démarre, il sélectionne par défaut l'option Managed platform (Plateforme gérée).

**Platform**

Managed platform  
Platforms published and maintained by AWS Elastic Beanstalk. [Learn more](#)

Custom platform  
Platforms created and owned by you.

Platform

Tomcat

Platform branch

Tomcat 8.5 with Java 8 running on 64bit Amazon Linux

Platform version

3.3.2 (Recommended)

Sélectionnez une plateforme, une branche de plateforme au sein de cette plateforme et une version de plateforme spécifique dans la branche. Lorsque vous sélectionnez une branche de plateforme, la version recommandée dans la branche est sélectionnée par défaut. En outre, vous pouvez sélectionner n'importe quelle version de plateforme que vous avez utilisée auparavant.

#### Note

Pour un environnement de production, nous vous recommandons de choisir une version de plateforme dans une branche de plateforme prise en charge. Pour plus d'informations sur les états de branche de plateforme, consultez la définition d'une branche de plate-forme dans le [the section called “Glossaire des plateformes” \(p. 25\)](#).

#### Plateforme personnalisée

Si une plateforme prête à l'emploi ne répond pas à vos besoins, vous pouvez créer un nouvel environnement à partir d'une plateforme personnalisée. Pour spécifier une plateforme personnalisée, choisissez l'option Custom platform (Plateforme personnalisée), puis sélectionnez l'une des plateformes personnalisées disponibles. Si aucune plateforme personnalisée n'est disponible, cette option est grisée.

## Fourniture du code d'application

Maintenant que vous avez sélectionné la plateforme à utiliser, l'étape suivante consiste à fournir votre code d'application.

**Application code**

- Sample application**  
Get started right away with sample code.
- Existing version**  
Application versions that you have uploaded for getting-started-app.
- Upload your code**  
Upload a source bundle from your computer or copy one from Amazon S3.

Vous avez plusieurs options:

- Vous pouvez utiliser l'exemple d'application fourni par Elastic Beanstalk pour chaque plateforme.
- Vous pouvez utiliser le code que vous avez déjà déployé dans Elastic Beanstalk. Choisissez Version existante et votre application dans la section Code de l'application.
- Vous pouvez charger un nouveau code. Sélectionnez Charger votre code, puis Charger. Vous pouvez charger un nouveau code d'application à partir d'un fichier local, ou spécifier l'URL du compartiment Amazon S3 contenant le code de votre application.

#### Note

En fonction de la version de plateforme que vous avez sélectionnée, vous pouvez charger votre application dans un [bundle source \(p. 415\)](#) au format ZIP, un [fichier WAR \(p. 118\)](#) ou une [configuration Docker en texte brut \(p. 48\)](#). La taille du fichier ne doit pas dépasser 512 Mo.

Lorsque vous choisissez de charger un nouveau code, vous pouvez également fournir les balises à associer à votre nouveau code. Pour plus d'informations sur le balisage d'une version d'application, consultez [the section called "Balisage des versions d'application" \(p. 413\)](#).

## Application code

- Sample application  
Get started right away with sample code.
- Existing version  
Application versions that you have uploaded for getting-started-app.
- Upload your code  
Upload a source bundle from your computer or copy one from Amazon S3.

### ▼ Source code origin

(Maximum size 512 MB)

Local file

Public S3 URL

File name : **java-tomcat-v3.zip**

 File successfully uploaded

**Version label**  
Unique name for this version of your application code.

getting-started-app-source

### ▼ Application code tags

Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key within the resource and is case-sensitive. [Learn more](#)

Key	Value	Remove
<input type="text"/>	<input type="text"/>	<input type="button" value="Remove"/>
<input type="button" value="Add tag"/>		

50 remaining

Pour créer rapidement un environnement à l'aide des options de configuration par défaut, vous pouvez maintenant choisir Create environment (Créer un environnement). Choisissez Configurer plus d'options pour apporter d'autres modifications de configuration, comme décrit dans les sections suivantes.

## Page de configuration de l'Assistant

Lorsque vous choisissez Configure more options (Configurer d'autres options), l'Assistant affiche la page Configure (Configurer). Sur cette page, vous pouvez sélectionner une configuration prédéfinie, modifier la version de la plateforme que vous souhaitez utiliser pour votre environnement ou faire des choix de configuration spécifiques pour le nouvel environnement.

### Choisir une configuration prédéfinie

Dans la section Presets (Préréglages) de la page, Elastic Beanstalk fournit plusieurs préréglages de configuration pour différents cas d'utilisation. Chaque préréglage inclut les valeurs recommandées pour plusieurs [options de configuration \(p. 658\)](#).

The screenshot shows the AWS Elastic Beanstalk Configuration page for an environment named 'GettingStartedApp-env-1'. At the top, there is a breadcrumb navigation: 'Elastic Beanstalk > Applications > getting-started-app'. Below the breadcrumb, the environment name is displayed. Under the heading 'Presets', it says 'Start from a preset that matches your use case or choose *Custom configuration* to unset recommended values and use the settings defined in your configuration files.' A list of configuration presets follows:

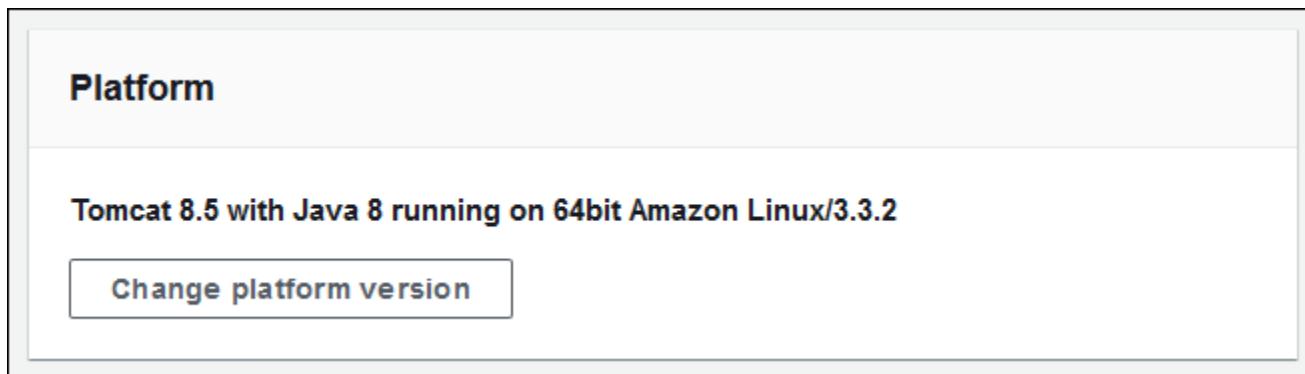
- Single instance (*Free Tier eligible*)
- Single instance (using Spot instance)
- High availability
- High availability (using Spot and On-Demand instances)
- Custom configuration

Les préréglages haute disponibilité comprennent un équilibrEUR de charge et sont recommandés pour les environnements de production. Choisissez-les si vous souhaitez un environnement capable d'exécuter plusieurs instances pour une haute disponibilité et d'évoluer en fonction de la charge. Les préréglages pour instance unique sont principalement recommandés pour le développement. Deux des paramètres prédéfinis permettent d'activer les demandes d'instance Spot. Pour de plus amples informations sur la configuration de capacité Elastic Beanstalk, veuillez consulter [Groupe Auto Scaling \(p. 547\)](#).

Le dernier préréglage, Custom configuration (Configuration personnalisée) supprime toutes les valeurs recommandées à l'exception des paramètres de rôle et utilise les valeurs par défaut de l'API. Choisissez cette option si vous déployez un bundle source avec des [fichiers de configuration \(p. 737\)](#) qui définissent les options de configuration. L'option Configuration personnalisée est également sélectionnée automatiquement si vous modifiez les préréglages de configuration Faible coût ou Haute disponibilité.

### Changer la version de la plateforme

Dans la section Platform (Plateforme) de la page, vous pouvez modifier la version de la plateforme que votre nouvel environnement utilisera. Vous pouvez choisir la version recommandée dans n'importe quelle branche de plateforme ou n'importe quelle version de plateforme que vous avez utilisée dans le passé.



## Personnalisation de votre configuration

En plus (ou au lieu) de choisir un préréglage de configuration, vous pouvez ajuster les [options de configuration \(p. 658\)](#) de votre environnement. L'Assistant Configure (Configurer) affiche plusieurs catégories de configuration. Chaque catégorie de configuration affiche un résumé des valeurs pour un groupe de paramètres de configuration. Choisissez Edit (Modifier) pour modifier ce groupe de paramètres.

### Catégories de configuration

- [Paramètres du logiciel \(p. 448\)](#)
- [Instances \(p. 449\)](#)
- [Capacity \(p. 450\)](#)
- [Équilibrer de charge \(p. 450\)](#)
- [Propagation des mises à jour et déploiements \(p. 452\)](#)
- [Security \(p. 453\)](#)
- [Monitoring \(p. 454\)](#)
- [Mises à jour gérées \(p. 455\)](#)
- [Notifications \(p. 456\)](#)
- [Network \(p. 457\)](#)
- [Database \(p. 458\)](#)
- [Tags \(p. 459\)](#)
- [Environnement de travail \(p. 460\)](#)

### Paramètres du logiciel

Utilisez la page [Modify software configuration \(Modifier la configuration du logiciel\)](#) pour configurer le logiciel sur les instances Amazon Elastic Compute Cloud (Amazon EC2) exécutant votre application. Vous pouvez configurer les propriétés de l'environnement, le débogage AWS X-Ray, le stockage et la diffusion des journaux d'instance et les paramètres spécifiques à la plateforme. Pour plus d'informations, consultez [the section called “Paramètres du logiciel” \(p. 632\)](#).

Elastic Beanstalk > Applications > getting-started-app

## Modify software

The following settings control platform behavior and let you pass key-value pairs in as OS environment variables. [Learn more](#)

### Platform options

Target .NET runtime

4.0

Enable 32-bit applications

False

### AWS X-Ray

X-Ray daemon

### Instances

Utilisez la page Modify instances (Modifier les instances) pour configurer les instances Amazon EC2 exécutant votre application. Pour plus d'informations, consultez [the section called “Instances Amazon EC2” \(p. 538\)](#).

Elastic Beanstalk > Applications > getting-started-app

## Modify instances

### Amazon CloudWatch monitoring

The time interval between when metrics are reported from the EC2 instances.

Monitoring interval

5 minute

### Root volume (boot device)

Root volume type

(Container default)

## Capacity

Utilisez la page de configuration **Modify capacity** (Modifier la capacité) pour configurer la capacité de calcul de votre environnement et les paramètres de groupe AUTO sCALING pour optimiser le nombre et le type d'instances que vous utilisez. Vous pouvez également modifier la capacité de votre environnement en fonction des déclencheurs ou d'une planification.

Un environnement à charge équilibrée peut exécuter plusieurs instances pour une haute disponibilité et éviter les temps d'arrêt lors de mises à jour de configuration et de déploiements. Dans un environnement à charge équilibrée, le nom de domaine est mappé à l'équilibrEUR de charge. Dans un environnement d'instance unique, il est mappé à une adresse IP Elastic sur l'instance.

### Warning

Un environnement d'instance unique n'est pas prêt pour la production. Si l'instance devient instable lors du déploiement, ou si Elastic Beanstalk résilie l'instance lors d'une mise à jour de configuration, votre application peut être indisponible pendant un certain temps. Utilisez des environnements d'instance unique pour le développement ou les tests, ou comme environnement intermédiaire. Utilisez des environnements à charge équilibrée pour la production.

Pour plus d'informations sur les paramètres de capacité de l'environnement, consultez [the section called "Groupe Auto Scaling" \(p. 547\)](#).

Elastic Beanstalk > Applications > getting-started-app

## Modify capacity

Configure the compute capacity of your environment and Auto Scaling settings to optimize the number of instances used.

### Auto Scaling Group

#### Environment type

Load balanced

#### Instances

Min

1



Max

2



#### Fleet composition

Choose a mix of On-Demand and Spot Instances with multiple instance types. Spot Instances are automatically launched at the lowest available price.

On-Demand instances

Combine purchase options and instances

#### Maximum spot price

The maximum price per instance-hour, in USD, that you're willing to pay for a Spot Instance. Setting a custom price limits your chances to fulfill your requests.

## ÉquilibrEUR de charge

Utilisez la page de configuration **Modifier l'équilibrEUR de charge** pour sélectionner un type d'équilibrEUR de charge et configurer les paramètres correspondants. Dans un environnement à charge équilibrée,

L'équilibrer de charge de votre environnement correspond au point d'entrée pour tout le trafic qui se dirige vers votre application. Elastic Beanstalk prend en charge plusieurs types d'équilibrer de charge. Par défaut, la console Elastic Beanstalk crée un équilibrer Application Load Balancer et le configure pour servir le trafic HTTP sur le port 80.

#### Note

Vous ne pouvez uniquement sélectionner le type d'équilibrage de charge de votre environnement lors de la création de l'environnement.

Pour plus d'informations sur les types et paramètres d'équilibrer de charge, consultez [the section called "Équilibrer de charge" \(p. 562\)](#) et [the section called "HTTPS" \(p. 792\)](#).

Elastic Beanstalk > Applications > getting-started-app

## Modify load balancer

### Application Load Balancer

Application layer load balancer—routing HTTP and HTTPS traffic based on protocol, port, and route to environment processes.



### Classic Load Balancer

*Previous generation — HTTP, HTTPS, and*

### Network Load Balancer

Ultra-high performance and static IP addresses for your application.



## Application Load Balancer

You can specify listeners for your load balancer. Each listener routes incoming client traffic on a specified port using a specific protocol to one or more environment processes. By default, we've configured your load balancer with a standard web server on port 80.

Action

<input type="checkbox"/>	Port	Protocol	SSL certificate
<input type="checkbox"/>	80	HTTP	--

## Processes

For each environment process, you can specify the protocol and port that the load balancer uses to route requests to the process.

## Propagation des mises à jour et déploiements

Utilisez la page Modify rolling updates and deployments (Modifier les mises à jour et les déploiements évolutifs) pour configurer le traitement des déploiements d'applications et des mises à jour de configuration par Elastic Beanstalk pour votre environnement.

Les déploiements d'applications se produisent lorsque vous téléchargez un bundle source d'application mis à jour et le déployez dans votre environnement. Pour plus d'informations sur la configuration des déploiements, consultez [the section called "Options de déploiement" \(p. 479\)](#).

Elastic Beanstalk > Environments > GettingStartedApp-env > Configuration

## Modify rolling updates and deployments

### Application deployments

Choose how AWS Elastic Beanstalk propagates source code changes and software configuration updates. [Learn more](#)

Deployment policy

All at once

Batch size:

Percentage

Fixed

100 % of instances at a time

Traffic split

10 % to new application version

Traffic splitting evaluation time

5 minutes

Les changements de configuration qui modifient la [configuration du lancement \(p. 681\)](#) ou les [paramètres de VPC \(p. 696\)](#) nécessitent de résilier toutes les instances de votre environnement et de les remplacer. Pour plus d'informations sur la définition du type de mise à jour et d'autres options, consultez [the section called "Configuration changes" \(p. 488\)](#).

**Configuration updates**

Changes to virtual machine settings and VPC configuration trigger rolling updates to replace the instances in your environment.

[Learn more](#)

**Rolling update type**

▼

**Batch size**

▲

The maximum number of instances to replace in each phase of the update.

**Minimum capacity**

▲

The minimum number of instances to keep in service at all times.

**Pause time**

Pause the update for up to an hour between each batch.

## Security

Utilisez la page Modify security (Modifier la sécurité) pour configurer les paramètres de sécurité du service et de l'instance.

Pour obtenir une description des concepts de sécurité Elastic Beanstalk, veuillez consulter [Autorisations \(p. 21\)](#). Pour plus d'informations sur la configuration des paramètres de sécurité de l'environnement, consultez [the section called "Sécurité" \(p. 626\)](#).

## Modify security

### Service role

Service role

aws-elasticbeanstalk-service-role



### Virtual machine permissions

EC2 key pair

-- Choose a key pair --



IAM instance profile

aws-elasticbeanstalk-ec2-role



Cancel

### Monitoring

Utilisez la page Modify monitoring (Modifier la surveillance) pour configurer les rapports d'intégrité, les règles de surveillance et la diffusion d'événements d'intégrité. Pour plus d'informations, consultez the section called “Activation des rapports d'intégrité améliorée” (p. 845), the section called “Règles d'intégrité améliorée” (p. 859) et the section called “Diffusion des informations d'intégrité de l'environnement” (p. 903).

## Modify monitoring

### Health reporting

Enhanced health reporting provides free real-time application and operating system monitoring of the instances and resources in your environment. The `EnvironmentHealth` custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see [Amazon CloudWatch Pricing](#).

#### System

- Enhanced
- Basic

#### CloudWatch Custom Metrics - Instance

[Choose metrics](#)

### Mises à jour gérées

Utilisez la page Modify managed updates (Modifier les mises à jour gérées) pour configurer les mises à jour de plateformes gérées. Vous pouvez décider si vous voulez les activer, définir la planification et configurer d'autres propriétés. Pour plus d'informations, consultez [the section called "Mises à jour gérées" \(p. 501\)](#).

## Modify managed updates

### Managed platform updates

Enable managed platform updates to apply platform updates automatically during a weekly maintenance window that your application stays available during the update process.

#### Managed updates

Enabled

#### Weekly update window

Tuesday at 12 : 00 UTC

Any available managed updates will run between Tuesday, 4:00 AM and Tuesday, 6:00 AM (-0800 GMT).

#### Update level

Minor and patch

#### Instance replacement

If enabled, an instance replacement will be scheduled if no other updates are available.

Enabled

### Notifications

Utilisez la page [Modify notifications](#) (Modifier les notifications) pour spécifier une adresse e-mail où recevoir des [notifications par e-mail](#) (p. 644) pour les événements importants de votre environnement.

Elastic Beanstalk > Applications > getting-started-app

## Modify notifications

### Email notifications

Enter an email address to receive email notifications for important events from your environment. [Learn more](#)

Email

`user@example.com`

[Cancel](#)

[Save](#)

### Network

Si vous avez créé un [VPC personnalisé \(p. 649\)](#), la page de configuration Modify network (Modifier le réseau) pour configurer votre environnement pour l'utiliser. Si vous ne choisissez pas de VPC, Elastic Beanstalk utilise le VPC et les sous-réseaux par défaut.

## Modify network

### Virtual private cloud (VPC)

#### VPC

Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console.

vpc-0f9c96ae77f3c49c1 (172.31.0.0/16) | private-public



[Create custom VPC](#)

### Load balancer settings

Assign your load balancer to a subnet in each Availability Zone (AZ) in which your application runs. For a publicly accessible application, choose Public and choose public subnets.

#### Visibility

Make your load balancer internal if your application serves requests only from connected VPCs. Public load balancers serve requests from the Internet.

Public



### Load balancer subnets

### Database

Utilisez la page Modify database configuration (Modifier la configuration de la base de données) pour ajouter une base de données Amazon RDS (Amazon Relational Database Service) à votre environnement à des fins de développement et de test. Elastic Beanstalk fournit les informations de connexion aux instances en définissant les propriétés de l'environnement pour le nom d'hôte de la base de données, le nom d'utilisateur, le mot de passe, le nom de la table et le port.

Pour plus d'informations, consultez [the section called “Base de données” \(p. 617\)](#).

## Modify database

Add an Amazon RDS SQL database to your environment for development and testing. AWS Elastic Beanstalk provides information to your instances by setting environment properties for the database hostname, username, password, and port. When you add a database to your environment, its lifecycle is tied to your environment's. For production environments, you can configure your instances to connect to a database. [Learn more](#)

### Restore a snapshot

Restore an existing snapshot in your account, or create a new database.

Snapshot

None



### Database settings

Choose an engine and instance type for your environment's database.

Engine

mysql



Engine version

### Tags

Utilisez la page de configuration Modify tags (Modifier les balises) pour ajouter des [balises](#) aux ressources de votre environnement. Pour plus d'informations sur l'utilisation de balises d'environnement, consultez [Balisage des ressources dans vos environnements Elastic Beanstalk \(p. 629\)](#).

Elastic Beanstalk > Applications > getting-started-app

## Modify tags

Apply up to 50 tags to the resources in your environment in addition to the default tags.

Key	Value
mytag1	value1

Add tag      Remove tag

49 remaining

### Environnement de travail

Si vous créez un environnement au niveau du travail, utilisez la page **Modify worker** (Modifier l'environnement de travail) pour configurer l'environnement de travail. Le démon de l'environnement de travail sur les instances de votre environnement extrait les éléments d'une file d'attente Amazon Simple Queue Service (Amazon SQS) et les relaie sous forme de messages de publication vers votre application de travail. Vous pouvez choisir la file d'attente Amazon SQS à partir de laquelle le démon de l'environnement de travail lit (générée automatiquement ou existante). Vous pouvez également configurer les messages que le démon de l'environnement de travail envoie à votre application.

Pour plus d'informations, consultez [the section called "Environnements de travail" \(p. 521\)](#).

## Modify worker

You can create a new Amazon SQS queue for your worker application or pull work items from an existing queue. The instances in your environment pulls an item from the queue and relays it in the body of a POST request to a local host.

The screenshot shows the 'Modify worker' configuration page in the AWS Elastic Beanstalk console. In the 'Queue' section, 'Worker queue' is selected in the dropdown, and 'Autogenerated queue' is chosen from the options. A note below says 'SQS queue from which to read work items.' In the 'Messages' section, there is a field labeled 'HTTP path'.

## Clonage d'un environnement Elastic Beanstalk

Vous pouvez utiliser un environnement Elastic Beanstalk existant comme base d'un nouvel environnement en clonant l'environnement existant. Par exemple, vous pouvez créer un clone de façon à utiliser une version plus récente de la branche de plateforme utilisée par la plateforme de l'environnement d'origine. Elastic Beanstalk configure le clone avec les mêmes paramètres d'environnement que ceux utilisés par l'environnement d'origine. Lorsque vous clonez un environnement existant au lieu de créer un nouvel environnement, vous n'avez pas besoin de configurer manuellement les paramètres d'option, les variables d'environnement et les autres paramètres. Elastic Beanstalk crée également une copie de toutes les ressources AWS associées à l'environnement d'origine. Toutefois, pendant le processus de clonage, Elastic Beanstalk ne copie pas les données depuis Amazon RDS vers le clone. Une fois que vous avez créé l'environnement cloné, vous pouvez modifier les paramètres de configuration de l'environnement selon vos besoins.

Vous pouvez uniquement cloner un environnement vers une version de plateforme différente de la même branche de plateforme. Une autre branche de plateforme ne sera pas forcément compatible. Pour utiliser une autre branche de plateforme, vous devez créer manuellement un nouvel environnement, déployer votre code d'application et apporter les modifications nécessaires au code et aux options pour garantir que votre application fonctionne correctement sur la nouvelle branche de plateforme.

### Note

Elastic Beanstalk n'inclut aucune modification non gérée des ressources dans le clone. Les modifications que vous apportez aux ressources AWS à l'aide d'outils autres que la console de gestion Elastic Beanstalk, les outils de ligne de commande ou l'API sont considérées comme des modifications non gérées.

## AWSConsole de gestion

Pour cloner un environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans la page de vue d'ensemble de l'environnement, choisissez Environment actions (Actions, d'environnement), puis effectuez l'une des opérations suivantes :
  - Choisissez Cloner un environnement pour cloner l'environnement sans apporter aucune modification à la version de la plateforme.
  - Choisissez Cloner avec la dernière plateforme pour cloner l'environnement avec une version plus récente de la branche de plateforme de l'environnement d'origine.
4. Sur la page Clone Environment (cloner l'environnement), examinez les informations de la section Original Environment (Environnement original), afin de vérifier si vous avez choisi l'environnement à partir duquel vous souhaitez créer un clone.
5. Dans la section New Environment (Nouvel environnement), vous pouvez modifier les valeurs des champs Environment name (Nom de l'environnement), Environment URL (URL de l'environnement), Description, Platform version (Version de la plateforme) et Service role (Fonction du service), qui sont définies automatiquement par Elastic Beanstalk en fonction de l'environnement d'origine.

### Note

Si la version de plateforme utilisée dans l'environnement d'origine n'est pas celle recommandée pour l'utiliser dans la branche de plateforme, vous êtes averti qu'une autre version de plateforme est recommandée. Choisissez Platform version (Version de la plateforme) pour afficher la version de la plateforme recommandée dans la liste : par exemple, 3.3.2 (Recommended).

Elastic Beanstalk > Environments > GettingStartedApp-env > Clone environment

## Clone environment

You can launch a new environment based on an existing environment's configuration settings while optionally choosing a different version for the new environment. [Learn more](#)

### Original environment

Environment name

GettingStartedApp-env

Environment URL

GettingStartedApp-env.gap8pzvmti.us-east-2.elasticbeanstalk.com

Platform

Tomcat 8.5 with Java 8 running on 64bit Amazon Linux/3.3.1

### New environment

Environment name

GettingStartedApp-env-1

Environment URL

gettingstartedapp-env-1

.us-east-2.elasticbeanstal

[Check availability](#)

Description

Clone of GettingStartedApp-env

Platform branch

Tomcat 8.5 with Java 8 running on 64bit Amazon Linux

Platform version

3.3.1



Warning

A different platform version is recommended.

463

6. Une fois que vous êtes prêt, choisissez Cloner.

## Interface de ligne de commande (CLI) Elastic Beanstalk

Utilisez la commande eb clone pour cloner un environnement en cours d'exécution, comme suit :

```
~/workspace/my-app$ eb clone my-env1
Enter name for Environment Clone
(default is my-env1-clone): my-env2
Enter DNS CNAME prefix
(default is my-env1-clone): my-env2
```

Vous pouvez spécifier le nom de l'environnement source dans la commande de clonage ou l'omettre afin de cloner l'environnement par défaut pour le dossier de projet actuel. L'interface de ligne de commande (CLI) EB vous invite à saisir un nom et un préfixe DNS pour le nouvel environnement.

Par défaut, eb clone crée le nouvel environnement avec la dernière version disponible de la plateforme de l'environnement source. Pour forcer l'interface de ligne de commande (CLI) EB à utiliser la même version, même si une version plus récente est disponible, utilisez l'option --exact.

```
~/workspace/my-app$ eb clone --exact
```

Pour plus d'informations sur cette commande, consultez [eb clone \(p. 1068\)](#).

## Arrêt d'un environnement Elastic Beanstalk

Vous pouvez résilier un environnement AWS Elastic Beanstalk en cours d'exécution à l'aide de la console Elastic Beanstalk. De ce fait, vous évitez d'engager des frais pour les ressources AWS inutilisées.

### Note

Vous pouvez toujours lancer un nouvel environnement en utilisant la même version ultérieurement. Si vous possédez des données d'un environnement que vous souhaitez conserver, définissez la stratégie de suppression de base de données sur Retain avant de résilier l'environnement. Cela permet de maintenir la base de données opérationnelle en dehors d'Elastic Beanstalk. Tous les environnements Elastic Beanstalk doivent ensuite s'y connecter en tant que base de données externe. Si vous souhaitez sauvegarder les données sans maintenir la base de données opérationnelle, définissez la stratégie de suppression pour qu'elle prenne un instantané de la base de données avant de résilier l'environnement. Pour de plus amples informations, consultez [Cycle de vie de base de données \(p. 618\)](#) dans le chapitre Configuration des environnements de ce guide.

Elastic Beanstalk peut ne pas réussir à arrêter votre environnement. Une raison courante est que le groupe de sécurité d'un autre environnement comporte une dépendance sur le groupe de sécurité de l'environnement que vous souhaitez résilier. Pour plus d'informations sur la manière d'éviter ce problème, consultez [Groupes de sécurité \(p. 542\)](#) sur la page Instances EC2 de ce guide.

## Console Elastic Beanstalk

Pour résilier un environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

### Note

Lorsque vous résiliez votre environnement, le CNAME associé à l'environnement résilié devient disponible pour que tout le monde puisse l'utiliser.

Quelques minutes sont nécessaires à Elastic Beanstalk pour résilier les ressources AWS en cours d'exécution dans l'environnement.

## AWS CLI

Pour résilier un environnement

- Exécutez la commande suivante.

```
$ aws elasticbeanstalk terminate-environment --environment-name my-env
```

## API

Pour résilier un environnement

- Appelez `TerminateEnvironment` avec le paramètre suivant :

```
EnvironmentName = SampleAppEnv
```

```
https://elasticbeanstalk.us-west-2.amazonaws.com/?EnvironmentName=SampleAppEnv
&Operation=TerminateEnvironment
&AuthParams
```

## Création d'environnements Elastic Beanstalk avec l'interface de ligne de commande (CLI) AWS

1. Vérifiez si le CNAME pour l'environnement est disponible.

```
$ aws elasticbeanstalk check-dns-availability --cname-prefix my-cname
{
    "Available": true,
    "FullyQualifiedCNAME": "my-cname.elasticbeanstalk.com"
}
```

2. Assurez-vous que la version de votre application existe.

```
$ aws elasticbeanstalk describe-application-versions --application-name my-app --
version-label v1
```

Si vous ne disposez pas d'une version de l'application pour votre source, créez-la. Par exemple, la commande suivante crée une version d'application à partir d'un bundle de fichiers source dans Amazon Simple Storage Service (Amazon S3).

```
$ aws elasticbeanstalk create-application-version --application-name my-app --version-label v1 --source-bundle S3Bucket=DOC-EXAMPLE-BUCKET,S3Key=my-source-bundle.zip
```

- Créez un modèle de configuration pour l'application.

```
$ aws elasticbeanstalk create-configuration-template --application-name my-app --template-name v1 --solution-stack-name "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.2 (Passenger Standalone)"
```

- Créez un environnement.

```
$ aws elasticbeanstalk create-environment --cname-prefix my-cname --application-name my-app --template-name v1 --version-label v1 --environment-name v1clone --option-settings file://options.txt
```

Les paramètres d'option sont définis dans le fichier options.txt :

```
[  
 {  
     "Namespace": "aws:autoscaling:launchconfiguration",  
     "OptionName": "IamInstanceProfile",  
     "Value": "aws-elasticbeanstalk-ec2-role"  
 }  
]
```

Le paramètre d'option ci-dessus définit le profil d'instance IAM. Vous pouvez spécifier l'ARN ou le nom du profil.

- Déterminez si le nouvel environnement est vert et prêt.

```
$ aws elasticbeanstalk describe-environments --environment-names my-env
```

Si le nouvel environnement n'est pas vert et prêt, vous devez décider si vous souhaitez recommencer l'opération ou laisser l'environnement dans son état actuel pour enquête. Veillez à mettre fin à l'environnement lorsque vous avez terminé, puis nettoyez toutes les ressources inutilisées.

#### Note

Vous pouvez régler la période d'expiration si l'environnement ne se lance pas dans un délai raisonnable.

## Création d'environnements Elastic Beanstalk avec l'API

- Appelez CheckDNSAvailability avec le paramètre suivant :

- CNAMEPrefix = SampleApp

### Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?CNAMEPrefix=sampleapplication  
&Operation=CheckDNSAvailability  
&AuthParams
```

2. Appelez `DescribeApplicationVersions` avec les paramètres suivants :

- `ApplicationName = SampleApp`
- `VersionLabel = Version2`

### Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp  
&VersionLabel=Version2  
&Operation=DescribeApplicationVersions  
&AuthParams
```

3. Appelez `CreateConfigurationTemplate` avec les paramètres suivants :

- `ApplicationName = SampleApp`
- `TemplateName = MyConfigTemplate`
- `SolutionStackName = 64bit%20Amazon%20Linux%202015.03%20v2.0.0%20running%20Ruby%202.2%20(Passenger%20Standalone)`

### Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp  
&TemplateName=MyConfigTemplate  
&Operation/CreateConfigurationTemplate  
&SolutionStackName=64bit%20Amazon%20Linux%202015.03%20v2.0.0%20running%20Ruby%202.2%20(Passenger%20Standalone)  
&AuthParams
```

4. Appelez `CreateEnvironment` avec l'un des ensembles de paramètres suivants.

- a. Utilisez les paramètres suivants pour un niveau d'environnement « serveur web » :

- `EnvironmentName = SampleAppEnv2`
- `VersionLabel = Version2`
- `Description = description`
- `TemplateName = MyConfigTemplate`
- `ApplicationName = SampleApp`
- `CNAMEPrefix = sampleapplication`
- `OptionSettings.member.1.Namespace = aws:autoscaling:launchconfiguration`
- `OptionSettings.member.1.OptionName = IamInstanceProfile`
- `OptionSettings.member.1.Value = aws-elasticbeanstalk-ec2-role`

### Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp  
&VersionLabel=Version2
```

```
&EnvironmentName=SampleAppEnv2
&TemplateName=MyConfigTemplate
&CNAMEPrefix=sampleapplication
&Description=description
&Operation/CreateEnvironment
&OptionSettings.member.1.Namespace=aws%3Aautoscaling%3Alaunchconfiguration
&OptionSettings.member.1.OptionName=IamInstanceProfile
&OptionSettings.member.1.Value=aws-elasticbeanstalk-ec2-role
&AuthParams
```

- b. Utilisez les paramètres suivants pour un niveau d'environnement de travail :
- EnvironmentName = SampleAppEnv2
  - VersionLabel = Version2
  - Description = description
  - TemplateName = MyConfigTemplate
  - ApplicationName = SampleApp
  - Tier = Worker
  - OptionSettings.member.1.Namespace = aws:autoscaling:launchconfiguration
  - OptionSettings.member.1.OptionName = IamInstanceProfile
  - OptionSettings.member.1.Value = aws-elasticbeanstalk-ec2-role
  - OptionSettings.member.2.Namespace = aws:elasticbeanstalk:sqsd
  - OptionSettings.member.2.OptionName = WorkerQueueURL
  - OptionSettings.member.2.Value = sqsd.elasticbeanstalk.us-east-2.amazonaws.com
  - OptionSettings.member.3.Namespace = aws:elasticbeanstalk:sqsd
  - OptionSettings.member.3.OptionName = HttpPath
  - OptionSettings.member.3.Value = /
  - OptionSettings.member.4.Namespace = aws:elasticbeanstalk:sqsd
  - OptionSettings.member.4.OptionName = MimeType
  - OptionSettings.member.4.Value = application/json
  - OptionSettings.member.5.Namespace = aws:elasticbeanstalk:sqsd
  - OptionSettings.member.5.OptionName = HttpConnections
  - OptionSettings.member.5.Value = 75
  - OptionSettings.member.6.Namespace = aws:elasticbeanstalk:sqsd
  - OptionSettings.member.6.OptionName = ConnectTimeout
  - OptionSettings.member.6.Value = 10
  - OptionSettings.member.7.Namespace = aws:elasticbeanstalk:sqsd
  - OptionSettings.member.7.OptionName = InactivityTimeout
  - OptionSettings.member.7.Value = 10
  - OptionSettings.member.8.Namespace = aws:elasticbeanstalk:sqsd
  - OptionSettings.member.8.OptionName = VisibilityTimeout
  - OptionSettings.member.8.Value = 60
  - OptionSettings.member.9.Namespace = aws:elasticbeanstalk:sqsd
  - OptionSettings.member.9.OptionName = RetentionPeriod
  - OptionSettings.member.9.Value = 345600

### Example

```
https://elasticbeanstalk.us-east-2.amazonaws.com/?ApplicationName=SampleApp
&VersionLabel=Version2
&EnvironmentName=SampleAppEnv2
&TemplateName=MyConfigTemplate
&Description=description
&Tier=Worker
&Operation=CreateEnvironment
&OptionSettings.member.1.Namespace=aws%3Aautoscaling%3Alaunchconfiguration
&OptionSettings.member.1.OptionName=IamInstanceProfile
&OptionSettings.member.1.Value=aws-elasticbeanstalk-ec2-role
&OptionSettings.member.2.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.2.OptionName=WorkerQueueURL
&OptionSettings.member.2.Value=sqsd.elasticbeanstalk.us-east-2.amazonaws.com
&OptionSettings.member.3.Namespace=aws%3Aelasticbeanstalk%3sqsd
&OptionSettings.member.3.OptionName=HttpPath
&OptionSettings.member.3.Value=%2F
&OptionSettings.member.4.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.4.OptionName=MimeType
&OptionSettings.member.4.Value=application%2Fjson
&OptionSettings.member.5.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.5.OptionName=HttpConnections
&OptionSettings.member.5.Value=75
&OptionSettings.member.6.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.6.OptionName=ConnectTimeout
&OptionSettings.member.6.Value=10
&OptionSettings.member.7.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.7.OptionName=InactivityTimeout
&OptionSettings.member.7.Value=10
&OptionSettings.member.8.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.8.OptionName=VisibilityTimeout
&OptionSettings.member.8.Value=60
&OptionSettings.member.9.Namespace=aws%3Aelasticbeanstalk%3Asqsd
&OptionSettings.member.9.OptionName=RetentionPeriod
&OptionSettings.member.9.Value=345600
&AuthParams
```

## Construction d'une URL Launch Now

Vous pouvez créer une URL personnalisée afin que tout le monde puisse rapidement déployer et exécuter une application web prédéfinie dans AWS Elastic Beanstalk. On l'appelle une URL Launch Now. Vous pourrez avoir besoin d'une URL Launch Now, par exemple, pour illustrer une application web construite pour s'exécuter sur Elastic Beanstalk. Avec une URL Launch Now, vous pouvez utiliser des paramètres pour ajouter les informations requises à l'assistant Create Application à l'avance. Une fois ces informations ajoutées à l'assistant, tout le monde peut utiliser le lien de l'URL pour lancer un environnement Elastic Beanstalk avec votre source d'application web en quelques étapes. Cela signifie que les utilisateurs ne doivent pas télécharger manuellement ou spécifier l'emplacement du groupe source de l'application. Ils ne doivent pas non plus fournir d'informations supplémentaires à l'assistant.

Une URL Launch Now fournit à Elastic Beanstalk les informations minimales requises pour créer une application : le nom de l'application, la pile de solutions, le type d'instance et le type d'environnement. Elastic Beanstalk utilise les valeurs par défaut pour d'autres détails de configuration qui ne sont pas explicitement spécifiés dans votre URL Launch Now personnalisée.

Une URL Launch Now utilise la syntaxe d'URL standard. Pour plus d'informations, consultez [RFC 3986 - Uniform Resource Identifier \(URI\): Generic Syntax](#).

## Paramètres d'URL

L'URL doit contenir les paramètres suivants, qui sont sensibles à la casse :

- région – Spécifier une région AWS. Pour obtenir la liste des régions prises en charge par Elastic Beanstalk, consultez [Points de terminaison et quotas AWS Elastic Beanstalk](#) dans la Référence générale AWS.
- applicationName : spécifiez le nom de votre application. Elastic Beanstalk affiche le nom de l'application dans la console Elastic Beanstalk pour le distinguer des autres applications. Par défaut, le nom de l'application constitue également la base du nom de l'environnement et de l'URL de l'environnement.
- platform : spécifiez la version de plateforme à utiliser pour l'environnement. Utilisez l'une des méthodes suivantes, puis encodez votre choix sous forme d'URL :
  - Spécifiez un ARN de plateforme sans version. Elastic Beanstalk sélectionne la dernière version majeure de la plateforme correspondante. Par exemple, pour sélectionner la dernière version de la plateforme Python 3.6, spécifiez `Python 3.6 running on 64bit Amazon Linux`.
  - Spécifiez le nom de la plateforme. Elastic Beanstalk sélectionne la dernière version du dernier runtime linguistique de la plateforme (par exemple, `Python`).

Pour obtenir la description de toutes les plateformes disponibles et de leurs versions, veuillez consulter [Plateformes prises en charge par Elastic Beanstalk \(p. 31\)](#).

Vous pouvez utiliser l'[AWS Command Line Interface](#) (AWS CLI) pour obtenir la liste de toutes les versions de plateforme disponibles avec leur ARN respectif. La commande `list-platform-versions` affiche des informations détaillées sur toutes les versions de plateforme disponibles. Utilisez l'argument `--filters` pour définir la liste. Par exemple, vous pouvez restreindre la liste de manière à n'afficher que les versions de plateforme d'un langage spécifique.

L'exemple suivant interroge toutes les versions de plateforme Python et affiche le résultat via une série de commandes. Le résultat est une liste d'ARN de version de plateforme (sans l'indication finale `/version`), dans un format contrôlable de visu, sans codage d'URL.

```
$ aws elasticbeanstalk list-platform-versions --filters
>Type="PlatformName",Operator="contains",Values="Python" | grep PlatformArn | awk -F
'"' '{print $4}' | awk -F '/' '{print $2}'
Preconfigured Docker - Python 3.4 running on 64bit Debian
Preconfigured Docker - Python 3.4 running on 64bit Debian
Python 2.6 running on 32bit Amazon Linux
Python 2.6 running on 32bit Amazon Linux 2014.03
...
Python 3.6 running on 64bit Amazon Linux
```

L'exemple suivant ajoute une commande Perl pour le dernier exemple, afin de coder la sortie sous forme d'URL.

```
$ aws elasticbeanstalk list-platform-versions --filters
>Type="PlatformName",Operator="contains",Values="Python" | grep PlatformArn | awk
-F '"' '{print $4}' | awk -F '/' '{print $2}' | perl -MURI::Escape -ne 'chomp;print
uri_escape($_),"\n"'
Preconfigured%20Docker%20-%20Python%203.4%20running%20on%2064bit%20Debian
Preconfigured%20Docker%20-%20Python%203.4%20running%20on%2064bit%20Debian
Python%202.6%20running%20on%2032bit%20Amazon%20Linux
Python%202.6%20running%20on%2032bit%20Amazon%20Linux%202014.03
...
Python%203.6%20running%20on%2064bit%20Amazon%20Linux
```

Une URL Launch Now peut en option contenir les paramètres suivants. Si vous n'incluez pas les paramètres facultatifs dans votre URL Launch Now, Elastic Beanstalk utilise les valeurs par défaut pour créer et exécuter votre application. Si vous n'incluez pas le paramètre sourceBundleUrl, Elastic Beanstalk utilise l'exemple d'application par défaut pour la plateforme spécifiée.

- `sourceBundleUrl` : spécifiez l'emplacement du bundle de fichiers source de votre application web au format URL. Par exemple, si vous avez chargé votre bundle de fichiers source dans un compartiment Amazon S3, vous pouvez spécifier la valeur du paramètre `sourceBundleUrl` en tant que `https://mybucket.s3.amazonaws.com/myobject`.

#### Note

Vous pouvez spécifier la valeur du paramètre `sourceBundleUrl` en tant qu'URL HTTP, mais le navigateur web de l'utilisateur va convertir les caractères en fonction des besoins en appliquant un codage d'URL HTML.

- `environmentType` : indiquez si l'environnement est à charge équilibrée et évolutive ou s'il s'agit d'une seule instance. Pour plus d'informations, consultez [Types d'environnement \(p. 519\)](#). Vous pouvez spécifier `LoadBalancing` ou `SingleInstance` comme la valeur de paramètre.
- `tierName` : spécifiez si l'environnement prend en charge une application web traitant les demandes web ou une application web exécutant les tâches en arrière-plan. Pour plus d'informations, consultez [Environnements de travail Elastic Beanstalk \(p. 521\)](#). Vous pouvez spécifier `WebServer` ou `Worker`.
- `instanceType` : spécifiez un serveur dont les caractéristiques (y compris la taille de la mémoire et la puissance de l'UC) sont les mieux adaptées à votre application. Elastic Beanstalk ne prend pas en charge les types d'instance Amazon EC2 Mac et Amazon EC2 Graviton (basées sur ARM) pour le moment. Pour de plus amples informations sur les familles d'instance et types d'instance Amazon EC2, consultez [Types d'instance](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux ou [Types d'instance](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows. Pour de plus amples informations sur les types d'instance disponibles dans les régions, consultez [Types d'instance disponibles](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux ou [Types d'instance disponibles](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.
- `withVpc` : spécifiez si vous créez l'environnement dans un Amazon VPC. Vous pouvez spécifier `true` ou `false`. Pour de plus amples informations sur l'utilisation d'Elastic Beanstalk avec Amazon VPC, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#).
- `withRds` : spécifiez si vous créez une instance de base de données Amazon RDS avec cet environnement. Pour plus d'informations, consultez [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#). Vous pouvez spécifier `true` ou `false`.
- `rdsDBEngine` : spécifiez le moteur de base de données que vous souhaitez utiliser pour vos instances Amazon EC2 dans cet environnement. Vous pouvez spécifier `mysql`, `oracle-se1`, `sqlserver-ex`, `sqlserver-web` ou `sqlserver-se`. La valeur par défaut est `mysql`.
- `rdsDBAllocatedStorage` : spécifiez la taille de stockage allouée de base de données en gigaoctets (Go). Vous pouvez spécifier les valeurs suivantes :
  - MySQL – 5 à 1024. La valeur par défaut est 5.
  - Oracle – 10 à 1024. La valeur par défaut est 10.
  - Microsoft SQL Server Express Edition – 30.
  - Microsoft SQL Server Web Edition – 30.
  - Microsoft SQL Server Standard Edition – 200.
- `rdsDBInstanceClass` : spécifiez le type d'instance de base de données. La valeur par défaut est `db.t2.micro` (`db.m1.large` pour un environnement qui ne s'exécute pas dans un Amazon VPC). Pour obtenir la liste des classes d'instance de base de données prises en charge par Amazon RDS, reportez-vous à [Classe d'instance de base de données](#) dans le Guide de l'utilisateur Amazon Relational Database Service.
- `rdsMultiAZDatabase` : spécifiez si Elastic Beanstalk a besoin de créer l'instance de base de données sur plusieurs zones de disponibilité. Vous pouvez spécifier `true` ou `false`. Pour de plus amples

informations sur les déploiements multi-AZ avec Amazon RDS, consultez [Régions et zones de disponibilité](#) dans le Guide de l'utilisateur Amazon Relational Database Service.

- rdsDBDeletionPolicy : spécifiez s'il faut supprimer ou prendre un instantané de l'instance de base de données au moment de l'arrêt de l'environnement. Vous pouvez spécifier Delete ou Snapshot.

## Example

Voici un exemple d'URL Launch Now. Après que vous avez construit la vôtre, vous pouvez la donner à vos utilisateurs. Par exemple, vous pouvez intégrer l'URL dans une page web ou dans des documents de formation. Lorsque les utilisateurs créent une application à l'aide de l'URL Launch Now, l'assistant Elastic Beanstalk de création d'une application n'a besoin d'aucune entrée supplémentaire.

```
https://console.aws.amazon.com/elasticbeanstalk/home?region=us-west-2#/newApplication?applicationName=YourCompanySampleApp&platform=PHP%207.3%20running%20on%2064bit%20Amazon%20Linux&sourceBundleUrl=http://s3.amazonaws.com/mybucket/myobject&environmentType=SingleInstance&tierName=WebServer&instanceType=m1.small&withVpc=true&withRds=t
```

Lorsque les utilisateurs cliquent sur une URL Launch Now, Elastic Beanstalk affiche une page similaire à ce qui suit.



## Create a web app

Create a new application and environment with a sample application or your own code. By creating an environment, AWS Elastic Beanstalk manages AWS resources and permissions on your behalf. [Learn more](#)

### Application information

**Application name**

YourCompanySampleApp

Up to 100 Unicode characters, not including forward slash (/).

### Environment information

Choose the name, subdomain, and description for your environment. These cannot be changed later.

**Environment name**

Yourcompanysampleapp-env

**Domain**

Leave blank for autogenerated value

.us-east-1.elasticbeanstalk.com

[Check availability](#)

**Description**

### Base configuration

**Tier** Web Server ([Choose tier](#))

**Platform**

Preconfigured platform

Platforms published and maintained by AWS Elastic Beanstalk.

PHP

Custom platform NEW

Platforms created and owned by you. [Learn more](#)

-- Choose a custom platform --

**Application code**

Sample application

Get started right away with sample code.

Upload your code

Upload a source bundle from your computer or copy one from Amazon S3.

473

[Upload](#)

ZIP or WAR

#### Pour utiliser l'URL Launch Now

1. Cliquez sur l'URL Launch Now.
2. Une fois que la console Elastic Beanstalk s'ouvre, sur la page Création d'une application web, cliquez sur Review and launch (Revoir et lancer) pour afficher les paramètres qu'Elastic Beanstalk utilise pour créer l'application et lancer l'environnement dans lequel l'application s'exécute.
3. Sur la page Configurer, choisissez Créer une application pour créer l'application.

## Création et mise à jour de groupes d'environnements Elastic Beanstalk

Avec l'API `Compose Environments` AWS Elastic Beanstalk, vous pouvez créer et mettre à jour les groupes d'environnements Elastic Beanstalk au sein d'une seule application. Chaque environnement du groupe peut exécuter un composant distinct d'une application d'architecture orientée services. L'API `Compose Environments` utilise une liste des versions d'application et un nom de groupe facultatif. Le service Elastic Beanstalk crée un environnement pour chaque version d'application. Si les environnements existent déjà, il y déploie les versions d'application.

Créez des liens entre les environnements Elastic Beanstalk afin de définir une dépendance entre deux environnements. Lorsque vous créez un groupe d'environnements avec l'API `Compose Environments`, Elastic Beanstalk crée des environnements dépendants, mais uniquement une fois que leurs dépendances sont opérationnelles. Pour de plus amples informations sur les liens d'environnement, veuillez consulter [Création de liens entre les environnements Elastic Beanstalk \(p. 530\)](#).

L'API `Compose Environments` utilise un [manifeste d'environnement \(p. 785\)](#) pour stocker les détails de configuration qui sont partagés par les groupes d'environnements. Chaque composant doit inclure un fichier de configuration `env.yaml` dans le bundle source de son application, qui spécifie les paramètres utilisés pour créer son environnement.

Pour utiliser `Compose Environments`, vous devez spécifier les paramètres `EnvironmentName` et `SolutionStack` dans le manifeste d'environnement pour chaque composant d'application.

Vous pouvez utiliser l'API `Compose Environments` avec l'interface de ligne de commande Elastic Beanstalk (EB CLI), la AWS CLI ou un kit SDK. Pour obtenir les instructions sur l'interface de ligne de commande Elastic Beanstalk (EB CLI), veuillez consulter [Gestion de plusieurs environnements Elastic Beanstalk en tant que groupe avec l'interface de ligne de commande EB \(p. 1058\)](#).

## Utilisation de l'API `Compose Environments`

Par exemple, vous pouvez créer une application nommée `Media Library` qui permet aux utilisateurs de charger et gérer des images et des vidéos stockées dans Amazon Simple Storage Service (Amazon S3). L'application dispose d'un environnement frontal, `front`, qui exécute une application web permettant aux utilisateurs de charger et de télécharger des fichiers individuels, de visualiser leur bibliothèque et d'initier des tâches de traitement par lots.

Au lieu de traiter les tâches directement, l'application frontale les ajoute à une file d'attente Amazon SQS. Le deuxième environnement, `worker`, extrait les tâches de la file d'attente et les traite. `worker` utilise un type d'instance G2 doté d'un GPU très performant, tandis que `front` peut s'exécuter sur un type d'instance générique plus économique.

Vous pouvez organiser le dossier du projet, `Media Library`, dans des répertoires distincts pour chaque composant. Chaque répertoire contient un fichier de définition d'environnement (`env.yaml`) avec le code source pour chacun d'eux :

```
~/workspace/media-library
```

```
|-- front
|   '-- env.yaml
`-- worker
    '-- env.yaml
```

Les listes suivantes spécifient le fichier `env.yaml` pour chaque composant de l'application.

**`~/workspace/media-library/front/env.yaml`**

```
EnvironmentName: front+
EnvironmentLinks:
    "WORKERQUEUE" : "worker+"
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentTier:
    Name: WebServer
    Type: Standard
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.4 running Java 8
OptionSettings:
    aws:autoscaling:launchconfiguration:
        InstanceType: m4.large
```

**`~/workspace/media-library/worker/env.yaml`**

```
EnvironmentName: worker+
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentTier:
    Name: Worker
    Type: SQS/HTTP
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.4 running Java 8
OptionSettings:
    aws:autoscaling:launchconfiguration:
        InstanceType: g2.2xlarge
```

Une fois que vous avez [créé une version d'application \(p. 409\)](#) pour les deux composants de l'application (frontal (`front-v1`) et travail (`worker-v1`)), vous devez appeler l'API Compose Environments avec les noms de version. Dans cet exemple, nous utilisons la AWS CLI pour appeler l'API.

```
# Create application versions for each component:
-$ aws elasticbeanstalk create-application-version --application-name media-
library --version-label front-v1 --process --source-bundle S3Bucket="DOC-EXAMPLE-
BUCKET",S3Key="front-v1.zip"
{
    "ApplicationVersion": {
        "ApplicationName": "media-library",
        "VersionLabel": "front-v1",
        "Description": "",
        "DateCreated": "2015-11-03T23:01:25.412Z",
        "DateUpdated": "2015-11-03T23:01:25.412Z",
        "SourceBundle": {
            "S3Bucket": "DOC-EXAMPLE-BUCKET",
            "S3Key": "front-v1.zip"
        }
    }
}
-$ aws elasticbeanstalk create-application-version --application-name media-
library --version-label worker-v1 --process --source-bundle S3Bucket="DOC-EXAMPLE-
BUCKET",S3Key="worker-v1.zip"
{
    "ApplicationVersion": {
        "ApplicationName": "media-library",
        "VersionLabel": "worker-v1",
```

```

    "Description": "",
    "DateCreated": "2015-11-03T23:01:48.151Z",
    "DateUpdated": "2015-11-03T23:01:48.151Z",
    "SourceBundle": {
        "S3Bucket": "DOC-EXAMPLE-BUCKET",
        "S3Key": "worker-v1.zip"
    }
}
# Create environments:
~$ aws elasticbeanstalk compose-environments --application-name media-library --group-name dev --version-labels front-v1 worker-v1

```

Le troisième appel crée deux environnements : `front-dev` et `worker-dev`. L'API crée les noms des environnements en concaténant la valeur `EnvironmentName` spécifiée dans le fichier `env.yaml` avec l'option `group_name` spécifiée dans l'appel `Compose Environments`, séparées par un tiret. La longueur totale de ces deux options et du trait d'union ne doit pas dépasser la longueur maximale autorisée pour le nom d'un environnement, soit 23 caractères.

L'application en cours d'exécution dans l'environnement `front-dev` peut accéder au nom de la file d'attente Amazon SQS; associée à l'environnement `worker-dev` en lisant la variable `WORKERQUEUE`. Pour de plus amples informations sur les liens d'environnement, veuillez consulter [Création de liens entre les environnements Elastic Beanstalk \(p. 530\)](#).

## Déploiement d'applications dans des environnements Elastic Beanstalk

Vous pouvez utiliser la console AWS Elastic Beanstalk pour télécharger une [solution groupée \(p. 415\)](#) source mise à jour et la déployer dans votre environnement Elastic Beanstalk ou redéployer une version précédemment téléchargée.

Chaque utilisateur est identifié par un ID de déploiement. Les ID de déploiement démarrent à 1 et augmentent d'un à chaque changement de configuration d'instance et de déploiement. Si vous activez les [rapports améliorés sur l'état \(p. 837\)](#), Elastic Beanstalk affiche l'ID de déploiement dans la [console de surveillance de l'état \(p. 849\)](#) et dans la [CLI EB \(p. 1053\)](#) quand il rapporte l'état de santé de l'instance. L'ID de déploiement vous aide à déterminer l'état de votre environnement lorsqu'une mise à jour propagée échoue.

Elastic Beanstalk fournit plusieurs stratégies et paramètres de déploiement. Pour de plus amples informations sur la configuration d'une stratégie et d'autres paramètres, veuillez consulter [the section called "Options de déploiement" \(p. 479\)](#). Le tableau suivant répertorie les stratégies et les types d'environnements qui les prennent en charge.

### Stratégies de déploiement prises en charge

Stratégie de déploiement	Environnements à charge équilibrée	Environnements d'instance unique	Environnements Windows Server existants†
Simultanément	✓ Oui	✓ Oui	✓ Oui
Propagation	✓ Oui	✗ Non	✓ Oui
Propagation avec un lot supplémentaire	✓ Oui	✗ Non	✗ Non

Stratégie de déploiement	Environnements à charge équilibrée	Environnements d'instance unique	Environnements Windows Server existants†
Immuable	✓ Oui	✓ Oui	✗ Non
Répartition du trafic	✓ Oui (Équilibrage de charge Application Load Balancer)	✗ Non	✗ Non

† Dans ce tableau, un environnement Windows Server existant est un environnement basé sur une configuration de plateforme Windows Server qui utilise une version d'IIS antérieure à IIS 8.5.

#### Warning

Certaines stratégies remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous les équilibres de rafale Amazon EC2 cumulés. Une telle situation se produit dans les cas suivants :

- Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
- Mises à jour immuables
- Déploiements avec mises à jour immuables ou fractionnement du trafic activé

## Choix d'une stratégie de déploiement

Le choix de la stratégie de déploiement adaptée pour votre application est un compromis entre plusieurs considérations et dépend de vos besoins particuliers. La page [the section called “Options de déploiement” \(p. 479\)](#) contient de plus amples informations sur chaque stratégie et une description détaillée du fonctionnement de certaines d'entre elles.

La liste suivante fournit des informations récapitulatives sur les différentes stratégies de déploiement et ajoute des considérations connexes.

- Simultanée – Méthode de déploiement la plus rapide. Convient si vous pouvez accepter une courte perte de service et si des déploiements rapides sont importants pour vous. Avec cette méthode, Elastic Beanstalk déploie la nouvelle version de l'application sur chaque instance. Ensuite, le proxy Web ou le serveur d'applications peut avoir besoin de redémarrer. Par conséquent, votre application peut être indisponible pour les utilisateurs (ou présenter une faible disponibilité) pendant une courte période.
- Propagation – Évite les temps d'arrêt et minimise la disponibilité réduite, au prix d'un temps de déploiement plus long. Convient si vous ne pouvez accepter aucune période de service complètement perdu. Avec cette méthode, votre application est déployée dans votre environnement, un lot d'instances à la fois. La plupart de la bande passante est conservée tout au long du déploiement.
- Propagation avec un lot supplémentaire – Évite toute disponibilité réduite, au prix d'un temps de déploiement encore plus long que la méthode Propagation. Convient si vous devez conserver la même bande passante tout au long du déploiement. Avec cette méthode, Elastic Beanstalk lance un lot supplémentaire d'instances, puis effectue un déploiement par propagation. Le lancement du lot supplémentaire prend du temps et garantit que la même bande passante est conservée tout au long du déploiement.
- Immuable – Méthode de déploiement plus lente, qui garantit que la nouvelle version de votre application est toujours déployée sur les nouvelles instances, au lieu de mettre à jour les instances existantes. Elle présente également l'avantage supplémentaire d'une restauration rapide et sûre en cas d'échec du déploiement. Avec cette méthode, Elastic Beanstalk effectue une [mise à jour immuable \(p. 493\)](#) pour déployer votre application. Dans une mise à jour immuable, un second groupe Auto Scaling est lancé dans votre environnement et la nouvelle version sert le trafic parallèlement à l'ancienne version jusqu'à ce que les nouvelles instances transmettent les vérifications de l'état.

- Répartition du trafic – Méthode de déploiement avec tests Canary. Convient si vous souhaitez tester l'intégrité de la nouvelle version de votre application en utilisant une partie du trafic entrant, tout en conservant le reste du trafic servi par l'ancienne version de l'application.

Le tableau suivant compare les propriétés des méthodes de déploiement :

#### Méthodes de déploiement

Méthode	Impact d'un échec de déploiement	Temps de déploiement	Aucun interrup.	Pas de modif.	Procédure de restauration DNS	Code déployé sur
Simultanée	Temps d'arrêt	⌚	X Non	✓ Oui	Redéploie instances manuelles et existantes	instances existantes
Propagation	Lot unique hors service. Tout lot ayant abouti avant l'échec de l'exécution de la nouvelle version de l'application	⌚⌚†	✓ Oui	✓ Oui	Redéploie instances manuelles et existantes	instances existantes
Propagation avec un lot supplémentaire	Même si le premier lot échoue, sinon similaire à Propagation.	⌚⌚⌚†	✓ Oui	✓ Oui	Redéploie instances manuelles et existantes	instances existantes
Immuable	Minimale	⌚⌚⌚⌚	✓ Oui	✓ Oui	Résilie les nouvelles instances	Nouvelles instances
Répartie du trafic	Pourcentage du trafic client acheminé vers la nouvelle version temporairement affecté	⌚⌚⌚⌚††	✓ Oui	✓ Oui	Réachète le trafic et résilier les nouvelles instances	instances existantes
Bleu/vert	Minimale	⌚⌚⌚⌚	✓ Oui	X Non	Permet de changer les URL	Nouvelles instances

† Varie en fonction de la taille du lot.

†† Varie en fonction du réglage de l'option du temps d'évaluation.

## Déploiement d'une nouvelle version de l'application

Vous pouvez effectuer des déploiements à partir du tableau de bord de votre environnement.

Pour déployer une nouvelle version de l'application dans un environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Upload and Deploy (Charger et déployer).
4. Utilisez le formulaire à l'écran pour télécharger le bundle source de l'application.
5. Choisissez Deploy (Déployer).

## Redéploiement d'une version précédente

Vous pouvez également déployer une version précédemment chargée de votre application sur n'importe lequel de ses environnements depuis la page des versions d'application.

Pour déployer une version de l'application existante dans un environnement existant

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

#### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Application versions (Versions d'application).
4. Sélectionnez la version d'application à déployer.
5. Choisissez Actions, puis Deploy (Déployer).
6. Choisissez un environnement, puis Deploy (Déployer).

## Autres méthodes de déploiement de votre application

Si vous procédez à des déploiements fréquents, pensez à utiliser l'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (CLI EB) pour gérer vos environnements. La CLI EB crée un référentiel parallèlement à votre code source. Elle peut également créer un bundle de fichiers source, le charger dans Elastic Beanstalk et le déployer à l'aide d'une seule commande.

Pour les déploiements qui reposent sur des modifications de configuration de ressources ou une nouvelle version qui ne peut pas s'exécuter parallèlement à l'ancienne version, vous pouvez lancer un nouvel environnement avec la nouvelle version et effectuer un échange CNAME pour un [déploiement bleu/vert \(p. 486\)](#).

## Paramètres et stratégies de déploiement

AWS Elastic Beanstalk fournit plusieurs options pour le traitement des [déploiements \(p. 476\)](#), y compris des stratégies de déploiement (Simultanée, Propagation, Propagation avec un lot supplémentaire, Immutable et Répartition du trafic) et des options vous permettant de configurer la taille de lot et le comportement des vérifications de l'état au cours des déploiements. Par défaut, votre environnement utilise des déploiements « tout à la fois ». Si vous avez créé l'environnement avec la CLI EB et s'il s'agit d'un environnement évolutif (vous n'avez pas spécifié l'option `--single`), il utilise des déploiements par propagation.

Avec les déploiements propagés, Elastic Beanstalk divise les instances Amazon EC2 de l'environnement en lots et déploie la nouvelle version de l'application dans un lot à la fois. Les instances restantes sont laissées dans l'environnement exécutant l'ancienne version de l'application. Lors d'un déploiement par propagation, certaines instances traitent les demandes avec l'ancienne version d'application, tandis que les instances incluses dans les lots terminés traitent les autres demandes avec la nouvelle version. Pour plus d'informations, consultez [the section called "Fonctionnement de la propagation des déploiements" \(p. 483\)](#).

Afin de maintenir une pleine capacité au cours des déploiements, vous pouvez configurer votre environnement de façon à lancer un nouveau lot d'instances avant de mettre des instances hors service. Cette option correspond à un déploiement par propagation avec un lot supplémentaire. Une fois le déploiement terminé, Elastic Beanstalk résilie le lot d'instances supplémentaire.

Les déploiements immuables effectuent une [mise à jour immuable \(p. 493\)](#) pour lancer un ensemble complet de nouvelles instances exécutant la nouvelle version de l'application dans un groupe Auto Scaling distinct, parallèlement aux instances qui exécutent l'ancienne version. Les déploiements immuables peuvent éviter les problèmes causés par des propagations de déploiements partiellement terminées. Si les nouvelles instances ne réussissent pas les vérifications de l'état, Elastic Beanstalk les résilie tout en laissant intactes les instances d'origine.

Les déploiements avec répartition du trafic vous permettent d'effectuer des tests Canary dans le cadre du déploiement de votre application. Dans un déploiement avec répartition du trafic, Elastic Beanstalk lance un ensemble complet de nouvelles instances comme lors d'un déploiement immuable. Il transmet ensuite un pourcentage spécifié du trafic client entrant vers la nouvelle version de l'application pour une période d'évaluation spécifiée. Si les nouvelles instances restent en bonne santé, Elastic Beanstalk leur transmet l'ensemble du trafic et résilie les anciennes instances. Si les nouvelles instances ne réussissent pas les vérifications de l'état ou si vous choisissez d'abandonner le déploiement, Elastic Beanstalk renvoie le trafic vers les anciennes instances et résilie les nouvelles. Il n'y a jamais d'interruption de service. Pour plus d'informations, consultez [the section called "Fonctionnement des déploiements avec répartition du trafic" \(p. 484\)](#).

#### Warning

Certaines stratégies remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous les [équilibres de rafale Amazon EC2](#) cumulés. Une telle situation se produit dans les cas suivants :

- Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
- Mises à jour immuables
- Déploiements avec mises à jour immuables ou fractionnement du trafic activé

Si votre application ne réussit pas toutes les vérifications d'état, mais fonctionne tout de même correctement avec un statut d'état inférieur, vous pouvez autoriser les instances à réussir les vérifications d'état avec un statut inférieur ([Warning](#), par exemple), en modifiant l'option Seuil de bonne santé. Si vos déploiements échouent parce qu'ils ne réussissent pas les vérifications d'état et si vous avez besoin de forcer une mise à jour quel que soit le statut de l'état, sélectionnez l'option Ignorer la vérification de l'état.

Lorsque vous spécifiez une taille de lot pour la propagation des mises à jour, Elastic Beanstalk utilise également cette valeur pour les redémarrages progressifs de l'application. Utilisez les redémarrages progressifs lorsque vous souhaitez redémarrer les serveurs proxy et d'application exécutés sur les instances de votre environnement sans temps d'arrêt.

## Configuration des déploiements d'application

Dans la [console de gestion de l'environnement \(p. 429\)](#), activez et configurez les déploiements de version d'application par lots en modifiant l'option Updates and Deployments (Mises à jour et déploiements) sur la page Configuration de l'environnement.

### Pour configurer des déploiements (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Rolling updates and deployments (Propagation des mises à jour et déploiements), choisissez Edit (Modifier).
5. Dans la section Déploiements de l'application, choisissez une valeur pour l'option Stratégie de déploiement, les paramètres de lots et les options de vérification de l'état.
6. Choisissez Apply.

La section Déploiements de l'application de la page Propagation des mises à jour et déploiements propose les options suivantes pour les déploiements d'applications :

- Deployment policy (Stratégie de déploiement) – Choisissez l'une des options de déploiement suivantes :
  - All at once (Simultanée) – Déployez la nouvelle version dans toutes les instances en même temps. Pendant le déploiement, toutes les instances de votre environnement sont mises hors service pendant un temps limité.
  - Rolling (Propagation) – Déployez la nouvelle version par lots. Chaque lot est suspendu pendant la phase de déploiement. Par conséquent, la capacité de votre environnement est diminuée du nombre d'instances dans un lot.
  - Rolling with additional batch (Propagation avec un lot supplémentaire) – Déployez la nouvelle version par lots, en commençant par lancer un nouveau lot d'instances afin de garantir la pleine capacité au cours du processus de déploiement.
  - Immutable (Immuable) – Déployez la nouvelle version dans un nouveau groupe d'instances en effectuant une [mise à jour immuable \(p. 493\)](#).
  - Traffic splitting (Répartition du trafic) – Déployez la nouvelle version dans un nouveau groupe d'instances et répartissez temporairement le trafic client entrant entre la version de l'application existante et la nouvelle.

Pour les stratégies de déploiement Propagation et Propagation avec un lot supplémentaire, vous pouvez configurer :

- Batch size (Taille de lot) – Taille de l'ensemble d'instances à déployer dans chaque lot.

Choisissez Percentage (Pourcentage) pour configurer un pourcentage du nombre total d'instances EC2 dans le groupe Auto Scaling (jusqu'à 100 %), ou Fixed (Fixe) pour configurer un nombre fixe d'instances (jusqu'au nombre maximal d'instances dans la configuration Auto Scaling de votre environnement).

Pour la stratégie de déploiement Répartition du trafic, vous pouvez configurer les éléments suivants :

- Traffic split (Répartition du trafic) – Pourcentage initial du trafic client entrant que Elastic Beanstalk déplace vers les instances d'environnement exécutant la nouvelle version de l'application que vous déployez.

- Traffic splitting evaluation time (Temps d'évaluation de la répartition du trafic) – Période d'attente d'Elastic Beanstalk, en minutes, à la suite d'un déploiement sain initial avant de déplacer l'ensemble du trafic client entrant vers la nouvelle version de l'application que vous déployez.

The screenshot shows the AWS Elastic Beanstalk Configuration page for the 'GettingStartedApp-env' environment. The navigation path is: Elastic Beanstalk > Environments > GettingStartedApp-env > Configuration. The main heading is 'Modify rolling updates and deployments'. Under 'Application deployments', the 'Deployment policy' is set to 'All at once'. The 'Batch size' section shows 'Percentage' selected (radio button is checked), with a value of '100 % of instances at a time'. The 'Traffic split' section shows '10 % to new application version'. The 'Traffic splitting evaluation time' section shows '5 minutes'. A note below states: 'La section Préférences de déploiement inclut des options relatives aux vérifications de l'état.'

La section Préférences de déploiement inclut des options relatives aux vérifications de l'état.

- Ignore health check (Ignorer la vérification de l'état) – Empêche la restauration du déploiement lorsqu'un lot ne passe pas avec succès les vérifications de l'état avant l'expiration du délai de commande.
- Healthy threshold (Seuil de bonne santé) – Réduit le seuil au niveau auquel une instance est considérée comme saine pendant les déploiements par propagation, les mises à jour par propagation et les mises à jour immuables.
- Command timeout (Délai de commande) – Nombre de secondes d'attente pour qu'une instance devienne saine avant d'annuler le déploiement ou, si l'option Ignore health check (Ignorer la vérification de l'état) est sélectionnée, avant de passer au lot suivant.

The screenshot shows the 'Deployment preferences' section of the AWS Elastic Beanstalk developer guide. It includes fields for 'Ignore health check' (set to 'False'), 'Healthy threshold' (set to 'Ok'), and 'Command timeout' (set to '600'). Each field has a dropdown arrow icon to its right.

**Deployment preferences**  
Customize health check requirements and deployment timeouts.

Ignore health check  
False ▾  
Don't fail deployments due to health check failures.

Healthy threshold  
Ok ▾  
Lower the threshold for an instance in a batch to pass health checks during an update or deployment.

Command timeout  
600 ▾  
Change the amount of time in seconds that AWS Elastic Beanstalk allows an instance to complete deployment commands.

## Fonctionnement de la propagation des déploiements

Lors du traitement d'un lot, Elastic Beanstalk détache de l'équilibrEUR de charge toutes les instances du lot, déploie la nouvelle version de l'application, puis attache de nouveau les instances. Lorsque le [Connection Draining \(p. 570\)](#) est activé, Elastic Beanstalk draine les connexions existantes des instances Amazon EC2 dans chaque lot avant de commencer le déploiement.

Après avoir associé de nouveau les instances d'un lot à l'équilibrEUR de charge, Elastic Load Balancing attend qu'elles réussissent un nombre minimal de vérifications de l'état Elastic Load Balancing (valeur du paramètre Healthy check count threshold (Seuil du nombre de vérifications de l'état saines)), puis commence à acheminer le trafic vers ces instances. Si aucune [URL de vérification de l'état \(p. 571\)](#) n'est configurée, cela peut se produire très rapidement, car une instance réussit la vérification de l'état dès qu'elle peut accepter une connexion TCP. Si une URL de vérification de l'état est configurée, l'équilibrEUR de charge n'achemine pas le trafic vers les instances mises à jour tant qu'elles n'ont pas renvoyé un code d'état 200 OK en réponse à une demande HTTP GET envoyée à l'URL de vérification de l'état.

Elastic Beanstalk attend que toutes les instances du lot soient saines avant de passer au lot suivant. Avec les [rapports de base sur l'état \(p. 834\)](#), l'état de l'instance dépend du statut de vérification de l'état d'Elastic Load Balancing. Lorsque toutes les instances du lot ont réussi suffisamment de vérifications de l'état pour être considérées comme saines par Elastic Load Balancing, le traitement du lot est terminé. Si les [rapports améliorés sur l'état \(p. 837\)](#) sont activés, Elastic Beanstalk tient compte de plusieurs autres facteurs, dont le résultat des demandes entrantes. Avec les rapports sur l'état de santé améliorés, toutes les instances doivent réussir 12 vérifications de l'état consécutives avec un [statut OK \(p. 855\)](#) en deux minutes pour les environnements de serveurs web et 18 vérifications de l'état en 3 minutes pour les environnements de travail.

Si un lot d'instances ne devient pas sain avant l'expiration du [délai de commande \(p. 480\)](#), le déploiement échoue. Dès lors qu'un déploiement a échoué, [vérifiez l'état des instances de votre environnement \(p. 849\)](#) pour obtenir des informations sur la cause de la défaillance. Ensuite, effectuez un autre déploiement avec une version corrigée de votre application à restaurer ou une version dont vous savez qu'elle est correcte.

Si un déploiement échoue alors qu'un ou plusieurs lots ont été correctement créés, les lots terminés exécutent la nouvelle version de votre application, tandis que les lots en attente continuent d'exécuter l'ancienne version. Vous pouvez identifier la version en cours d'exécution sur les instances de votre environnement sur la page [Santé \(p. 850\)](#) de la console. Cette page affiche l'ID du déploiement le plus récent ayant été exécuté sur chaque instance de votre environnement. Si vous résiliez des instances à partir du déploiement ayant échoué, Elastic Beanstalk les remplace par des instances exécutant la version de l'application à partir du déploiement le plus récent ayant réussi.

## Fonctionnement des déploiements avec répartition du trafic

Les déploiements avec répartition du trafic vous permettent d'effectuer des tests Canary. Vous dirigez une partie du trafic client entrant vers la nouvelle version de votre application pour vérifier l'état de l'application avant de valider la nouvelle version et de diriger tout le trafic vers celle-ci.

Au cours d'un déploiement avec répartition du trafic, Elastic Beanstalk crée un nouvel ensemble d'instances dans un groupe Auto Scaling temporaire distinct. Elastic Beanstalk demande ensuite à l'équilibrEUR de charge de diriger un certain pourcentage du trafic entrant de votre environnement vers les nouvelles instances. Ensuite, Elastic Beanstalk contrôle l'état du nouvel ensemble d'instances pendant une durée configurée. Si tout va bien, Elastic Beanstalk déplace le trafic restant vers les nouvelles instances et les attache au groupe Auto Scaling d'origine de l'environnement, en remplaçant les anciennes instances. Elastic Beanstalk procÈde ensuite à un nettoyage en rÈsiliant les anciennes instances et en supprimant le groupe Auto Scaling temporaire.

### Note

La capacitÈ de l'environnement ne change pas lors d'un déploiement avec rÈpartition du trafic. Elastic Beanstalk lance le mÈme nombre d'instances dans le groupe Auto Scaling temporaire que dans le groupe Auto Scaling d'origine au dÈbut du déploiement. Il maintient ensuite un nombre constant d'instances dans les deux groupes Auto Scaling pour toute la durÈe du déploiement. Tenez compte de ce fait lors de la configuration du temps d'évaluation de la rÈpartition du trafic de l'environnement.

La restauration du déploiement vers la version précédente de l'application est rapide et n'affecte pas le service sur le trafic client. Si les nouvelles instances ne réussissent pas les vÈrifications de l'état ou si vous choisissez d'abandonner le déploiement, Elastic Beanstalk renvoie le trafic vers les anciennes instances et rÈsile les nouvelles. Pour abandonner un déploiement, accÈdez à la page de prÈsentation de l'environnement dans la console Elastic Beanstalk et choisissez Abort current operation (Interrompre l'opÈration en cours) dans Environment actions (Actions d'environnement). Vous pouvez également appeler l'API [AbortEnvironmentUpdate](#) ou la commande d'AWS CLI ´quivalente.

Les déploiements avec rÈpartition du trafic exigent un ´quilibrEUR de charge Application Load Balancer. Elastic Beanstalk utilise ce type d'´quilibrEUR de charge par dÈfaut lorsque vous crÈez votre environnement ´ l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande (CLI) EB.

## Espaces de noms pour les options de déploiements

Vous pouvez utiliser les [options de configuration \(p. 658\)](#) dans l'espace de noms [aws:elasticbeanstalk:command \(p. 699\)](#) pour configurer vos dÈploiements. Si vous choisissez la stratÈgie avec rÈpartition du trafic, des options supplémentaires pour cette stratÈgie sont disponibles dans l'espace de noms [aws:elasticbeanstalk:trafficsplitting \(p. 712\)](#).

Utilisez l'option DeploymentPolicy pour dÈfinir le type de dÈploiement. Les valeurs suivantes sont prÈes en charge :

- AllAtOnce – Désactive les propagations de dÈploiements et effectue systÈmatiquement un dÈploiement dans toutes les instances en mÈme temps.

- **Rolling** – Permet d'effectuer des propagations de déploiements standard.
- **RollingWithAdditionalBatch** – Lance un lot d'instances supplémentaire avant de commencer le déploiement afin de maintenir une pleine capacité.
- **Immutable** – Effectue une [mise à jour immuable \(p. 493\)](#) pour chaque déploiement.
- **TrafficSplitting** – Effectue des déploiements avec répartition du trafic pour réaliser des tests Canary sur les déploiements de votre application.

Lorsque vous activez les propagations de déploiements, définissez les options `BatchSize` et `BatchSizeType` afin de configurer la taille de chaque lot. Par exemple, pour déployer 25 % de toutes les instances dans chaque lot, spécifiez les options et valeurs suivantes.

Example `.ebextensions/rolling-updates.config`

```
option_settings:  
  aws:elasticbeanstalk:command:  
    DeploymentPolicy: Rolling  
    BatchSizeType: Percentage  
    BatchSize: 25
```

Pour déployer jusqu'à cinq instances dans chaque lot, quel que soit le nombre d'instances en cours d'exécution, et mettre en place un lot supplémentaire de cinq instances exécutant la nouvelle version avant de mettre des instances hors service, spécifiez les options et valeurs ci-après.

Example `.ebextensions/rolling-additionalbatch.config`

```
option_settings:  
  aws:elasticbeanstalk:command:  
    DeploymentPolicy: RollingWithAdditionalBatch  
    BatchSizeType: Fixed  
    BatchSize: 5
```

Pour effectuer une mise à jour immuable pour chaque déploiement avec un seuil de vérification de l'état défini sur Avertissement et poursuivre le déploiement même si les instances d'un lot ne réussissent pas les vérifications de l'état dans un délai de 15 minutes, spécifiez les options et les valeurs ci-après.

Example `.ebextensions/immutable-ignorehealth.config`

```
option_settings:  
  aws:elasticbeanstalk:command:  
    DeploymentPolicy: Immutable  
    HealthCheckSuccessThreshold: Warning  
    IgnoreHealthCheck: true  
    Timeout: "900"
```

Pour effectuer des déploiements avec répartition du trafic, avec le transfert de 15 % du trafic client vers la nouvelle version de l'application et l'évaluation de l'intégrité pendant 10 minutes, spécifiez les options et valeurs suivantes.

Example `.ebextensions/traffic-splitting.config`

```
option_settings:  
  aws:elasticbeanstalk:command:  
    DeploymentPolicy: TrafficSplitting  
  aws:elasticbeanstalk:trafficsplitting:
```

```
NewVersionPercent: "15"  
EvaluationTime: "10"
```

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez [Valeurs recommandées \(p. 660\)](#) pour plus de détails.

## Déploiements bleu/vert avec Elastic Beanstalk

Comme AWS Elastic Beanstalk exécute une mise à niveau sur place lorsque vous mettez à jour vos versions d'application, les utilisateurs peuvent ne plus avoir accès à votre application pendant une brève période. Afin d'éviter cela, effectuez un déploiement bleu/vert. Pour ce faire, déployez la nouvelle version dans un environnement distinct, puis échangez les CNAME des deux environnements afin de rediriger instantanément le trafic vers la nouvelle version.

Un déploiement bleu/vert est également requis si vous souhaitez mettre à jour un environnement vers une version de plateforme non compatible. Pour de plus amples informations, veuillez consulter [the section called "Mises à jour de plateforme" \(p. 496\)](#).

Les déploiements bleu/vert nécessitent que votre environnement s'exécute indépendamment de votre base de données de production, si votre application en utilise une. Si votre environnement inclut une base de données créée par Elastic Beanstalk en votre nom, la base de données et la connexion de l'environnement ne sont pas préservées, sauf si vous effectuez des actions spécifiques. Si vous possédez une base de données que vous souhaitez conserver, utilisez l'une des options de cycle de vie de base de données Elastic Beanstalk. Vous pouvez choisir l'option Retain (Conserver) afin de garder la base de données et l'environnement opérationnels après le découplage de la base de données. Pour de plus amples informations, consultez [Cycle de vie de base de données \(p. 618\)](#) dans le chapitre Configuration des environnements de ce guide.

Pour de plus amples informations sur la configuration de votre application pour la connecter à une instance Amazon RDS qui n'est pas gérée par Elastic Beanstalk, consultez [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#).

Pour effectuer un déploiement bleu/vert

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. [Clonez votre environnement actuel \(p. 461\)](#) ou lancez un nouvel environnement pour exécuter la version de la plateforme souhaitée.
3. [Déployez la nouvelle version d'application \(p. 478\)](#) dans le nouvel environnement.
4. Testez la nouvelle version sur le nouvel environnement.
5. Sur la page de présentation de l'environnement, choisissez Environment actions (Actions d'environnement), puis Swap environment URLs (URL de l'environnement Swap).
6. Dans Environment name (Nom de l'environnement), sélectionnez l'environnement actuel.

## Swap environment URLs

When you swap an environment's URL with another environment's URL, you can deploy versions with no downtime.

**⚠️** Swapping the environment URL will modify the Route 53 DNS configuration, which may take a few minutes to propagate. Your application will continue to run while the changes are propagated.

### Environment details

Environment name:

staging-env

Environment URL:

staging-env.bx7dx222kw.us-east-2.elasticbeanstalk.com

### Select an environment to swap

Environment name:

prod-env (e-2mvwbhpfcs)

Environment URL:

prod-env.bx7dx222kw.us-east-2.elasticbeanstalk.com

7. Choisissez Permuter.

Elastic Beanstalk échange les enregistrements CNAME de l'ancien et du nouvel environnement, en redirigeant le trafic depuis l'ancienne version vers la nouvelle version, et inversement.

Une fois l'opération d'échange effectuée par Elastic Beanstalk, assurez-vous que le nouvel environnement répond lorsque vous essayez de vous connecter à l'URL de l'ancien environnement. Toutefois, avant

de résilier votre ancien environnement, attendez que les modifications DNS se propagent et que vos anciens enregistrements DNS arrivent à expiration. Les serveurs DNS n'effacent pas toujours les anciens enregistrements de leur cache, selon la durée de vie (TTL, time-to-live) que vous avez définie dans vos enregistrements DNS.

## Configuration changes

Lorsque vous modifiez les paramètres d'option de configuration dans la section Configuration de la [console de gestion de l'environnement \(p. 429\)](#), AWS Elastic Beanstalk propage les modifications à toutes les ressources concernées. Ces ressources incluent l'équilibrEUR de charge qui répartit le trafic vers les instances Amazon EC2 exécutant votre application, le groupe Auto Scaling qui gère ces instances et les instances EC2 elles-mêmes.

De nombreux changements de configuration peuvent s'appliquer à un environnement en cours d'exécution sans remplacer les instances existantes. Par exemple, la définition d'une [URL de vérification de l'état \(p. 571\)](#) déclenche une mise à jour de l'environnement pour modifier les paramètres de l'équilibrEUR de charge, mais n'entraîne aucun temps d'arrêt, car les instances qui exécutent votre application continuent à traiter les demandes pendant la propagation de la mise à jour.

Les changements de configuration qui modifient la [configuration du lancement \(p. 681\)](#) ou les [paramètres de VPC \(p. 696\)](#) nécessitent de résilier toutes les instances de votre environnement et de les remplacer. Par exemple, lorsque vous modifiez le type d'instance ou le paramètre de clé SSH pour votre environnement, les instances EC2 doivent être résiliées et remplacées. Elastic Beanstalk fournit plusieurs stratégies qui déterminent la façon dont ce remplacement est effectué.

- Mises à jour propagées – Elastic Beanstalk applique vos modifications de configuration par lots, en conservant un nombre minimal d'instances en cours d'exécution et en servant le trafic à tout moment. Cette approche empêche les temps d'arrêt pendant le processus de mise à jour. Pour plus d'informations, consultez [Mises à jour propagées \(p. 489\)](#).
- Mises à jour immuables – Elastic Beanstalk lance un groupe Auto Scaling temporaire en dehors de votre environnement avec un ensemble distinct d'instances s'exécutant avec la nouvelle configuration. Ensuite, Elastic Beanstalk place ces instances derrière l'équilibrEUR de charge de votre environnement. Les anciennes et les nouvelles instances servent le trafic jusqu'à ce que les nouvelles instances réussissent les contrôles d'intégrité. À ce moment-là, Elastic Beanstalk déplace les nouvelles instances dans le groupe Auto Scaling de votre environnement et résilie le groupe temporaire et les anciennes instances. Pour plus d'informations, consultez [Mises à jour immuables \(p. 493\)](#).
- Désactivé – Elastic Beanstalk ne tente pas d'éviter les temps d'arrêt. Il met fin aux instances existantes de votre environnement et les remplace par de nouvelles instances exécutées avec la nouvelle configuration.

### Warning

Certaines stratégies remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous les [équilibrEURS de rafale Amazon EC2](#) cumulés. Une telle situation se produit dans les cas suivants :

- Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
- Mises à jour immuables
- Déploiements avec mises à jour immuables ou fractionnement du trafic activé

### Types de mise à jour pris en charge

Paramètre de mise à jour propagée	Environnements à charge équilibrée	Environnements d'instance unique	Environnements Windows Server existants†
Désactivé	✓ Oui	✓ Oui	✓ Oui
Propagation en fonction de la santé	✓ Oui	✗ Non	✓ Oui
Propagation en fonction de la durée	✓ Oui	✗ Non	✓ Oui
Immutable	✓ Oui	✓ Oui	✗ Non

† Pour les besoins de cette table, un environnement Windows Server existant est un environnement basé sur une [configuration de plateforme Windows Server](#) qui utilise une version IIS antérieure à IIS 8.5.

#### Rubriques

- [Mises à jour propagées de la configuration de l'environnement Elastic Beanstalk \(p. 489\)](#)
- [Mises à jour immuables de l'environnement \(p. 493\)](#)

## Mises à jour propagées de la configuration de l'environnement Elastic Beanstalk

Lorsqu'une [modification de configuration nécessite de remplacer les instances \(p. 488\)](#), Elastic Beanstalk peut effectuer la mise à jour par lots afin d'éviter les temps d'arrêt pendant que la modification est propagée. Pendant une mise à jour propagée, la capacité est réduite uniquement de la taille d'un seul lot, que vous pouvez configurer. Elastic Beanstalk prend un lot d'instances hors service, le résilie, puis lance un lot avec la nouvelle configuration. Une fois que le nouveau lot commence à traiter les demandes, Elastic Beanstalk passe au lot suivant.

Les lots de mise à jour de configuration peuvent être traités périodiquement (en fonction du temps), avec un délai entre chaque lot, ou sur selon l'intégrité. Pour les mises à jour propagées basées sur le temps, vous pouvez configurer la durée d'attente pour Elastic Beanstalk, après le lancement d'un lot d'instances avant de passer au lot suivant. Ce temps d'interruption permet à votre application d'amorcer et de commencer à traiter les demandes.

Avec des mises à jour propagées basées sur l'état, Elastic Beanstalk attend que des instances dans un lot transmettent des vérifications de l'état avant de passer au lot suivant. L'intégrité d'une instance est déterminée par le système de création de rapports d'intégrité, qui peut être de base ou améliorée. Avec [l'état de base \(p. 834\)](#), un lot est considéré comme sain dès que toutes les instances qu'il contient passent les vérifications de l'état Elastic Load Balancing (ELB).

Avec la [création de rapports d'état amélioré \(p. 837\)](#), toutes les instances dans un lot doivent réussir plusieurs vérifications de l'état consécutives avant qu'Elastic Beanstalk passe au lot suivant. Outre les vérifications de l'état ELB, qui ne vérifient que vos instances, les rapports améliorés sur l'état de santé surveillent les journaux d'application et l'état des autres ressources de votre environnement. Dans un environnement de serveur web à l'intégrité améliorée, toutes les instances doivent réussir 12 vérifications de l'état au cours des deux minutes (18 vérifications au cours des trois minutes pour les environnements de travail). Si une instance échoue à une vérification de l'état, le nombre se réinitialise.

Si un lot ne devient pas sain dans le délai de mise à jour propagée (la valeur par défaut est de 30 minutes), la mise à jour est annulée. L'expiration de la mise à jour propagée est une [option de configuration \(p. 658\)](#) qui est disponible dans l'espace de noms

`aws:autoscaling:updatepolicy:rollingupdate` (p. 492). Si votre application ne réussit pas les vérifications de l'état avec le statut Ok mais qu'elle est stable à un autre niveau, vous pouvez définir l'option `HealthCheckSuccessThreshold` dans l'espace de noms `aws:elasticbeanstalk:healthreporting:system` (p. 706) afin de modifier le niveau auquel Elastic Beanstalk considère une instance comme étant saine.

Si le processus de mise à jour propagée échoue, Elastic Beanstalk commence une autre mise à jour propagée pour restaurer la configuration précédente. Une mise à jour propagée peut échouer en raison de vérifications de l'état ayant échoué, ou si le lancement de nouvelles instances vous conduit à dépasser les quotas de votre compte. Si vous atteignez le quota du nombre d'instances Amazon EC2, par exemple, la mise à jour propagée peut échouer lorsqu'elle tente d'allouer un lot de nouvelles instances. Dans ce cas, la restauration échoue également.

Si la restauration échoue, le processus de mise à jour est interrompu et votre environnement reste défaillant. Les lots non traités continuent à exécuter des instances avec l'ancienne configuration, tandis que les lots qui ont été traités correctement disposent de la nouvelle configuration. Pour réparer un environnement après une restauration qui a échoué, commencez par résoudre le problème sous-jacent qui a provoqué l'échec de la mise à jour, puis lancez une autre mise à jour de l'environnement.

Une autre méthode consiste à déployer la nouvelle version de votre application dans un environnement différent, puis à effectuer une permutation CNAME pour rediriger le trafic sans interruption. Pour de plus amples informations, veuillez consulter [Déploiements bleu/vert avec Elastic Beanstalk \(p. 486\)](#).

## Mises à jour propagées et déploiements propagés

Des mises à jour propagées se produisent lorsque vous modifiez des paramètres qui ont besoin que de nouvelles instances Amazon EC2 soient mises en service pour votre environnement. Cela inclut des modifications apportées à la configuration du groupe Auto Scaling, telles que le type d'instance et des paramètres de paire de clés et les modifications apportées aux paramètres VPC. Dans une mise à jour propagée, chaque lot d'instances est résilié avant l'allocation d'un nouveau lot pour le remplacer.

Les [déploiements propagés \(p. 479\)](#) se produisent lorsque vous déployez votre application, et peuvent généralement être effectués sans remplacer d'instances dans votre environnement. Elastic Beanstalk met chaque lot hors service, déploie la nouvelle version de l'application, puis le remet en service.

L'exception à cela est si vous modifiez les paramètres qui ont besoin de remplacement d'instance en même temps vous déployez une nouvelle version de l'application. Par exemple, si vous modifiez les paramètres de [nom de clé \(p. 681\)](#) dans un [fichier de configuration \(p. 737\)](#) dans votre bundle source et que vous les déployez dans votre environnement, vous déclenchez une mise à jour propagée. Au lieu de déployer votre nouvelle version de l'application dans chaque lot d'instances existantes, un nouveau lot d'instances est fourni avec la nouvelle configuration. Dans ce cas, aucun déploiement distinct ne se produit, car les nouvelles instances sont installées avec la nouvelle version de l'application.

Chaque fois que de nouvelles instances sont déployées dans le cadre d'une mise à jour de l'environnement, une phase de déploiement est effectuée, au cours de laquelle le code source de votre application est déployé dans les nouvelles instances et tous les paramètres de configuration modifiant le système d'exploitation ou le logiciel sur les instances sont appliqués. Les [paramètres de vérification de l'état du déploiement \(p. 480\)](#) (Ignorer la vérification de l'état, Seuil de bonne santé et Délai de commande) s'appliquent également aux mises à jour propagées basées sur l'intégrité et aux mises à jour immuables lors de la phase de déploiement.

## Configuration des mises à jour propagées

Vous pouvez activer et configurer les mises à jour propagées dans la console Elastic Beanstalk.

Pour activer des mises à jour propagées

- Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Rolling updates and deployments (Propagation des mises à jour et déploiements), choisissez Edit (Modifier).
5. Dans la section Mises à jour de la configuration, pour Propagation du type de mises à jour, sélectionnez l'une des options Propagation.

The screenshot shows the 'Configuration updates' section of the AWS Elastic Beanstalk configuration. It includes fields for 'Rolling update type' (set to 'Rolling based on Health'), 'Batch size' (set to 1), 'Minimum capacity' (set to 1), and 'Pause time' (set to 'hh:mm:ss'). A note below the pause time field states: 'Pause the update for up to an hour between each batch.'

6. Choisissez Taille de lot, Capacité minimale et les paramètres Temps d'interruption.
7. Choisissez Apply.

La section Mises à jour de la configuration de la page Propagation des mises à jour et déploiements propose les options suivantes pour la propagation des mises à jour :

- Rolling update type (Propagation du type de mise à jour) – Elastic Beanstalk attend la fin de la mise à jour d'un lot d'instances avant de passer au lot suivant, afin de permettre à ces instances de finaliser l'action d'amorçage et de commencer à traiter le trafic. Choisissez parmi les options suivantes :
- Rolling based on Health (Propagation en fonction de la santé) – Patientez jusqu'à ce que les instances dans le lot actuel soient saines avant de mettre des instances en service et de démarrer le lot suivant.
- Rolling based on Time(Propagation en fonction de la durée) – Spécifiez un laps de temps d'attente entre le lancement de nouvelles instances et leur mise en service avant de démarrer le lot suivant.

- Immutable (Immutable) – Appliquez la modification de la configuration à un nouveau groupe d'instances en effectuant une [mise à jour immuable \(p. 493\)](#).
- Batch size (Taille de lot) – Le nombre d'instances à remplacer dans chaque lot, entre **1** et **10000**. Par défaut, cette valeur représente un tiers de la taille minimale du groupe Auto Scaling, arrondi au nombre entier.
- Minimum capacity (Capacité minimum) – Le nombre minimum d'instances à maintenir en cours d'exécution pendant que d'autres instances sont mises à jour, entre **0** et **9999**. La valeur par défaut est soit la taille minimale du groupe Auto Scaling ou un niveau en dessous de la taille maximale du groupe Auto Scaling, le chiffre le plus bas prévalant.
- Pause time (Temps d'interruption) (basé sur le temps uniquement) – La durée d'attente après qu'un lot est mis à jour avant de passer au lot suivant, pour permettre à votre application de commencer à recevoir du trafic. Entre 0 seconde et 1 heure.

## L'espace de noms aws:autoscaling:updatepolicy:rollingupdate

Vous pouvez également utiliser les [options de configuration \(p. 658\)](#) dans l'espace de noms `aws:autoscaling:updatepolicy:rollingupdate` ([p. 690](#)) pour configurer des mises à jour propagées.

Utilisez l'option `RollingUpdateEnabled` pour activer les mises à jour propagées, et `RollingUpdateType` pour choisir le type de mise à jour. Les valeurs suivantes sont prises en charge pour `RollingUpdateType` :

- Health – Patientez jusqu'à ce que les instances dans le lot actuel soient saines avant de mettre des instances en service et de démarrer le lot suivant.
- Time – Spécifie un laps de temps d'attente entre le lancement de nouvelles instances et leur mise en service avant de démarrer le lot suivant.
- Immutable – Appliquez la modification de la configuration à un nouveau groupe d'instances en effectuant une [mise à jour immuable \(p. 493\)](#).

Lorsque vous activez les mises à jours propagées, définissez les options `MaxBatchSize` et `MinInstancesInService` afin de configurer la taille de chaque lot. Pour les mises à jour propagées basées sur l'intégrité et basées sur le temps, vous pouvez également configurer un `PauseTime` et `Timeout`, respectivement.

Par exemple, afin de lancer jusqu'à cinq instances à la fois, tout en conservant au moins deux instances en service, puis attendre cinq minutes et 30 secondes entre les lots, spécifiez les options et les valeurs suivantes.

### Example .ebextensions/timebased.config

```
option_settings:  
  aws:autoscaling:updatepolicy:rollingupdate:  
    RollingUpdateEnabled: true  
    MaxBatchSize: 5  
    MinInstancesInService: 2  
    RollingUpdateType: Time  
    PauseTime: PT5M30S
```

Pour activer les mises à jour propagées basées sur l'intégrité, avec une expiration à 45 minutes pour chaque lot, spécifiez les valeurs et les options suivantes.

### Example .ebextensions/healthbased.config

```
option_settings:
```

```
aws:autoscaling:updatepolicy:rollingupdate:  
  RollingUpdateEnabled: true  
  MaxBatchSize: 5  
  MinInstancesInService: 2  
  RollingUpdateType: Health  
  Timeout: PT45M
```

Les valeurs `Timeout` et `PauseTime` doivent être spécifiées au format de [durée ISO8601](#) : `PT#H#M#S`, où chaque # correspond au nombre d'heures, de minutes ou de secondes, respectivement.

L'interface de ligne de commande EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Pour de plus amples informations, veuillez consulter [Valeurs recommandées](#) (p. 660).

## Mises à jour immuables de l'environnement

Les mises à jour de l'environnement immuables sont une alternative aux [mises à jour propagées](#) (p. 489). Les mises à jour de l'environnement immuables garantissent que les modifications de configuration qui nécessitent le remplacement d'instances sont appliquées efficacement et en toute sécurité. En cas de défaillance d'une mise à jour de l'environnement immuable, le processus de restauration nécessite uniquement l'arrêt d'un groupe Auto Scaling. Un échec de mise à jour propagée, en revanche, nécessite de procéder à une mise à jour propagée supplémentaire pour restaurer les modifications.

Pour effectuer une mise à jour de l'environnement immuable, Elastic Beanstalk crée un second groupe Auto Scaling temporaire derrière l'équilibreur de charge de votre environnement pour contenir les nouvelles instances. Tout d'abord, Elastic Beanstalk lance une instance unique avec la nouvelle configuration dans le nouveau groupe. Cette instance traite le trafic en même temps que toutes les instances dans le groupe Auto Scaling d'origine qui exécutent la configuration précédente.

Lorsque la première instance passe les vérifications de l'état, Elastic Beanstalk lance des instances supplémentaires avec la nouvelle configuration, correspondant au nombre d'instances en cours d'exécution dans le groupe Auto Scaling d'origine. Lorsque toutes les nouvelles instances réussissent les vérifications de l'état, Elastic Beanstalk les transfère au groupe Auto Scaling d'origine et résilie le groupe Auto Scaling temporaire et les anciennes instances.

### Note

Pendant une mise à jour d'environnement immuable, la capacité de votre environnement double pendant une brève durée lorsque les instances dans le nouveau groupe Auto Scaling commencent à traiter les demandes et avant que les instances du groupe Auto Scaling d'origine soient résiliées.

Si votre environnement a de nombreuses instances, ou des instances avec un faible [quota d'instances à la demande](#), assurez-vous d'avoir suffisamment de capacité pour effectuer une mise à jour de l'environnement immuable. Si vous êtes proche du quota, pensez à utiliser des mises à jour propagées à la place.

Les mises à jour immuables nécessitent des [rapports sur l'état de santé améliorés](#) (p. 837) afin d'évaluer l'intégrité de votre environnement pendant la mise à jour. Les rapports sur l'état de santé améliorés associent des vérifications de l'état standard de l'équilibreur de charge avec la surveillance de l'instance afin de garantir que les instances en cours d'exécution avec la nouvelle configuration [traitent les demandes avec succès](#) (p. 840).

Vous pouvez également utiliser des mises à jour immuables pour déployer de nouvelles versions de votre application, comme une alternative aux déploiements propagés. Lorsque vous [configurez Elastic Beanstalk pour utiliser des mises à jour immuables pour des déploiements d'applications](#) (p. 479), il remplace toutes les instances de votre environnement chaque fois que vous déployez une nouvelle version de votre application. Si un déploiement d'application immuable échoue, Elastic Beanstalk rétablit immédiatement les modifications en résiliant le nouveau groupe Auto Scaling. Cela peut empêcher des déploiements de flottes partiels qui peuvent se produire lorsqu'un déploiement propagé échoue après que certains lots sont déjà terminés.

### Warning

Certaines stratégies remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous les [équilibres de rafale Amazon EC2](#) cumulés. Une telle situation se produit dans les cas suivants :

- Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
- Mises à jour immuables
- Déploiements avec mises à jour immuables ou fractionnement du trafic activé

Si une mise à jour immuable échoue, les nouvelles instances téléchargent [les journaux groupés \(p. 884\)](#) vers Amazon S3 avant qu'Elastic Beanstalk ne les résilie. Elastic Beanstalk laisse les journaux d'une mise à jour immuable ayant échoué dans Amazon S3 pendant une heure avant de les supprimer, au lieu des 15 minutes standard pour les journaux groupés et de processus.

### Note

Si vous utilisez des mises à jour immuables pour des déploiements de version d'application, mais pas pour la configuration, vous pouvez rencontrer une erreur si vous tentez de déployer une version de l'application qui contient des modifications de configuration qui déclenchaient normalement une mise à jour propagée (par exemple, des configurations modifiant le type d'instance). Pour éviter ce problème, modifiez la configuration dans une mise à jour séparée, ou configurez les mises à jour immuables à la fois pour les changements de configuration et pour les déploiements.

Vous ne pouvez pas effectuer une mise à jour immuable de concert avec les changements de configuration de ressource. Par exemple, vous ne pouvez pas modifier [les paramètres qui ont besoin d'un remplacement d'instance \(p. 488\)](#) tout en mettant également à jour d'autres paramètres, ni exécuter un déploiement immuable avec des fichiers de configuration qui modifient les paramètres de configuration ou les ressources supplémentaires dans votre code source. Si vous essayez de modifier les paramètres de ressource (par exemple, les paramètres d'équilibrage de charge) et de procéder simultanément à une mise à jour immuable, Elastic Beanstalk renvoie une erreur.

Si vos modifications de configuration de ressource ne dépendent pas de la modification de votre code source ou de la configuration d'instance, exécutez-les dans deux mises à jour. Si elles sont dépendantes, effectuez plutôt un [déploiement bleu/vert \(p. 486\)](#).

## Configuration de mises à jour immuables

Vous pouvez activer et configurer des mises à jour immuables dans la console Elastic Beanstalk.

Pour activer des mises à jour immuables (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Rolling updates and deployments (Propagation des mises à jour et déploiements), choisissez Edit (Modifier).
5. Dans la section Mises à jour de la configuration, définissez Rolling update type (Type des mises à jour propagées) sur Immutable.

The screenshot shows the 'Configuration updates' section of the AWS Elastic Beanstalk configuration interface. It includes the following settings:

- Rolling update type:** Immutable
- Batch size:** 1
- Minimum capacity:** 1
- Pause time:** hh:mm:ss

Below each setting is a brief description of its purpose.

6. Choisissez Apply.

## L'espace de noms aws:autoscaling:updatepolicy:rollingupdate

Vous pouvez également utiliser les options dans l'espace de noms `aws:autoscaling:updatepolicy:rollingupdate` pour configurer des mises à jour immuables. L'exemple suivant de [fichier de configuration \(p. 737\)](#) active les mises à jour immuables pour les changements de configuration.

Example `.ebextensions/immutable-updates.config`

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Immutable
```

L'exemple suivant active les mises à jour immuables pour les changements de configuration et les déploiements.

Example `.ebextensions/immutable-all.config`

```
option_settings:
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Immutable
  aws:elasticbeanstalk:command:
    DeploymentPolicy: Immutable
```

L'interface de ligne de commande EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Pour de plus amples informations, veuillez consulter [Valeurs recommandées \(p. 660\)](#).

## Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk

Elastic Beanstalk publie régulièrement de nouvelles versions de plateforme pour mettre à jour toutes les [plateformes \(p. 31\)](#) basées sur Linux et sur Windows Server. Les nouvelles versions de plateforme fournissent des mises à jour des composants logiciels existants et prennent en charge de nouvelles fonctionnalités ainsi que des options de configuration. Pour en savoir plus sur les plateformes et les versions de plateforme, veuillez consulter [Glossaire des plateformes Elastic Beanstalk \(p. 25\)](#).

Vous pouvez utiliser la console Elastic Beanstalk ou l'interface de ligne de commande EB pour mettre à jour la version de la plateforme de votre environnement. En fonction de la version de plateforme que vous souhaitez mettre à jour, Elastic Beanstalk recommande l'une des deux méthodes pour effectuer des mises à jour de la plateforme.

- [Méthode 1 – Mettre à jour la version de la plateforme de votre environnement \(p. 498\)](#). Nous vous recommandons d'utiliser cette méthode lorsque vous mettez à jour vers la dernière version de la plateforme dans une branche de plateforme : avec le même environnement d'exécution, le même serveur web, le même serveur d'application et le même système d'exploitation, et sans modifier la version de la plateforme principale. Il s'agit de la mise à jour de plateforme la plus courante et la plus régulière.
- [Méthode 2 – Effectuer un déploiement bleu/vert \(p. 500\)](#). Nous vous recommandons d'utiliser cette méthode lorsque vous effectuez une mise à jour vers une version de plateforme dans une autre branche de plateforme : avec un environnement d'exécution, un serveur web, un serveur d'applications ou un système d'exploitation différents, ou vers une autre version majeure de plateforme. Il s'agit d'une bonne approche lorsque vous voulez tirer parti des nouvelles fonctionnalités d'exécution ou des dernières fonctionnalités Elastic Beanstalk, ou encore lorsque vous voulez quitter une branche de plateforme obsolète ou hors service.

La [migration à partir d'une version de plateforme héritée \(p. 507\)](#) nécessite un déploiement bleu/vert, car ces versions de plateforme sont incompatibles avec les versions actuellement prises en charge.

La [migration d'une application Linux vers Amazon Linux 2 \(p. 508\)](#) nécessite un déploiement bleu/vert, car les versions de la plateforme Amazon Linux 2 sont incompatibles avec les versions précédentes de la plateforme AMI Amazon Linux.

Pour plus d'informations sur le choix de la meilleure méthode de mise à jour de la plateforme, développez la section pour la plateforme de votre environnement.

### Docker

Utilisez la [méthode 1 \(p. 498\)](#) pour effectuer des mises à jour de la plateforme.

### Docker multi-conteneurs

Utilisez la [méthode 1 \(p. 498\)](#) pour effectuer des mises à jour de la plateforme.

### Docker préconfiguré

Considérons les cas suivants :

- Si vous migrez votre application vers une autre plateforme, par exemple de Go 1.4 (Docker) vers Go 1.11 ou de Python 3.4 (Docker) vers Python 3.6, utilisez la [méthode 2 \(p. 500\)](#).
- Si vous migrez votre application vers une autre version de conteneur Docker, par exemple de Glassfish 4.1 (Docker) vers Glassfish 5.0 (Docker), utilisez la [méthode 2 \(p. 500\)](#).
- Si vous mettez à jour vers une version plus récente de la plateforme sans aucune modification de version majeure ou de conteneur, utilisez la [méthode 1 \(p. 498\)](#).

## Go

Utilisez la [méthode 1 \(p. 498\)](#) pour effectuer des mises à jour de la plateforme.

## Java SE

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version d'exécution Java, par exemple de Java 7 vers Java 8, utilisez la [méthode 2 \(p. 500\)](#).
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modification de la version d'exécution, utilisez la [méthode 1 \(p. 498\)](#).

## Java avec Tomcat

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version d'environnement d'exécution Java ou version de serveur d'application Tomcat, par exemple de Java 7 avec Tomcat 7 vers Java 8 avec Tomcat 8.5, utilisez la [méthode 2 \(p. 500\)](#).
- Si vous migrez votre application sur les versions majeures de Java avec la plateforme Tomcat (v1.x.x, v2.x.x et v3.x.x), utilisez la [méthode 2 \(p. 500\)](#).
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modifier la version d'exécution, la version du serveur d'application ou la version majeure, utilisez la [méthode 1 \(p. 498\)](#).

## .NET sur serveur Windows avec IIS

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version du système d'exploitation Windows, par exemple de Windows Server 2008 R2 vers Windows Server 2016, utilisez la [méthode 2 \(p. 500\)](#).
- Si vous migrez votre application à travers les principales versions de plateforme Windows Server, veuillez consulter [Migration à partir de versions majeures antérieures de la plateforme Windows Server \(p. 199\)](#), et utilisez la [méthode 2 \(p. 500\)](#).
- Si votre application est en cours d'exécution sur une plateforme Windows Server V2.x.x et que vous mettez à jour vers une version plus récente de la plateforme, utilisez la [méthode 1 \(p. 498\)](#).

### Note

Les [versions de plateforme Windows Server](#) antérieures à la version v2 ne sont pas sémantiquement versionnées. Vous ne pouvez lancer que la dernière version de ces versions de plateforme majeures Windows Server et vous ne pouvez pas restaurer après une mise à niveau.

## Node.js

Utilisez la [méthode 2 \(p. 500\)](#) pour effectuer des mises à jour de la plateforme.

## PHP

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version d'exécution PHP, par exemple de PHP 5.6 vers PHP 7.2, utilisez la [méthode 2 \(p. 500\)](#).
- Si vous migrez votre application sur les versions majeures de la plateforme PHP (v1.x.x et v2.x.x), utilisez la [méthode 2 \(p. 500\)](#).
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modifier la version d'exécution ou la version majeure, utilisez la [méthode 1 \(p. 498\)](#).

## Python

Considérons les cas suivants :

- Si vous migrez votre application vers un autre environnement d'exécution Python, par exemple de Python 2.7 vers Python 3.6, utilisez la [méthode 2 \(p. 500\)](#).
- Si vous migrez votre application sur les versions majeures de la plateforme Python (v1.x.x et v2.x.x), utilisez la [méthode 2 \(p. 500\)](#).
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modifier la version d'exécution ou la version majeure, utilisez la [méthode 1 \(p. 498\)](#).

## Ruby

Considérons les cas suivants :

- Si vous migrez votre application vers une autre version d'exécution de Ruby ou une version de serveur d'application, par exemple de Ruby 2.3 avec Puma vers Ruby 2.6 avec Puma, utilisez la [méthode 2 \(p. 500\)](#).
- Si vous migrez votre application sur les versions majeures de la plateforme Ruby (v1.x.x et v2.x.x), utilisez la [méthode 2 \(p. 500\)](#).
- Si vous mettez à jour vers une version plus récente de la plateforme, sans modifier la version d'exécution, la version du serveur d'application ou la version majeure, utilisez la [méthode 1 \(p. 498\)](#).

## Méthode 1 – Mettre à jour la version de la plateforme de votre environnement

Utilisez cette méthode pour mettre à jour vers la dernière branche de plateforme de votre environnement. Si vous avez préalablement créé un environnement à l'aide d'une version de plateforme antérieure ou mis à niveau votre environnement à partir d'une version plus ancienne, vous pouvez également utiliser cette méthode pour revenir à une version de plateforme précédente, à condition qu'elle se trouve dans la même branche de plateforme.

Pour mettre à jour la version de la plateforme de votre environnement

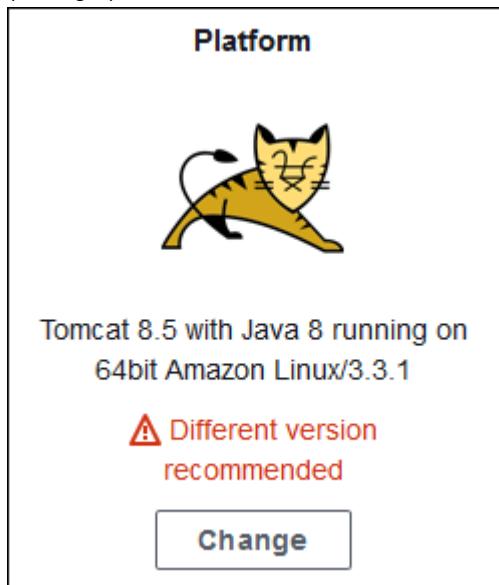
1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans la page de présentation de l'environnement, sous Platform (Plateforme), choisissez Change (Changer).



4. Dans la boîte de dialogue Update platform version (Mettre à jour la version de la plateforme) sélectionnez une version de la plate-forme. La version de la plateforme la plus récente (recommandée) de la branche est sélectionnée automatiquement. Vous pouvez mettre à jour vers n'importe quelle version que vous avez utilisée dans le passé.

The screenshot shows the 'Update platform version' section of the AWS Elastic Beanstalk console. It includes a warning message about replacing instances during the update, the current platform version (3.3.1), and a dropdown menu for selecting the new platform version (3.3.2 (Recommended)).

**Availability warning**  
This operation replaces your instances; your application is unavailable during the update. If you have one instance in service during the update, enable rolling updates. Another option is to clone the environment, which creates a newer version of the platform, and then swap the CNAME of the cloned environment when you are ready to deploy the clone. Learn more at [Updating AWS Elastic Beanstalk Environment](#), [Rolling Updates](#) and [Deploying Version with Zero Downtime](#).

Platform branch  
Tomcat 8.5 with Java 8 running on 64bit Amazon Linux

Current platform version  
3.3.1

New platform version  
3.3.2 (Recommended) ▾

5. Choisissez Enregistrer.

Pour davantage simplifier les mises à jour des plateformes, Elastic Beanstalk peut les gérer pour vous. Vous pouvez configurer votre environnement pour appliquer automatiquement des mises à jour de versions mineures et de correctifs pendant un créneau de maintenance hebdomadaire configurable. Elastic Beanstalk applique des mises à jour gérées sans interruption ni réduction de capacité et annule la mise à jour immédiatement si les instances exécutant votre application sur la nouvelle version échouent aux vérifications de l'état. Pour plus d'informations, consultez [Mises à jour gérées de la plateforme \(p. 501\)](#).

## Méthode 2 – Effectuer un déploiement bleu/vert

Utilisez cette méthode pour effectuer une mise à jour vers une autre branche de plateforme : avec un environnement d'exécution, un serveur web, un serveur d'applications ou un système d'exploitation différents, ou vers une autre version principale de plateforme. Cela est généralement nécessaire lorsque vous souhaitez tirer parti des nouvelles fonctionnalités d'exécution ou des dernières fonctionnalités Elastic Beanstalk. Cela est également requis lorsque vous migrez hors d'une branche de plateforme obsolète ou hors service.

Lorsque vous migrez sur plusieurs versions de plate-forme majeures ou vers des versions de plate-forme avec des mises à jour de composants principaux composant, il y a plus de chances que votre application ou certains aspects de celle-ci peuvent ne pas fonctionner comme prévu sur la nouvelle version de la plateforme, et peuvent nécessiter des changements.

Avant de réaliser la migration, mettez à jour votre machine de développement local vers les dernières versions d'exécution et d'autres composants de la plateforme que vous prévoyez de migrer. Vérifiez que votre application fonctionne comme prévu et apportez toutes les modifications et les correctifs de code nécessaire. Ensuite, utilisez la procédure de bonnes pratiques suivantes pour migrer votre environnement en toute sécurité vers la nouvelle version de la plateforme.

Pour migrer votre environnement vers une version de plateforme avec des mises à jour majeures

1. [Créez un environnement \(p. 439\)](#) en utilisant la nouvelle version de plateforme, puis déployez-y votre code d'application. Le nouvel environnement doit être dans l'application Elastic Beanstalk qui contient l'environnement que vous migrez. N'arrêtez pas encore l'environnement existant.
2. Utilisez le nouvel environnement pour migrer votre code d'application. En particulier :
  - Trouvez et corrigez les problèmes de compatibilité de l'application que vous n'avez pas pu détecter au cours de la phase de développement.
  - Assurez-vous que toutes les personnalisations que votre application effectue en utilisant les [fichiers de configuration \(p. 737\)](#) fonctionnent correctement dans le nouvel environnement. Celles-ci peuvent inclure les paramètres d'options, les packages installés supplémentaires, les stratégies de sécurité personnalisées, et les fichiers de configuration ou de script installés sur les instances de l'environnement.
  - Si votre application utilise une Amazon Machine Image (AMI) personnalisée, créez une AMI personnalisée basée sur l'AMI de la nouvelle version de la plateforme. Pour en savoir plus, consultez la section [Utilisation d'une Amazon Machine Image \(AMI\) personnalisée \(p. 787\)](#). Plus précisément, cette action est obligatoire si votre application utilise la plateforme Windows Server avec une AMI personnalisée, et que vous migrez vers une version de la plateforme Windows Server V2. Dans ce cas, veuillez consulter également [Migration à partir de versions majeures antérieures de la plateforme Windows Server \(p. 199\)](#).

Itérez les tests et déployez vos correctifs jusqu'à ce que vous soyez satisfait de l'application sur le nouvel environnement.

3. Activez le nouvel environnement en tant qu'environnement de production en échangeant son CNAME avec le CNAME de l'environnement de production existant. Pour plus d'informations, consultez [Déploiements bleu/vert avec Elastic Beanstalk \(p. 486\)](#).
4. Lorsque vous êtes satisfait de l'état de votre nouvel environnement de production, arrêtez l'ancien environnement. Pour plus d'informations, consultez [Arrêt d'un environnement Elastic Beanstalk \(p. 464\)](#).

## Mises à jour gérées de la plateforme

AWS Elastic Beanstalk publie régulièrement des [mises à jour de la plateforme \(p. 496\)](#) pour fournir des correctifs, des mises à jour logicielles et de nouvelles fonctionnalités. Grâce aux mises à jour gérées de la plateforme, vous pouvez configurer votre environnement pour effectuer automatiquement les mises à niveau vers la dernière version d'une plateforme au cours d'un [créneau de maintenance \(p. 504\)](#) planifié. Votre application reste en service pendant le processus de mise à jour, sans réduction de capacité. Les mises à jour gérées sont disponibles sur les environnements à l'instance unique et avec équilibrage de charge.

### Note

Cette fonction n'est pas disponible dans les [versions de la plateforme Windows Server](#) antérieures à la version 2 (v2).

Vous pouvez configurer votre environnement pour appliquer automatiquement les [mises à jour de version corrective \(p. 504\)](#) ou les mises à jour des deux versions : mineure et corrective. Les mises à jour gérées

de la plateforme ne prennent pas en charge les mises à jour entre les branches de la plateforme (mises à jour vers différentes versions majeures de composants de la plateforme tels que le système d'exploitation, l'exécution ou les composants Elastic Beanstalk), car elles peuvent introduire des modifications non rétrocompatibles.

Vous pouvez également configurer Elastic Beanstalk pour remplacer toutes les instances de votre environnement pendant la fenêtre de maintenance, même si aucune mise à jour de la plateforme n'est disponible. Le remplacement de toutes les instances de votre environnement est utile si votre application est confrontée à des bogues ou à des problèmes de mémoire lorsqu'elle est exécutée pendant une longue période.

Sur les environnements créés le 25 novembre 2019 ou ultérieurement à l'aide de la console Elastic Beanstalk, les mises à jour gérées sont activées par défaut dans la mesure du possible. Les mises à jour gérées nécessitent que l'[état amélioré \(p. 837\)](#) soit activé. L'état amélioré est activé par défaut lorsque vous sélectionnez un des [paramètres prédéfinis de configuration \(p. 447\)](#), et désactivé lorsque vous sélectionnez Configuration personnalisée. La console ne peut pas activer les mises à jour gérées pour les anciennes versions de plate-forme qui ne prennent pas en charge l'état amélioré, ou lorsque l'état amélioré est désactivé. Lorsque la console active les mises à jour gérées pour un nouvel environnement, la fenêtre de mise à jour hebdomadaire est définie sur un jour aléatoire de la semaine à un moment aléatoire. Niveau de mise à jour est défini sur Mineures et correctifs et Remplacement d'instance est désactivé. Vous pouvez désactiver ou reconfigurer les mises à jour gérées avant l'étape finale de création de l'environnement.

Pour un environnement existant, utilisez la console Elastic Beanstalk à tout moment pour configurer les mises à jour des plateformes gérées.

#### Pour configurer les mises à jour des plates-formes gérées

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie Managed updates (Mises à jour gérées), choisissez Edit (Modifier).
5. Désactivez ou activez Mises à jour gérées.
6. Si les mises à jour gérées sont activées, sélectionnez une fenêtre de maintenance, puis un niveau de mise à jour.
7. (Facultatif) Sélectionnez Remplacement de l'instance pour activer le remplacement d'instance hebdomadaire.

The screenshot shows the 'Modify managed updates' configuration page in the AWS Elastic Beanstalk console. The navigation path is: Elastic Beanstalk > Environments > GettingStartedApp-env > Configuration. The main section is titled 'Managed platform updates'. It explains that managed platform updates apply platform updates automatically during a weekly maintenance window while the application stays available. Under 'Managed updates', the 'Enabled' checkbox is checked. The 'Weekly update window' is set to Tuesday at 12:00 UTC. A note states that any available managed updates will run between Tuesday, 4:00 AM and Tuesday, 6:00 AM (-0800 GMT). The 'Update level' dropdown is set to 'Minor and patch'. Under 'Instance replacement', it says that if enabled, an instance replacement will be scheduled if no other updates are available, with an 'Enabled' checkbox. At the bottom right, there is a 'Cancel' button.

8. Choisissez Apply.

Les mises à jour gérées de la plateforme utilisent les [rapports sur l'état de santé améliorés \(p. 837\)](#) afin de déterminer si l'état de votre application est suffisamment satisfaisant pour estimer que la mise à jour de la plateforme a réussi. Pour obtenir des instructions, consultez [Activation des rapports améliorés sur l'état Elastic Beanstalk \(p. 845\)](#).

#### Sections

- [Autorisations requises pour effectuer des mises à jour gérées de la plateforme \(p. 504\)](#)
- [Fenêtre de maintenance des mises à jour gérées \(p. 504\)](#)
- [Mises à jour des versions mineures et correctives \(p. 504\)](#)
- [Mises à jour immuables de l'environnement \(p. 505\)](#)
- [Gestion des mises à jour gérées \(p. 505\)](#)
- [Espaces de noms avec options d'action gérée \(p. 506\)](#)

## Autorisations requises pour effectuer des mises à jour gérées de la plateforme

Elastic Beanstalk a besoin d'autorisations pour lancer une mise à jour de la plateforme en votre nom. Pour obtenir ces autorisations, Elastic Beanstalk assume le rôle de service de mises à jour gérées. Lorsque vous utilisez le [rôle de service \(p. 926\)](#) par défaut pour votre environnement, la console Elastic Beanstalk l'utilise également comme rôle de service de mises à jour gérées. La console affecte la stratégie [AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy \(p. 934\)](#) gérée à votre rôle de service. Cette stratégie dispose de toutes les autorisations dont Elastic Beanstalk a besoin pour effectuer des mises à jour de la plateforme gérée.

Pour de plus amples informations sur les autres méthodes de définition du rôle de service de mises à jour gérées, veuillez consulter [the section called “Rôles de service” \(p. 926\)](#).

### Note

Si vous utilisez des [fichiers de configuration \(p. 737\)](#) pour étendre votre environnement afin d'y inclure des ressources supplémentaires, vous pouvez avoir besoin d'ajouter des autorisations au rôle de service des mises à jour gérées de votre environnement. En général, vous devez ajouter des autorisations lorsque vous faites référence au nom de ces ressources dans d'autres sections ou fichiers.

En cas d'échec d'une mise à jour, vous pouvez trouver la raison de l'échec sur la page des [mises à jour gérées \(p. 505\)](#).

## Fenêtre de maintenance des mises à jour gérées

Lorsqu'AWS lance une nouvelle version de configuration de la plateforme de votre environnement, Elastic Beanstalk planifie une mise à jour gérée de la plateforme au cours de la prochaine fenêtre de maintenance hebdomadaire. Une fenêtre de maintenance dure deux heures. Elastic Beanstalk démarre une mise à jour planifiée pendant la fenêtre de maintenance. La mise à jour ne se termine avant la fin de la fenêtre.

### Note

Dans la plupart des cas, Elastic Beanstalk planifie la mise à jour gérée de telle sorte qu'elle se produise pendant la fenêtre de maintenance hebdomadaire suivante. Le système considère différents aspects de sécurité et de disponibilité des services lors de la planification des mises à jour gérées. Dans de rares cas, une mise à jour peut ne pas être planifiée pour la première fenêtre de maintenance à venir. Si cela se produit, le système tente à nouveau durant la fenêtre de maintenance suivante. Pour procéder manuellement à une mise à jour gérée, choisissez Appliquer maintenant comme expliqué dans la section [Gestion des mises à jour gérées \(p. 505\)](#) sur cette page.

## Mises à jour des versions mineures et correctives

Vous pouvez autoriser les mises à jour gérées de la plateforme à appliquer des mises à jour de version corrective uniquement, ou des mises à jour des deux versions : mineure et corrective. Les mises à jour de version corrective incluent des correctifs de bogues et des améliorations des performances. Elles peuvent aussi comporter des modifications mineures de configuration pour les logiciels sur les instances, les scripts et les options de configuration. Les mises à jour mineures de version prennent en charge les nouvelles fonctions d'Elastic Beanstalk. Dans le cadre des mises à jour gérées de la plateforme, vous ne pouvez pas appliquer de mises à jour de version majeure, lesquelles peuvent entraîner des modifications non rétrocompatibles.

Dans le numéro de version d'une plateforme, le deuxième numéro correspond à la mise à jour de version mineure, et le troisième numéro correspond à la version corrective. Par exemple, une plateforme version 2.0.7 a une version mineure de 0 et une version corrective de 7.

## Mises à jour immuables de l'environnement

Les mises à jour gérées de la plateforme exécutent des [mises à jour immuables de l'environnement \(p. 493\)](#) pour mettre à niveau votre environnement vers une nouvelle version de la plateforme. Les mises à jour immuables mettent à jour votre environnement sans suspendre aucune instance ni modifier votre environnement, avant de vérifier que les instances qui exécutent la nouvelle version réussissent les vérifications de l'état.

Dans une mise à jour immuable, Elastic Beanstalk déploie autant d'instances qu'il y en a en cours d'exécution avec la nouvelle version de la plateforme. Les nouvelles instances commencent à accepter des demandes en même temps que celles qui exécutent l'ancienne version. Si le nouvel ensemble d'instances réussit toutes les vérifications de l'état, Elastic Beanstalk résilie l'ancien ensemble d'instances, en ne laissant que les instances associées à la nouvelle version.

Les mises à jour gérées de la plateforme effectuent toujours des mises à jour immuables, même lorsque vous les appliquez en dehors de la fenêtre de maintenance. Si vous modifiez la version de la plateforme dans le tableau de bord, Elastic Beanstalk applique la stratégie de mise à jour que vous avez choisie pour les mises à jour de configuration.

### Warning

Certaines stratégies remplacent toutes les instances pendant le déploiement ou la mise à jour. Cela entraîne la perte de tous les [équilibres de rafale Amazon EC2](#) cumulés. Une telle situation se produit dans les cas suivants :

- Mises à jour de la plate-forme gérée avec le remplacement d'instance activé
- Mises à jour immuables
- Déploiements avec mises à jour immuables ou fractionnement du trafic activé

## Gestion des mises à jour gérées

La console Elastic Beanstalk présente des informations détaillées sur les mises à jour gérées sur la page Présentation des mises à jour gérées.

Pour afficher les informations relatives aux mises à jour gérées (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Managed updates (Mises à jour gérées).

La section Présentation des mises à jour gérées fournit des informations sur les mises à jour gérées planifiées et en attente. La section Historique répertorie les mises à jour ayant réussi et les tentatives ayant échoué.

Vous pouvez choisir d'appliquer une mise à jour planifiée immédiatement au lieu d'attendre la fenêtre de maintenance.

Pour appliquer une mise à jour gérée de la plateforme immédiatement (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Managed updates (Mises à jour gérées).
4. Choisissez Appliquer maintenant.
5. Vérifiez les détails de la mise à jour, puis choisissez Apply (Appliquer).

Lorsque vous appliquez une mise à jour gérée de la plateforme en dehors de la fenêtre de maintenance, Elastic Beanstalk effectue une mise à jour immuable. Si vous mettez à jour la plateforme de l'environnement dans le [tableau de bord \(p. 430\)](#) ou via un autre client, Elastic Beanstalk utilise le type de mise à jour que vous avez sélectionné pour les [changements de configuration \(p. 488\)](#).

Si vous n'avez planifié aucune mise à jour gérée, il est possible que votre environnement exécute déjà la dernière version. Voici d'autres raisons possibles pour lesquelles aucune mise à jour n'est planifiée :

- Une mise à jour de [version mineure \(p. 504\)](#) est disponible, mais votre environnement est configuré pour n'appliquer automatiquement que les mises à jour correctives de version.
- Votre environnement n'a pas été analysé depuis la publication de la mise à jour. Elastic Beanstalk vérifie généralement les mises à jour toutes les heures.
- Une mise à jour est en attente ou déjà en cours.

Lorsque la fenêtre de maintenance démarre ou que vous choisissez Appliquer maintenant, les mises à jour planifiées passent à l'état « en attente » avant l'exécution.

## Espaces de noms avec options d'action gérée

Vous pouvez utiliser les [options de configuration \(p. 658\)](#) des espaces de noms `aws:elasticbeanstalk:managedactions` ([p. 707](#)) et `aws:elasticbeanstalk:managedactions:platformupdate` ([p. 708](#)) pour activer et configurer les mises à jour gérées de la plateforme.

L'option `ManagedActionsEnabled` active les mises à jour gérées de la plateforme. Définissez cette option sur `true` pour activer les mises à jour gérées de la plateforme, et utilisez les autres options pour configurer le comportement de mise à jour.

Utilisez `PreferredStartTime` pour configurer le début du créneau de maintenance hebdomadaire au format `jour:heure:minute`.

Définissez `UpdateLevel` sur `minor` ou `patch` afin d'appliquer respectivement des mises à jour de version mineure et de version corrective, ou uniquement des mises à jour de version corrective.

Lorsque les mises à jour gérées de la plateforme sont activées, vous pouvez activer le remplacement d'instance en définissant l'option `InstanceRefreshEnabled` sur `true`. Lorsque ce paramètre est activé, Elastic Beanstalk exécute une mise à jour immuable de l'environnement chaque semaine, qu'une nouvelle version de la plateforme soit disponible ou non.

L'exemple suivant de [fichier de configuration \(p. 737\)](#) active les mises à jour gérées de la plateforme pour les mises à jour correctives de version, avec une fenêtre de maintenance commençant à 9 h UTC chaque mardi.

### Example .ebextensions/managed-platform-update.config

```
option_settings:  
  aws:elasticbeanstalk:managedactions:  
    ManagedActionsEnabled: true  
    PreferredStartTime: "Tue:09:00"  
  aws:elasticbeanstalk:managedactions:platformupdate:  
    UpdateLevel: patch  
    InstanceRefreshEnabled: true
```

## Migration de votre application depuis une version de plateforme héritée

Si vous avez déployé une application Elastic Beanstalk qui utilise une version de plateforme héritée, vous devez migrer votre application vers un nouvel environnement utilisant une version de plateforme non héritée afin d'accéder aux nouvelles fonctions. Si vous n'êtes pas sûr d'exécuter votre application à l'aide d'une version de plate-forme héritée, vous pouvez vérifier cette information dans la console Elastic Beanstalk. Pour obtenir des instructions, consultez [Pour vérifier si vous utilisez une version de plate-forme héritée \(p. 507\)](#).

### Quelles nouvelles fonctionnalités ne figurent pas dans les versions de plate-forme héritée ?

Les plateformes héritées ne prennent pas en charge les fonctions suivantes :

- Fichiers de configuration, comme décrit dans la rubrique [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#)
- Vérifications de l'état ELB, comme décrit dans la rubrique [Création de rapports d'intégrité de base \(p. 834\)](#)
- Profils d'instance, comme décrit dans la rubrique [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#)
- VPC, comme décrit dans la rubrique [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#)
- Niveaux de données, comme décrit dans la rubrique [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#)
- Niveaux de travail, comme décrit dans la rubrique [Environnements de travail \(p. 16\)](#)
- Environnements d'instance unique, comme décrit dans la rubrique [Types d'environnement \(p. 519\)](#)
- Balises, comme décrit dans la rubrique [Balisage des ressources dans vos environnements Elastic Beanstalk \(p. 629\)](#)
- Mises à jour propagées, comme décrit dans la rubrique [Mises à jour propagées de la configuration de l'environnement Elastic Beanstalk \(p. 489\)](#)

### Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?

Certaines anciennes versions de plateforme ne prennent pas en charge les dernières fonctions Elastic Beanstalk. Ces versions affichent la mention (legacy) [(héritée)] sur la page de présentation de l'environnement dans la console Elastic Beanstalk.

[Pour vérifier si vous utilisez une version de plate-forme héritée](#)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, affichez le nom de la Platform (Plateforme).

Votre application utilise une version de plateforme héritée si (legacy) [(héritée)] apparaît en regard du nom de la plateforme.

Pour migrer votre application

1. Déployez votre application dans un nouvel environnement. Pour plus d'informations, consultez [Création d'un environnement Elastic Beanstalk \(p. 439\)](#).
2. Si vous disposez d'une instance DB Amazon RDS, mettez à jour votre groupe de sécurité de base de données afin d'autoriser l'accès à votre groupe de sécurité EC2 pour votre nouvel environnement. Pour savoir comment trouver le nom de votre groupe de sécurité EC2 via la console de gestion AWS, consultez [Groupes de sécurité \(p. 542\)](#). Pour de plus amples informations sur la configuration de votre groupe de sécurité EC2, veuillez consulter la section « Autoriser un accès réseau à un groupe de sécurité Amazon EC2 » de la page [Utilisation de groupes de sécurité de base de données](#) dans le Guide de l'utilisateur Amazon Relational Database Service.
3. Echangez votre URL d'environnement. Pour plus d'informations, consultez [Déploiements bleu/vert avec Elastic Beanstalk \(p. 486\)](#).
4. Suspendez votre ancien environnement. Pour plus d'informations, consultez [Arrêt d'un environnement Elastic Beanstalk \(p. 464\)](#).

Note

Si vous utilisez AWS Identity and Access Management (IAM), vous devrez mettre à jour vos stratégies pour inclure AWS CloudFormation et Amazon RDS (le cas échéant). Pour plus d'informations, consultez [Utilisation d'Elastic Beanstalk avec AWS Identity and Access Management \(p. 920\)](#).

## Migration de votre application Elastic Beanstalk Linux vers Amazon Linux 2

AWS Elastic Beanstalk utilise [Amazon Linux 2](#) comme système d'exploitation pour les plateformes Linux. Les branches de plateforme de génération précédente basées sur l'[AMI Amazon Linux](#) sont désormais obsolètes. Pour plus d'informations sur les plateformes Linux, consultez [the section called “Plateformes Linux” \(p. 32\)](#).

Si votre application Elastic Beanstalk est basée sur une branche de plateforme AMI Amazon Linux, utilisez cette page pour savoir comment migrer les environnements de votre application vers Amazon Linux 2. Les deux générations de plateforme ne sont pas garanties rétrocompatibles avec votre application existante. En outre, même si votre code d'application se déploie avec succès sur la nouvelle version de plateforme, il peut se comporter ou fonctionner différemment en raison des différences de système d'exploitation et d'exécution. Bien que l'AMI Amazon Linux et Amazon Linux 2 partagent le même noyau Linux, ils diffèrent au niveau de leur système d'initialisation, des versions `libc`, de la chaîne d'outils du compilateur, et de divers packages. Nous avons également mis à jour des versions de l'environnement d'exécution, d'outils de génération et d'autres dépendances spécifiques à la plateforme. Par conséquent, nous vous recommandons de prendre votre temps, de tester soigneusement votre application dans un environnement de développement et d'effectuer les ajustements nécessaires.

Lorsque vous êtes prêt à passer en production, Elastic Beanstalk nécessite un déploiement bleu/vert pour effectuer la mise à niveau. Pour plus d'informations sur les stratégies de mise à jour des plateformes, consultez [the section called “Mises à jour de plateforme” \(p. 496\)](#).

#### Note

La console Elastic Beanstalk fournit des conseils de migration spécifiques à la plateforme pour de nombreuses branches de plateforme AMI Amazon Linux obsolètes. Vous pouvez voir ces informations lorsque vous affichez le tableau de bord d'un environnement qui utilise une de ces branches de plateforme obsolètes, ou lorsque vous choisissez l'une de ces branches lors de la création d'un environnement. Dans les deux cas, la console affiche un avertissement avec un lien Infos. Pour afficher les informations de migration de cette plateforme, choisissez Infos. Un panneau d'aide s'ouvre et affiche les informations associées.

## Considérations pour toutes les plateformes Linux

Le tableau suivant présente les considérations que vous devez prendre en compte lors de la planification de la migration d'une application vers Amazon Linux 2. Ces considérations s'appliquent à toutes les plateformes Linux Elastic Beanstalk, quels que soient les langages de programmation ou les serveurs d'applications spécifiques.

Area	Modifications et informations
Fichiers de configuration	<p>Sur les plateformes Amazon Linux 2, vous pouvez utiliser les <a href="#">fichiers de configuration (p. 737)</a> comme précédemment, et toutes les sections fonctionnent de la même manière. Cependant, des paramètres spécifiques peuvent ne pas fonctionner de la même manière que sur les plateformes AMI Amazon Linux précédentes. Exemples :</p> <ul style="list-style-type: none"><li>• Certains packages logiciels que vous installez à l'aide d'un fichier de configuration peuvent ne pas être disponibles sur Amazon Linux 2, ou leur nom peut avoir changé.</li><li>• Certaines options de configuration spécifiques à la plateforme ont vu leur espace de noms spécifique à la plate-forme être changé en un espace de noms différent et indépendant de la plateforme.</li><li>• Les fichiers de configuration de proxy fournis dans le répertoire <code>.ebextensions/nginx</code> doivent être déplacés vers le répertoire des hooks de plateforme <code>.platform/nginx</code>. Pour plus d'informations, développez la section Reverse Proxy Configuration dans <a href="#">the section called “Extension des plateformes Linux” (p. 34)</a>.</li></ul> <p>Nous vous recommandons d'utiliser des hooks de plateforme pour exécuter du code personnalisé sur vos instances d'environnement. Vous pouvez toujours utiliser des commandes et des commandes de conteneur dans les fichiers de configuration <code>.ebextensions</code>, mais elles ne sont pas aussi simples à utiliser. Par exemple, écrire des scripts de commande dans un fichier YAML peut être lourd et difficile à tester.</p> <p>Vous devez toujours utiliser des fichiers de configuration <code>.ebextensions</code> pour tout script nécessitant une référence à une ressource AWS CloudFormation.</p>
Hooks de plateforme	Amazon Linux 2 introduit une nouvelle façon d'étendre la plateforme de votre environnement en ajoutant des fichiers exécutables aux répertoires de crochets sur les instances de l'environnement. Avec les versions précédentes de la plateforme Linux, vous avez peut-être utilisé des <a href="#">hooks de plateforme personnalisée (p. 1153)</a> . Ces hooks n'étaient pas conçus pour les plateformes gérées et n'étaient pas pris en charge, mais pouvaient fonctionner de manière utile dans certains cas. Avec les versions de plateforme Amazon Linux 2, les crochets de plateforme personnalisés ne fonctionnent pas. Vous devez migrer tous les hooks vers les nouveaux hooks de plateforme. Pour plus de détails, développez la section Platform Hooks dans <a href="#">the section called “Extension des plateformes Linux” (p. 34)</a> .

Area	Modifications et informations
Serveurs proxy pris en charge	<p>Les versions de plateforme Amazon Linux 2 prennent en charge les mêmes serveurs proxy inverse que chaque plateforme prise en charge dans ses versions de plateforme d'AMI Amazon Linux. Toutes les versions de plateforme Amazon Linux 2 utilisent nginx comme serveur proxy inverse par défaut. Les plateformes Tomcat, Node.js, PHP et Python prennent également en charge Apache HTTPD comme alternative. Toutes les plateformes permettent la configuration du serveur proxy de manière uniforme, comme décrit dans cette section. Toutefois, la configuration du serveur proxy est légèrement différente de celle de l'AMI Amazon Linux. Voici les différences pour toutes les plateformes :</p> <ul style="list-style-type: none"> <li>La valeur par défaut est nginx – Le serveur proxy par défaut sur toutes les versions de la plateforme Amazon Linux 2 est nginx. Sur les versions de plateforme AMI Amazon Linux de Tomcat, PHP et Python, le serveur proxy par défaut était Apache HTTPD.</li> <li>Espace de noms cohérent – Toutes les versions de la plateforme Amazon Linux 2 utilisent l'espace de noms <code>aws:elasticbeanstalk:environment:proxy</code> pour configurer le serveur proxy. Sur les versions de plateforme AMI Amazon Linux, il s'agissait d'une décision par plateforme, et Node.js utilisait un espace de noms différent.</li> <li>Emplacement du fichier de configuration – Vous devez placer les fichiers de configuration proxy dans les répertoires <code>.platform/nginx</code> et <code>.platform/httpd</code> sur toutes les versions de la plateforme Amazon Linux 2. Sur les versions de la plateforme AMI Amazon Linux, ces emplacements étaient <code>.ebextensions/nginx</code> et <code>.ebextensions/httpd</code>, respectivement.</li> </ul> <p>Pour les modifications de configuration de proxy spécifiques à la plateforme, veuillez consulter <a href="#">the section called “Considérations spécifiques à la plateforme” (p. 511)</a>. Pour de plus amples informations sur la configuration de proxy Amazon Linux 2, développez la section Configuration de proxy inverse dans <a href="#">the section called “Extension des plateformes Linux” (p. 34)</a>.</p>
Profil d'instance	Les plateformes Amazon Linux 2 nécessitent la configuration d'un profil d'instance. La création de l'environnement peut aboutir temporairement sans profil d'instance, mais l'environnement peut afficher des erreurs rapidement après sa création lorsque des actions nécessitant un profil d'instance commencent à échouer. Pour plus d'informations, consultez <a href="#">the section called “Profils d'instance” (p. 921)</a> .
Intégrité améliorée	Les versions de plateforme Amazon Linux 2 autorisent l'état amélioré par défaut. Il s'agit d'un changement si vous n'utilisez pas la console Elastic Beanstalk pour créer vos environnements. La console active l'intégrité améliorée par défaut chaque fois que cela est possible, quelle que soit la version de la plateforme. Pour plus d'informations, consultez <a href="#">the section called “Surveillance et création de rapports d'intégrité améliorée” (p. 837)</a> .
AMI personnalisée	Si votre environnement utilise une <a href="#">AMI personnalisée (p. 787)</a> , créez une nouvelle AMI basée sur Amazon Linux 2 pour votre nouvel environnement à l'aide d'une plateforme Elastic Beanstalk Amazon Linux 2.
Plateformes personnalisées	Les AMI gérées des versions de la plateforme Amazon Linux 2 ne prennent pas en charge les <a href="#">plateformes personnalisées (p. 1147)</a> .

## Considérations spécifiques à la plateforme

Cette section traite des considérations de migration spécifiques à certaines plateformes Linux Elastic Beanstalk.

### Docker

Le tableau ci-dessous répertorie les informations de migration pour les versions de plateforme Amazon Linux 2 dans la [plateforme Docker \(p. 46\)](#).

Area	Modifications et informations
Stockage	<p>Elastic Beanstalk configure Docker pour qu'il utilise des <a href="#">pilotes de stockage</a> afin de stocker des images Docker et des données de conteneur. Sur l'AMI Amazon Linux, Elastic Beanstalk a utilisé le <a href="#">pilote de stockage Device Mapper</a>. Pour améliorer les performances, Elastic Beanstalk a provisionné un volume Amazon EBS supplémentaire. Sur les versions de la plateforme Amazon Linux 2 Docker, Elastic Beanstalk utilise le <a href="#">pilote de stockage OverlayFS</a>, et améliore encore les performances tout en ne nécessitant plus de volume distinct.</p> <p>Avec l'AMI Amazon Linux, si vous avez utilisé l'option <code>BlockDeviceMappings</code> de l'espace de noms <code>aws:autoscaling:launchconfiguration</code> pour ajouter des volumes de stockage personnalisés à un environnement Docker, nous vous conseillons d'ajouter également le volume <code>/dev/xvdcz</code> Amazon EBS prévu par Elastic Beanstalk. Elastic Beanstalk ne provisionne plus ce volume, vous devriez donc le supprimer de vos fichiers de configuration. Pour plus d'informations, consultez <a href="#">the section called “Configuration Docker sur l'AMI Amazon Linux (antérieure à Amazon Linux 2)” (p. 92)</a>.</p>
Authentification du référentiel privé	Lorsque vous fournissez un fichier d'authentification généré par Docker pour vous connecter à un référentiel privé, vous n'avez plus besoin de le convertir dans l'ancien format requis par les versions de la plateforme AMI Docker Amazon Linux. Les versions de la plateforme Amazon Linux 2 Docker prennent en charge le nouveau format. Pour plus d'informations, consultez <a href="#">the section called “Utilisation d'images à partir d'un référentiel privé” (p. 90)</a> .
Serveur proxy	Les versions de la plateforme Amazon Linux 2 Docker ne prennent pas en charge les conteneurs autonomes qui ne s'exécutent pas derrière un serveur proxy. Sur les versions de la plateforme AMI Docker Amazon Linux, cela était possible grâce à la valeur <code>none</code> de l'option <code>ProxyServer</code> dans l'espace de noms <code>aws:elasticbeanstalk:environment:proxy</code> .

### Go

Le tableau ci-dessous répertorie les informations de migration pour les versions de plateforme Amazon Linux 2 dans la [plateforme Go \(p. 101\)](#).

Area	Modifications et informations
Transmission de port	Sur les plateformes Amazon Linux 2, Elastic Beanstalk ne transmet pas une valeur de port à votre processus d'application via la variable d'environnement <code>PORT</code> . Vous pouvez simuler ce comportement pour votre processus en configurant vous-même une propriété d'environnement <code>PORT</code> . Cependant, si vous avez plusieurs processus et que vous comptez sur Elastic Beanstalk pour transmettre des valeurs de port incrémentielles à vos processus (5000, 5100, 5200, etc.), vous devez modifier votre implémentation. Pour plus d'informations, développez la section Configuration du proxy inverse dans <a href="#">the section called “Extension des plateformes Linux” (p. 34)</a> .

## Amazon Corretto

Le tableau ci-dessous répertorie les informations de migration pour les branches de plateforme Corretto dans la [plateforme Java SE \(p. 129\)](#).

Area	Modifications et informations
Corretto et OpenJDK	Pour implémenter la plateforme Java édition standard (Java SE), les branches de plateforme Amazon Linux 2 utilisent <a href="#">Amazon Corretto</a> , une distribution AWS de l'Open Java Development Kit (OpenJDK). Les branches précédentes de la plateforme Elastic Beanstalk Java SE utilisent les packages OpenJDK inclus avec l'AMI Amazon Linux.
Outils de génération	Les plateformes Amazon Linux 2 disposent de versions d'outils de génération plus récentes : <code>gradle</code> , <code>maven</code> et <code>ant</code> .
Gestion des fichiers JAR	Sur les plateformes Amazon Linux 2, si votre bundle de fichiers source (fichier ZIP) contient un seul fichier JAR et aucun autre fichier, Elastic Beanstalk ne renomme plus le fichier JAR <code>application.jar</code> . Le changement de nom se produit uniquement si vous soumettez un fichier JAR seul et non inclus dans un fichier ZIP.
Transmission de port	Sur les plateformes Amazon Linux 2, Elastic Beanstalk ne transmet pas une valeur de port à votre processus d'application via la variable d'environnement <code>PORT</code> . Vous pouvez simuler ce comportement pour votre processus en configurant vous-même une propriété d'environnement <code>PORT</code> . Cependant, si vous avez plusieurs processus et que vous comptez sur Elastic Beanstalk pour transmettre des valeurs de port incrémentielles à vos processus (5000, 5100, 5200, etc.), vous devez modifier votre implémentation. Pour plus d'informations, développez la section Configuration du proxy inverse dans <a href="#">the section called “Extension des plateformes Linux” (p. 34)</a> .
Java 7	Elastic Beanstalk ne prend pas en charge une branche de plateforme Amazon Linux 2 Java 7. Si vous avez une application Java 7, migrez vers Corretto 8 ou Corretto 11.

## Tomcat

Le tableau ci-dessous répertorie les informations de migration pour les versions de plateforme Amazon Linux 2 dans la [plateforme Tomcat \(p. 118\)](#).

Area	Modifications et informations						
Options de configuration	<p>Sur les versions de plateforme Amazon Linux 2, Elastic Beanstalk ne prend en charge qu'un sous-ensemble des options de configuration et des valeurs d'option dans l'espace de noms <code>aws:elasticbeanstalk:environment:proxy</code>. Voici les informations de migration pour chaque option.</p> <table border="1"> <thead> <tr> <th>Option</th><th>Informations sur la migration</th></tr> </thead> <tbody> <tr> <td><code>GzipCompression</code></td><td>Non pris en charge sur les versions de plateforme Amazon Linux 2.</td></tr> <tr> <td><code>ProxyServer</code></td><td>Les versions de plateforme Amazon Linux 2 Tomcat prennent en charge à la fois les serveurs proxy nginx et Apache HTTPD version 2.4. Cependant, Apache version 2.2 n'est pas prise en charge.  Sur les versions de la plateforme AMI Amazon Linux, le proxy par défaut était Apache 2.4. Si vous avez utilisé le paramètre de proxy par défaut et ajouté des fichiers de configuration de proxy personnalisés, votre configuration de proxy doit toujours fonctionner sur Amazon Linux 2. Cependant, si vous avez utilisé la valeur de l'option <code>apache/2.2</code>, vous</td></tr> </tbody> </table>	Option	Informations sur la migration	<code>GzipCompression</code>	Non pris en charge sur les versions de plateforme Amazon Linux 2.	<code>ProxyServer</code>	Les versions de plateforme Amazon Linux 2 Tomcat prennent en charge à la fois les serveurs proxy nginx et Apache HTTPD version 2.4. Cependant, Apache version 2.2 n'est pas prise en charge.  Sur les versions de la plateforme AMI Amazon Linux, le proxy par défaut était Apache 2.4. Si vous avez utilisé le paramètre de proxy par défaut et ajouté des fichiers de configuration de proxy personnalisés, votre configuration de proxy doit toujours fonctionner sur Amazon Linux 2. Cependant, si vous avez utilisé la valeur de l'option <code>apache/2.2</code> , vous
Option	Informations sur la migration						
<code>GzipCompression</code>	Non pris en charge sur les versions de plateforme Amazon Linux 2.						
<code>ProxyServer</code>	Les versions de plateforme Amazon Linux 2 Tomcat prennent en charge à la fois les serveurs proxy nginx et Apache HTTPD version 2.4. Cependant, Apache version 2.2 n'est pas prise en charge.  Sur les versions de la plateforme AMI Amazon Linux, le proxy par défaut était Apache 2.4. Si vous avez utilisé le paramètre de proxy par défaut et ajouté des fichiers de configuration de proxy personnalisés, votre configuration de proxy doit toujours fonctionner sur Amazon Linux 2. Cependant, si vous avez utilisé la valeur de l'option <code>apache/2.2</code> , vous						

Area	Modifications et informations	
	Option	Informations sur la migration
		devez maintenant migrer votre configuration proxy vers Apache version 2.4.
		<p>L'option <code>XX:MaxPermSize</code> de l'espace de noms <code>aws:elasticbeanstalk:container:tomcat:jvmoptions</code> n'est pas prise en charge sur les versions de la plateforme Amazon Linux 2. Le paramètre JVM permettant de modifier la taille de la génération permanente s'applique uniquement à Java 7 et versions antérieures et n'est donc pas applicable aux versions de la plateforme Amazon Linux 2.</p>
Chemin d'accès de l'application	Sur les plateformes Amazon Linux 2, le chemin d'accès au répertoire de l'application sur les instances Amazon EC2 de votre environnement est <code>/var/app/current</code> . C'était <code>/var/lib/tomcat8/webapps</code> sur les plateformes AMI Amazon Linux.	

## Node.js

Le tableau ci-dessous répertorie les informations de migration pour les versions de plateforme Amazon Linux 2 dans la [plateforme Node.js \(p. 255\)](#).

Area	Modifications et informations				
Versions de Node.js installées	<p>Sur les plateformes Amazon Linux 2, Elastic Beanstalk conserve plusieurs branches de plateforme Node.js et installe uniquement la dernière version majeure de Node.js correspondant à la branche de plateforme sur chaque version de plateforme. Par exemple, chaque version de plateforme dans la branche de plateforme Node.js 12 n'a que Node.js 12.x.y installé par défaut. Sur les versions de plateforme AMI Amazon Linux, nous avons installé les versions multiples de plusieurs versions de Node.js sur chaque version de plateforme, et nous n'avons maintenu qu'une seule branche de plateforme.</p> <p>Choisissez la branche de plateforme Node.js qui correspond à la version majeure de Node.js dont votre application a besoin.</p>				
Noms des fichiers journaux HTTPD Apache	<p>Sur les plateformes Amazon Linux 2, si vous utilisez le serveur proxy HTTPD Apache, les noms des fichiers journaux HTTPD sont <code>access_log</code> et <code>error_log</code>, ce qui est cohérent avec toutes les autres plateformes prenant en charge Apache HTTPD. Sur les versions de la plateforme d'AMI Amazon Linux, ces fichiers journaux étaient nommés <code>access.log</code> et <code>error.log</code>, respectivement.</p> <p>Pour de plus amples informations sur les noms et les emplacements des fichiers journaux de toutes les plateformes, veuillez consulter <a href="#">the section called "Méthode de configuration de CloudWatch Logs par Elastic Beanstalk" (p. 898)</a>.</p>				
Options de configuration	<p>Sur les plateformes Amazon Linux 2, Elastic Beanstalk ne prend pas en charge les options de configuration de l'espace de noms <code>aws:elasticbeanstalk:container:nodejs</code>. Certaines options ont des solutions de rechange. Voici les informations de migration pour chaque option.</p> <table border="1"> <thead> <tr> <th>Option</th><th>Informations sur la migration</th></tr> </thead> <tbody> <tr> <td><code>NodeCommand</code></td><td>Utilisez un <code>Procfile</code> ou le mot-clé <code>scripts</code> dans un fichier <code>package.json</code> pour spécifier le script de démarrage.</td></tr> </tbody> </table>	Option	Informations sur la migration	<code>NodeCommand</code>	Utilisez un <code>Procfile</code> ou le mot-clé <code>scripts</code> dans un fichier <code>package.json</code> pour spécifier le script de démarrage.
Option	Informations sur la migration				
<code>NodeCommand</code>	Utilisez un <code>Procfile</code> ou le mot-clé <code>scripts</code> dans un fichier <code>package.json</code> pour spécifier le script de démarrage.				

Area	Modifications et informations	
	Option	Informations sur la migration
	NodeVersion	<p>Utilisez le mot-clé engines dans un fichier package.json pour spécifier la version de Node.js. Sachez que vous ne pouvez spécifier qu'une version de Node.js qui correspond à votre branche de plateforme. Par exemple, si vous utilisez la branche de plateforme Node.js 12, vous ne pouvez spécifier qu'une version 12.x.y de Node.js. Pour plus d'informations, consultez <a href="#">the section called "Spécification des dépendances Node.js avec un fichier package.json" (p. 260)</a>.</p>
	GzipCompression	<p>Nouvelles options en charge sur les versions de plateforme Amazon Linux 2.</p> <p>Sur les versions de la plateforme Amazon Linux 2 Node.js, cette option a été déplacée vers l'espace de noms aws:elasticbeanstalk:environment:proxy. Vous pouvez choisir entre nginx (par défaut) et apache.</p> <p>Les versions de la plateforme Amazon Linux 2 Node.js ne prennent pas en charge les applications autonomes qui ne s'exécutent pas derrière un serveur proxy. Sur les versions de la plateforme Node.js d'AMI Amazon Linux, cela était possible via la valeur none de l'option ProxyServer dans l'espace de noms aws:elasticbeanstalk:container:nodejs. Si votre environnement exécute une application autonome, mettez à jour votre code pour écouter le port vers lequel le serveur proxy (nginx ou Apache) transfère le trafic.</p> <pre>var port = process.env.PORT    8080; app.listen(port, function() {   console.log('Server running at http://127.0.0.1:%s', port); });</pre>

## PHP

Le tableau ci-dessous répertorie les informations de migration pour les versions de plateforme Amazon Linux 2 dans la [plateforme PHP \(p. 293\)](#).

Area	Modifications et informations
Traitement des fichiers PHP	Sur les plateformes Amazon Linux 2, les fichiers PHP sont traités à l'aide de PHP-FPM (gestionnaire de processus CGI). Sur les plateformes AMI Amazon Linux, nous avons utilisé mod_php (un module Apache).
Serveur proxy	<p>Les versions de plateforme Amazon Linux 2 PHP prennent en charge à la fois les serveurs proxy nginx et Apache HTTPD. La valeur par défaut est nginx.</p> <p>Les versions de la plateforme PHP AMI Amazon Linux prenaient uniquement en charge Apache HTTPD. Si vous avez ajouté des fichiers de configuration Apache personnalisés, vous pouvez définir l'option ProxyServer dans l'espace de noms aws:elasticbeanstalk:environment:proxy sur apache.</p>

## Python

Le tableau ci-dessous répertorie les informations de migration pour les versions de plateforme Amazon Linux 2 dans la [plateforme Python \(p. 359\)](#).

Area	Modifications et informations
Serveur WSGI	<p>Sur les plateformes Amazon Linux 2, Gunicorn est le serveur WSGI par défaut. Par défaut, Gunicorn écoute sur le port 8000. Le port peut être différent de celui utilisé par votre application sur la plateforme d'AMI Amazon Linux. Si vous définissez l'option <code>WSGIPath</code> de l'espace de noms <code>aws:elasticbeanstalk:container:python</code> (p. 734), remplacez la valeur par la syntaxe de Gunicorn. Pour plus d'informations, consultez <a href="#">the section called “Espaces de noms de la configuration Python” (p. 361)</a>.</p> <p>Vous pouvez également utiliser un <code>Procfile</code> pour spécifier et configurer le serveur WSGI. Pour plus d'informations, consultez <a href="#">the section called “Procfile” (p. 363)</a>.</p>
Chemin d'accès de l'application	<p>Sur les plateformes Amazon Linux 2, le chemin d'accès au répertoire de l'application sur les instances Amazon EC2 de votre environnement est <code>/var/app/current</code>. C'était <code>/opt/python/current/app</code> sur les plateformes AMI Amazon Linux.</p>
Serveur proxy	<p>Les versions de plateforme Amazon Linux 2 Python prennent en charge à la fois les serveurs proxy nginx et Apache HTTPD. La valeur par défaut est nginx.</p> <p>Les versions de plateforme AMI Python Amazon Linux prennent uniquement en charge Apache HTTPD. Si vous avez ajouté des fichiers de configuration Apache personnalisés, vous pouvez définir l'option <code>ProxyServer</code> dans l'espace de noms <code>aws:elasticbeanstalk:environment:proxy</code> sur apache.</p>

## Ruby

Le tableau ci-dessous répertorie les informations de migration pour les versions de plateforme Amazon Linux 2 dans la [plateforme Ruby \(p. 387\)](#).

Area	Modifications et informations
Versions de Ruby installées	<p>Sur les plateformes Amazon Linux 2, Elastic Beanstalk n'installe que la dernière version d'une seule version de Ruby, correspondant à la branche de plateforme, sur chaque version de plateforme. Par exemple, chaque version de plateforme de la branche de plateforme Ruby 2.6 dispose uniquement de Ruby 2.6.x installé. Sur les versions de la plateforme AMI Amazon Linux, nous avons installé les dernières versions de plusieurs versions de Ruby, par exemple 2.4.x, 2.5.x et 2.6.x.</p> <p>Si votre application utilise une version de Ruby qui ne correspond pas à la branche de plateforme que vous utilisez, nous vous recommandons de basculer vers une branche de plateforme qui dispose de la version de Ruby correcte pour votre application.</p>
Serveur d'application	<p>Sur les plateformes Amazon Linux 2, Elastic Beanstalk installe uniquement le serveur d'applications Puma sur toutes les versions de la plateforme Ruby. Vous pouvez utiliser un <code>Procfile</code> pour démarrer un autre serveur d'applications et un <code>Gemfile</code> pour l'installer.</p> <p>Sur la plateforme AMI Amazon Linux, nous avons pris en charge deux types de branches de plateforme pour chaque version Ruby : l'une avec le serveur d'applications Puma et l'autre avec le serveur d'applications Passenger. Si votre application utilise Passenger, vous pouvez configurer votre environnement Ruby pour installer et utiliser Passenger.</p>

Area	Modifications et informations
	Pour plus d'informations et d'exemples, consultez <a href="#">the section called “La plateforme Ruby” (p. 387)</a> .

## Annulation de mises à jour de configuration d'environnement et de déploiements d'application

Vous pouvez annuler des mises à jour en cours qui sont déclenchées par des changements de configuration d'environnement. Vous pouvez également annuler le déploiement d'une nouvelle version de l'application en cours. Par exemple, vous pouvez vouloir annuler une mise à jour si vous décidez que vous souhaitez continuer à utiliser la configuration de l'environnement existant au lieu d'appliquer les nouveaux paramètres de configuration d'environnement. Ou bien, vous pouvez réaliser que la nouvelle version de l'application que vous déployez a des problèmes susceptibles de ne pas démarrer ou de ne pas fonctionner correctement. En annulant une mise à jour d'environnement ou de version d'application, vous pouvez éviter d'attendre que le processus de mise à jour ou de déploiement soit effectué avant de lancer une nouvelle tentative pour mettre à jour la version de l'environnement ou de l'application.

### Note

Pendant la phase de nettoyage au cours de laquelle les anciennes ressources qui ne sont plus nécessaires sont supprimées, après que le dernier lot d'instances a été mis à jour, vous ne pouvez plus annuler la mise à jour.

Elastic Beanstalk effectue la restauration de la même manière qu'il a réalisé la dernière mise à jour réussie. Par exemple, si des mises à jour propagées à durée définie sont activées dans votre environnement, alors Elastic Beanstalk attend le temps de pause spécifié entre la restauration des modifications sur un lot d'instances avant de restaurer les modifications sur le lot suivant. Ou bien, si vous avez récemment activé des mises à jour propagées, mais que la dernière fois que vous avez réussi une mise à jour des paramètres de configuration de votre environnement elle n'incluait pas de mises à jour propagées, Elastic Beanstalk procède à la restauration sur toutes les instances en même temps.

Une fois qu'il commence à annuler la mise à jour, vous ne pouvez pas arrêter la restauration d'Elastic Beanstalk vers la configuration de l'environnement précédent. Le processus de restauration continue jusqu'à ce que toutes les instances de l'environnement aient la configuration de l'environnement précédent ou jusqu'à ce que le processus de restauration échoue. Pour les déploiements de version d'application, l'annulation du déploiement arrête tout simplement le déploiement ; certaines instances auront la nouvelle version de l'application et d'autres continueront à exécuter la version d'application existante. Vous pouvez déployer la même ou une autre version de l'application ultérieurement.

Pour plus d'informations sur les mises à jour propagées, consultez [Mises à jour propagées de la configuration de l'environnement Elastic Beanstalk \(p. 489\)](#). Pour plus d'informations sur les déploiements de version d'application par lots, consultez [Paramètres et stratégies de déploiement \(p. 479\)](#).

### Pour annuler une mise à jour

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Environment actions (Actions d'environnement), puis Abort current operation (Abandonner l'opération en cours).

## Reconstruction d'environnements Elastic Beanstalk

Votre environnement AWS Elastic Beanstalk peut devenir inutilisable si vous n'utilisez pas la fonctionnalité Elastic Beanstalk pour modifier ou suspendre les ressources AWS sous-jacentes de l'environnement. Si cela se produit, vous pouvez reconstruire l'environnement pour tenter de le restaurer à un état de fonctionnement. La reconstruction d'un environnement met hors service toutes les ressources de celui-ci et les remplace par de nouvelles ressources avec la même configuration.

Vous pouvez également reconstruire des environnements suspendus dans les six semaines (42 jours) suivant leur suspension. Lors de la reconstruction, Elastic Beanstalk tente de créer un nouvel environnement avec le même nom, le même ID et la même configuration.

### Reconstruction d'un environnement en cours d'exécution

Vous pouvez reconstruire un environnement via la console Elastic Beanstalk ou à l'aide de l'API `RebuildEnvironment`.

Pour reconstruire un environnement en cours d'exécution (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis sélectionnez Rebuild environment (Reconstruire l'environnement).
4. Choisissez Rebuild (Reconstruire).

La reconstruction d'un environnement en cours d'exécution crée de nouvelles ressources ayant la même configuration que les anciennes ressources ; toutefois, les ID de ressource sont différents et les données figurant sur les anciennes ressources ne sont pas restaurées. Par exemple, la reconstruction d'un environnement avec une instance de base de données Amazon RDS crée une nouvelle base de données avec la même configuration, mais elle n'applique aucun instantané à la nouvelle base de données.

Pour reconstruire un environnement en cours d'exécution avec l'API Elastic Beanstalk, utilisez l'action `RebuildEnvironment` avec l'AWS CLI ou le kit SDK AWS.

```
$ aws elasticbeanstalk rebuild-environment --environment-id e-vdnftxubwg
```

### Reconstruction d'un environnement suspendu

Vous pouvez reconstruire et restaurer un environnement arrêté à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande EB ou de l'API `RebuildEnvironment`.

### Note

Si vous n'utilisez pas votre propre nom de domaine personnalisé avec votre environnement suspendu, l'environnement utilise un sous-domaine d'elasticbeanstalk.com. Ces sous-domaines sont partagés au sein d'une région Elastic Beanstalk. Ils peuvent donc être utilisés par un environnement créé par un client dans la même région. Si votre environnement a été suspendu, un autre environnement peut utiliser son sous-domaine. Dans ce cas, la reconstruction échoue. Vous pouvez éviter cette erreur en utilisant un domaine personnalisé. Pour de plus amples informations, veuillez consulter [Nom de domaine de votre environnement Elastic Beanstalk \(p. 656\)](#).

Les environnements récemment suspendus apparaissent dans la présentation des applications pendant une heure au maximum. Au cours de cette période, vous pouvez afficher les événements de l'environnement dans le [tableau de bord \(p. 429\)](#) de celui-ci et utiliser l'[action \(p. 432\)](#) Restore environment (Restaurer l'environnement) pour le reconstruire.

Pour reconstruire un environnement qui n'est plus visible, utilisez l'option Restore terminated environment (Restaurer l'environnement résilié) depuis la page d'application.

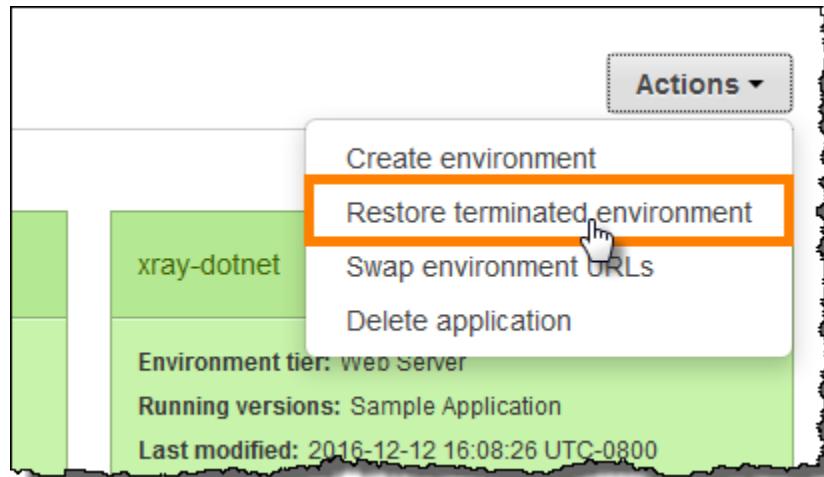
Pour reconstruire un environnement suspendu (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Choisissez Actions, puis Restore terminated environment (Restaurer l'environnement résilié).



4. Choisissez un environnement suspendu.
5. Choisissez Restore.

Name	Date terminated	Platform configuration	App
beta-java	2016-12-12 10:50:49 UTC-0800	64bit Amazon Linux 2016.03 v2.1.6 running Java 8	app-

Elastic Beanstalk tente de créer un nouvel environnement avec le même nom, le même ID et la même configuration. S'il existe un environnement avec le même nom ou la même URL lorsque vous tentez de reconstruire, la reconstruction échoue. La suppression de la version d'application qui avait été déployée dans l'environnement entraîne également l'échec de la reconstruction.

Si vous utilisez l'interface de ligne de commande EB pour gérer votre environnement, exécutez la commande `eb restore` pour reconstruire un environnement suspendu.

```
$ eb restore e-vdnftxubwq
```

Consultez [eb restore \(p. 1111\)](#) pour plus d'informations.

Pour reconstruire un environnement arrêté avec l'API Elastic Beanstalk, utilisez l'action `RebuildEnvironment` avec l'AWS CLI ou le kit SDK AWS.

```
$ aws elasticbeanstalk rebuild-environment --environment-id e-vdnftxubwq
```

## Types d'environnement

Dans AWS Elastic Beanstalk, vous pouvez créer un environnement évolutif, d'équilibrage de charge ou un environnement instance unique. Le type d'environnement dont vous avez besoin dépend de l'application que vous déployez. Par exemple, vous pouvez développer et tester une application dans un environnement instance unique pour économiser de l'argent, mettre à niveau cet environnement vers un environnement évolutif, d'équilibrage de charge lorsque l'application est prête pour la production.

### Note

Un niveau d'environnement de travail pour une application web qui traite des tâches en arrière-plan n'inclut pas un équilibrer de charge. Cependant, un environnement de travail monte efficacement en puissance en ajoutant des instances au groupe Auto Scaling pour traiter les données à partir de la file d'attente Amazon SQS lorsque la charge le nécessite.

## Environnement évolutif et équilibré en charge

Un environnement évolutif et équilibré de charge utilise les services Elastic Load Balancing et Amazon EC2 Auto Scaling pour provisionner les instances Amazon EC2 requises pour votre application déployée.

Amazon EC2 Auto Scaling démarre automatiquement les instances supplémentaires pour vous adapter à une charge croissante sur votre application. Si la charge sur votre application diminue, Amazon EC2 Auto Scaling arrête les instances mais laisse toujours votre nombre d'instances minimum spécifié en cours d'exécution. Si votre application a besoin d'une évolutivité avec l'option d'exécution dans plusieurs zones de disponibilité, utilisez un environnement évolutif et d'équilibrage de charge. Si vous ne savez pas quel type d'environnement sélectionner, vous pouvez en choisir un et, si nécessaire, changer le type d'environnement plus tard.

## Environnement à instance unique

Un environnement instance unique contient une instance d'Amazon EC2 avec une adresse IP Elastic. Un environnement instance unique n'a pas d'équilibrage de charge, ce qui peut vous aider à réduire les coûts par rapport à un environnement évolutif et d'équilibrage de charge. Même si un environnement instance unique utilise le service Amazon EC2 Auto Scaling, les paramètres pour le nombre minimum d'instances, le nombre maximum d'instances et la capacité souhaitée sont tous sur 1. Par conséquent, de nouvelles instances ne sont pas démarrées pour adapter la charge croissante sur votre application.

Utilisez un environnement instance unique si vous attendez un faible trafic de votre application de production ou si vous effectuez un développement à distance. Si vous ne savez pas quel type d'environnement sélectionner, vous pouvez en choisir un et, si nécessaire, changer le type d'environnement plus tard. Pour plus d'informations, consultez [Changement de type d'environnement \(p. 520\)](#).

## Changement de type d'environnement

Vous pouvez remplacer votre type d'environnement par un environnement instance unique ou un environnement évolutif, d'équilibrage de charge, en modifiant la configuration de votre environnement. Dans certains cas, vous souhaiterez peut-être modifier votre type d'environnement d'un type à un autre. Imaginons par exemple que vous avez développé et testé une application dans un environnement instance unique afin d'économiser de l'argent. Lorsque votre application est prête pour la production, vous pouvez remplacer le type d'environnement par un environnement évolutif, d'équilibrage de charge afin qu'il puisse évoluer pour répondre aux exigences de vos clients.

Pour modifier le type d'un environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la liste Type d'environnement, sélectionnez un type d'environnement.

## Modify capacity

Configure the compute capacity of your environment and Auto Scaling settings to optimize the number of instances used.

### Auto Scaling Group

#### Environment type

Load balanced

#### Instances

Min 1

Max 2

#### Fleet composition

Choose a mix of On-Demand and Spot Instances with multiple instance types. Spot Instances are automatically launched at the lowest available price.

On-Demand instances

Combine purchase options and instances

#### Maximum spot price

The maximum price per instance-hour, in USD, that you're willing to pay for a Spot Instance. Setting a custom price limits your chances to find instances.

6. Choisissez Enregistrer.

La mise à jour de l'environnement peut nécessiter plusieurs minutes pendant qu'Elastic Beanstalk alloue les ressources AWS.

Si votre environnement se trouve dans un VPC, sélectionnez les sous-réseaux dans lesquels placer les instances Elastic Load Balancing et Amazon EC2. Chaque zone de disponibilité dans laquelle votre application s'exécute doit avoir les deux. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#).

## Environnements de travail Elastic Beanstalk

Si votre application AWS Elastic Beanstalk effectue des opérations ou des flux de travail qui prennent beaucoup de temps, vous pouvez décharger ces tâches dans un environnement de travail dédié. Le découplage de l'élément frontal de votre application web d'un processus qui effectue des opérations de blocage est une façon courante de garantir que votre application reste réactive sous charge.

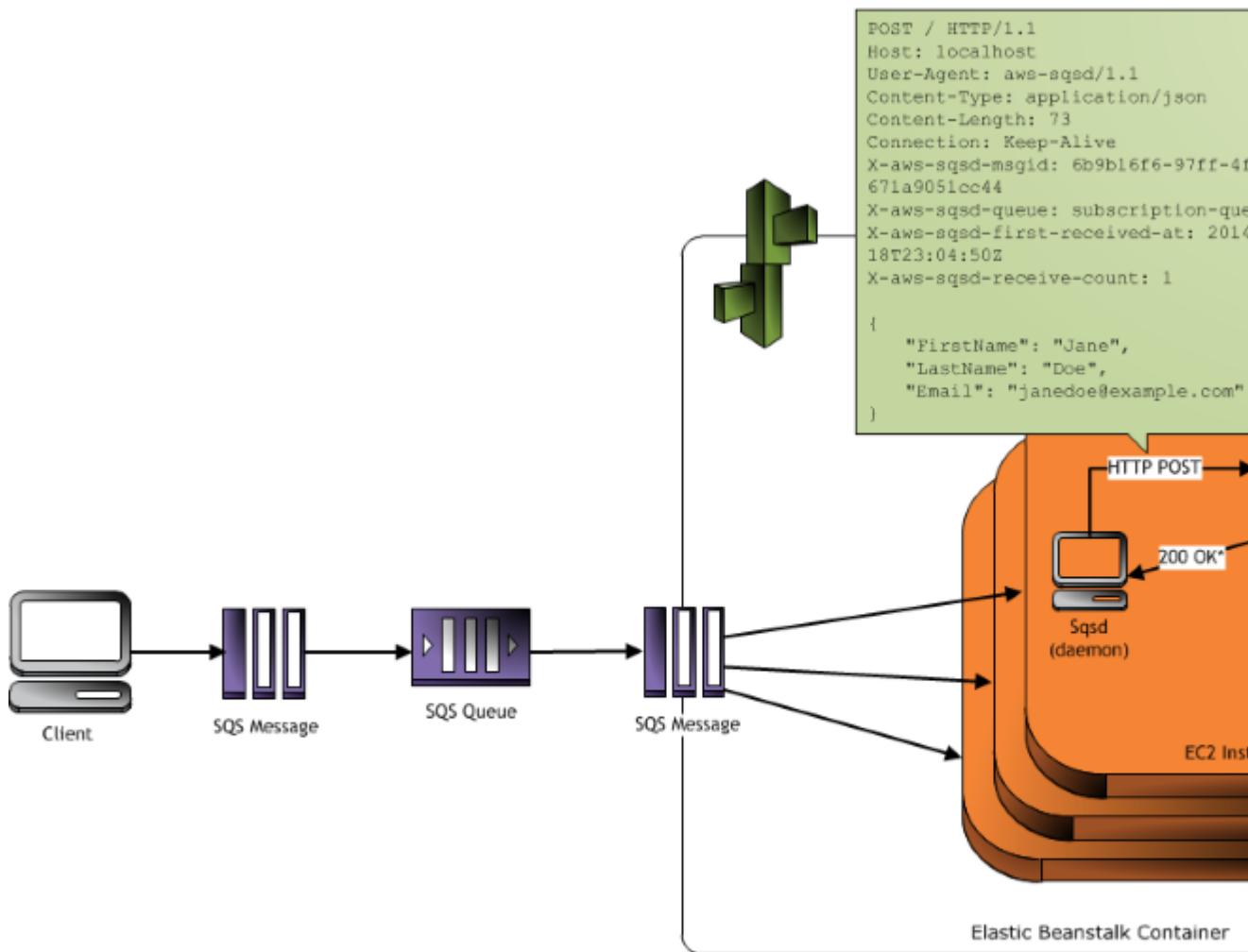
Une tâche à long terme est tout ce qui accroît considérablement le temps nécessaire pour effectuer une demande, tel que le traitement d'images ou de vidéos, l'envoi d'e-mails ou la génération d'une archive ZIP. Ces opérations peuvent ne prendre qu'une ou deux secondes, mais un délai de quelques secondes est beaucoup pour une requête web qui aboutirait sinon en moins de 500 ms.

Une option consiste à générer dynamiquement un processus de travail localement, à renvoyer une réussite et à traiter la tâche de manière asynchrone. Cela fonctionne si votre instance peut faire face à toutes

les tâches qui lui sont envoyées. Sous une charge élevée, toutefois, une instance peut être submergée par des tâches en arrière-plan et arrêter de répondre à des demandes de priorité plus élevée. Si des utilisateurs peuvent générer plusieurs tâches, l'augmentation de la charge peut ne pas correspondre à une augmentation des utilisateurs, ce qui complique l'augmentation de la taille de votre niveau de serveur web efficacement.

Pour éviter d'exécuter des tâches de longue durée localement, vous pouvez utiliser le kit SDK AWS comme langage de programmation pour les envoyer à une file d'attente Amazon Simple Queue Service (Amazon SQS) et exécuter le processus qui les exécute sur un ensemble distinct d'instances. Les instances de travail prennent des éléments de la file d'attente uniquement lorsqu'elles ont la capacité nécessaire pour les exécuter, ce qui leur évite d'être submergées.

Les environnements de travail Elastic Beanstalk simplifient ce processus en gérant la file d'attente Amazon SQS et en exécutant un [processus démon \(p. 524\)](#) sur chaque instance qui lit automatiquement depuis la file d'attente. Lorsque le démon extrait un élément de la file d'attente, il envoie une demande HTTP POST localement à `http://localhost/` sur le port 80 avec le contenu du message de file d'attente dans le corps. Tout ce que votre application doit faire consiste à effectuer la tâche longue durée en réponse au POST. Vous pouvez [configurer le démon \(p. 527\)](#) pour publier vers un chemin d'accès différent, utiliser un type MIME autre qu'application/JSON, vous connecter à une file d'attente existante ou personnaliser des connexions (nombre maximal de demandes simultanées), des expirations et des nouvelles tentatives de connexion.



\* HTTP Response of 200 OK = delete the message  
Any other HTTP Response = retry the message after the specified interval  
No response = retry the message after the Inactivity timeout

Avec des [tâches périodiques \(p. 525\)](#), vous pouvez également configurer le démon de travail pour mettre en attente les messages en fonction d'une planification CRON. Chaque tâche périodique peut PUBLIER vers un chemin différent. Activez des tâches périodiques en incluant un fichier YAML dans votre code source qui définit le calendrier et le chemin d'accès pour chaque tâche.

#### Note

La plateforme [.NET sous Windows Server \(p. 192\)](#) ne prend pas en charge les environnements de travail.

#### Sections

- [Démon SQS d'environnement de travail \(p. 524\)](#)
- [Files d'attente de lettres mortes \(p. 525\)](#)
- [Tâches périodiques \(p. 525\)](#)
- [Utilisation d'Amazon CloudWatch pour la mise à l'échelle automatique dans les niveaux d'environnement de travail \(p. 526\)](#)
- [Configuration des environnements de travail \(p. 527\)](#)

## Démon SQS d'environnement de travail

Les environnements de travail exécutent un processus démon fourni par Elastic Beanstalk. Ce démon est mis à jour régulièrement pour ajouter des fonctionnalités et résoudre des bogues. Pour obtenir la dernière version du démon, effectuez la mise à jour vers la [version de la plateforme \(p. 31\)](#) la plus récente.

Lorsque l'application dans l'environnement de travail renvoie une réponse 200 OK pour reconnaître qu'elle a reçu et traité avec succès la demande, le démon envoie un appel `DeleteMessage` à la file d'attente Amazon SQS afin que le message soit supprimé de la file d'attente. Si l'application renvoie une réponse autre que 200 OK, Elastic Beanstalk attend l'expiration de la période `ErrorVisibilityTimeout` configurée pour remettre le message dans la file d'attente. S'il n'y a pas de réponse, Elastic Beanstalk attend pour remettre le message dans la file d'attente après la période `InactivityTimeout` afin que le message soit disponible pour une autre tentative au moment du traitement.

### Note

Les propriétés des files d'attente Amazon SQS (ordre des messages, livraison au moins une fois et échantillonnage de message) peuvent affecter la façon dont vous concevez une application web pour un environnement de travail. Pour plus amples informations, veuillez consulter à [Propriétés des files d'attente distribuées](#) dans le [Guide du développeur Amazon Simple Queue Service](#).

Amazon SQS supprime automatiquement les messages qui ont été dans une file d'attente pendant une durée supérieure à la `RetentionPeriod` configurée.

Le démon définit les en-têtes HTTP suivants.

En-têtes HTTP	
Nom	Valeur
User-Agent	aws-sqsd
	aws-sqsd/1.11
X-Aws-Sqsd-Msgid	ID de message SQS, utilisé pour détecter des tempête de messages (un nombre anormalement élevé de nouveaux messages).
X-Aws-Sqsd-Queue	Nom de la file d'attente SQS.
X-Aws-Sqsd-First-Received-At	Le temps UTC, au <a href="#">format ISO 8601</a> , lorsque le message a été reçu pour la première fois.
X-Aws-Sqsd-Receive-Count	Nombre de message SQS reçus.
X-Aws-Sqsd-Attr- <i>message-attribute-name</i>	Attributs de message personnalisé attribués au message en cours de traitement. Le <code>message-attribute-name</code> est le nom d'attribut de message réel. Tous les attributs de message de numéro et de chaîne sont ajoutés à l'en-tête. Les attributs binaires sont ignorés et ils ne sont pas inclus dans l'en-tête.
Content-Type	Configuration de type MIME ; par défaut, <code>application/json</code> .

## Files d'attente de lettres mortes

Les environnements de travail Elastic Beanstalk prennent en charge les files d'attente de lettres mortes Amazon Simple Queue Service (Amazon SQS). Une file d'attente de lettres mortes est une file d'attente où les autres files d'attente (source) peuvent envoyer des messages qui, pour une raison quelconque, n'ont pas pu être traités avec succès. Un avantage principal de l'utilisation d'une file d'attente de lettres mortes est la possibilité d'isoler les messages dont le traitement a échoué. Vous pouvez ensuite analyser les messages envoyés à la file d'attente de lettres mortes pour tenter de déterminer pourquoi leur traitement a échoué.

Une file d'attente de lettres mortes est activée par défaut pour un environnement de travail si vous spécifiez une file d'attente Amazon SQS générée automatiquement au moment de la création de votre couche d'environnement de travail. Si vous sélectionnez une file d'attente SQS existante pour votre environnement de travail, vous devez utiliser SQS pour configurer une file d'attente de lettres mortes de façon indépendante. Pour de plus amples informations sur la manière d'utiliser SQS pour configurer une file d'attente de lettres mortes, veuillez consulter [Utilisation des files d'attente de lettres mortes Amazon SQS](#).

Vous ne pouvez pas désactiver les files d'attente de lettres mortes. Les messages qui ne peuvent pas être diffusés sont toujours envoyés à terme vers une file d'attente de lettres mortes. Vous pouvez, toutefois, désactiver effectivement cette fonctionnalité en définissant l'option `MaxRetries` à la valeur valide maximale de 100.

Si une file d'attente de lettres mortes n'est pas configurée pour la file d'attente Amazon SQS de votre environnement de travail, Amazon SQS conserve les messages dans la file d'attente jusqu'à l'expiration de la période de rétention. Pour de plus amples informations sur la configuration de la période de rétention, veuillez consulter [the section called "Configuration des environnements de travail" \(p. 527\)](#).

### Note

L'option Elastic Beanstalk `MaxRetries` est équivalente à l'option SQS `MaxReceiveCount`. Si votre environnement de travail n'utilise pas une file d'attente SQS générée automatiquement, utilisez l'option `MaxReceiveCount` dans SQS pour désactiver efficacement votre file d'attente de lettres mortes. Pour de plus amples informations, veuillez consulter [Utilisation des files d'attente de lettres mortes Amazon SQS](#).

Pour de plus amples informations sur le cycle de vie d'un message SQS, veuillez consulter [Cycle de vie des messages](#).

## Tâches périodiques

Vous pouvez définir des tâches périodiques dans un fichier nommé `cron.yaml` dans votre groupe source pour ajouter des tâches à la file d'attente de votre environnement de travail automatiquement à intervalles réguliers.

Par exemple, le fichier `cron.yaml` suivant crée deux tâches périodiques. La première s'exécute toutes les 12 heures et la seconde à 23 h UTC tous les jours.

### Example cron.yaml

```
version: 1
cron:
  - name: "backup-job"
    url: "/backup"
    schedule: "0 */12 * * *"
  - name: "audit"
    url: "/audit"
    schedule: "0 23 * * *"
```

Le **name** doit être unique pour chaque tâche. L'URL est le chemin d'accès auquel la demande POST est envoyée pour déclencher la tâche. Le programme est une [expression CRON](#) qui détermine le moment d'exécution de la tâche.

Lorsqu'une tâche s'exécute, le démon publie un message dans la file d'attente SQS de l'environnement avec un en-tête indiquant la tâche qui doit être effectuée. Toute instance de l'environnement peut prélever le message et traiter la tâche.

Note

Si vous configurez votre environnement de travail avec une file d'attente SQS et que vous choisissez une [file d'attente FIFO Amazon SQS](#), les tâches périodiques ne sont pas pris en charge.

Elastic Beanstalk utilise le choix principal pour déterminer quelle instance dans votre environnement de travail met en attente la tâche périodique. Chaque instance tente de devenir leader en écrivant dans une table Amazon DynamoDB. La première instance qui réussit est la principale et elle doit continuer à écrire dans la table pour conserver l'état de principale. Si la principale devient hors service, une autre instance prend rapidement sa place.

Pour les tâches périodiques, le démon de travail définit les en-têtes supplémentaires suivants.

En-têtes HTTP

Nom	Valeur
X-Aws-Sqsdk-Taskname	Pour les tâches périodiques, le nom de la tâche à effectuer.
X-Aws-Sqsdk-Scheduled-At	Heure à laquelle la tâche périodique a été planifiée
X-Aws-Sqsdk-Sender-Id	Numéro de compte AWS de l'expéditeur du message

## Utilisation d'Amazon CloudWatch pour la mise à l'échelle automatique dans les niveaux d'environnement de travail

Ensemble, Amazon EC2 Auto Scaling et CloudWatch surveillent l'utilisation de l'UC des instances en cours d'exécution dans l'environnement de travail. La manière dont vous configurez la limite de mise à l'échelle automatique pour la capacité de l'UC détermine le nombre d'instances que le groupe Auto Scaling exécute pour gérer adéquatement le débit de messages dans la file d'attente Amazon SQS. Chaque instance EC2 publie ses métriques d'utilisation de l'UC dans CloudWatch. Amazon EC2 Auto Scaling récupère à partir de CloudWatch l'utilisation moyenne de l'UC sur toutes les instances dans l'environnement de travail. Vous configurez les seuils supérieur et inférieur, ainsi que le nombre d'instances à ajouter ou à résilier, selon la capacité de l'UC. Quand Amazon EC2 Auto Scaling détecte que vous avez atteint le seuil supérieur spécifié sur la capacité de l'UC, Elastic Beanstalk crée les instances dans l'environnement de travail. Les instances sont supprimées quand la charge de l'UC est inférieure au seuil.

Note

Les messages qui n'ont pas été traités au moment de la résiliation d'une instance sont renvoyés à la file d'attente où ils peuvent être traités par un autre démon sur une instance qui est encore en cours d'exécution.

Vous pouvez également configurer d'autres alarmes CloudWatch, en fonction des besoins, à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande ou du fichier d'options. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon CloudWatch \(p. 894\)](#) et [Création d'un groupe Auto Scaling avec des stratégies de mise à l'échelle par étape](#).

## Configuration des environnements de travail

Vous pouvez gérer la configuration d'un environnement de travail en modifiant la catégorie Worker (Environnement de travail) sur la page Configuration dans la [console de gestion de l'environnement \(p. 429\)](#).

## Modify worker

You can create a new Amazon SQS queue for your worker application or pull work items from an existing queue. The worker daemon instances in your environment pulls an item from the queue and relays it in the body of a POST request to a local HTTP path relative to the environment.

### Queue

#### Worker queue

Autogenerated queue



SQS queue from which to read work items.

### Messages

#### HTTP path

/

The daemon pulls items from the Amazon SQS queue and posts them locally to this path.

#### MIME type

application/json



Change the MIME type of the POST requests that the worker daemon sends to your application.

#### HTTP connections

50

Maximum number of concurrent connections to the application.

#### Visibility timeout

300

seconds

The amount of time to lock an incoming message for processing before returning it to the queue.

#### Error visibility timeout

seconds

The amount of time to wait before resending a message after an error response from the application.

### ▼ Advanced options

The following settings control advanced behavior of the worker tier daemon. [Learn more](#)

#### Max retries

10

528

Maximum number of retries after which the message is discarded.

#### Connection timeout

### Note

Vous pouvez configurer le chemin d'URL pour la publication des messages de file d'attente de travail, mais vous ne pouvez pas configurer le port IP. Elastic Beanstalk publie toujours les messages de file d'attente de travail sur le port 80. L'application de l'environnement de travail ou son proxy doit être à l'écoute sur le port 80.

### Pour configurer le démon de travail

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration de l'Worker (Environnement de travail, choisissez Edit (Modifier).

La page de configuration Modify worker (Modifier l'environnement de travail) propose les options suivantes.

#### Dans la section Queue (File d'attente) :

- Worker queue (File d'attente de travail) : spécifiez la file d'attente Amazon SQS à partir de laquelle le démon lit. Vous pouvez choisir une file d'attente existante, si vous en avez une. Si vous choisissez Autogenerated queue (File d'attente générée automatiquement), Elastic Beanstalk crée une nouvelle file d'attente Amazon SQS et une Worker queue URL (URL de file d'attente de travail) correspondante.

### Note

Lorsque vous choisissez File d'attente générée automatiquement, la file d'attente créée par Elastic Beanstalk est une file d'attente Amazon SQS [standard](#). Lorsque vous choisissez une file d'attente existante, vous pouvez fournir une file standard ou une file d'attente Amazon SQS [FIFO](#). Sachez que si vous spécifiez une file d'attente FIFO, les [tâches périodiques \(p. 525\)](#) ne sont pas prises en charge.

- Worker queue URL (URL de la file d'attente de travail) : si vous choisissez une Worker queue (File d'attente de travail) existante, ce paramètre affiche l'URL associée à cette file d'attente Amazon SQS.

#### Dans la section Messages :

- HTTP path (Chemin HTTP) : spécifiez le chemin d'accès relatif à l'application qui reçoit les données à partir de la file d'attente Amazon SQS. Les données sont insérées dans le corps du message d'un message HTTP POST. La valeur par défaut est /.
- MIME type (Type MIME) : indiquez le type MIME que le message HTTP POST utilise. La valeur par défaut est [application/json](#). Cependant, n'importe quelle valeur est valide car vous pouvez créer, puis spécifier votre propre type MIME.
- HTTP Connections (Connexions HTTP) : spécifiez le nombre maximal de connexions simultanées que le démon peut réaliser sur toute application au sein d'une instance Amazon EC2. La valeur par défaut est **50**. Vous pouvez spécifier une valeur de **1 à 100**.
- Visibility timeout (Délai de visibilité) : indiquez la durée, en secondes, pendant laquelle un message entrant à partir de la file d'attente Amazon SQS est verrouillé pour traitement. Une fois que la durée configurée est écoulée, le message est à nouveau rendu visible dans la file d'attente permettant à un autre démon de le lire. Choisissez une valeur supérieure au délai estimé nécessaire à votre application pour traiter des messages, jusqu'à **43200** secondes.

- Error visibility timeout (Délai de visibilité de l'erreur) : indiquez la durée, en secondes, qui s'écoule avant qu'Elastic Beanstalk renvoie un message à la file d'attente Amazon SQS après qu'une tentative de le traiter a échoué avec une erreur explicite. Vous pouvez spécifier une valeur de **0** à **43200** secondes.

Dans la section Advanced options (Options avancées) :

- Max retries (Nombre maximal de nouvelles tentatives) : spécifiez le nombre maximal de fois qu'Elastic Beanstalk tente d'envoyer le message à la file d'attente Amazon SQS avant de déplacer le message dans la [file d'attente des lettres mortes \(p. 525\)](#). La valeur par défaut est **10**. Vous pouvez spécifier une valeur de **1** à **100**.

#### Note

L'option Max retries (Nombre maximum de nouvelles tentatives) ne s'applique pas si les files d'attente Amazon SQS de votre environnement de travail ne disposent pas d'une file d'attente de lettres mortes configurée. Dans ce cas, Amazon SQS conserve les messages dans la file d'attente et les traite jusqu'à ce que la période spécifiée par l'option Retention period (Période de conservation) expire.

- Connection timeout (Délai de connexion) : indiquez la durée, en secondes, d'attente de connexions réussies à une application. La valeur par défaut est **5**. Vous pouvez spécifier une valeur de **1** à **60** secondes.
- Inactivity timeout (Délai d'inactivité) : indiquez la durée, en secondes, d'attente d'une réponse sur une connexion existante à une application. La valeur par défaut est **180**. Vous pouvez spécifier une valeur de **1** à **36000** secondes.
- Retention period (Période de conservation) : indiquez la durée, en secondes, de validité et de traitement actif d'un message. La valeur par défaut est **345600**. Vous pouvez spécifier une valeur de **60** à **1209600** secondes.

Si vous utilisez une file d'attente Amazon SQS existante, les paramètres que vous configurez lorsque vous créez un environnement de travail peuvent entrer en conflit avec les paramètres que vous avez configurés directement dans Amazon SQS. Par exemple, si vous configurez un environnement de travail avec une valeur `RetentionPeriod` qui est supérieure à la valeur `MessageRetentionPeriod` que vous avez définie dans Amazon SQS, Amazon SQS supprime le message quand il dépasse la valeur `MessageRetentionPeriod`.

À l'inverse, si la valeur `RetentionPeriod` que vous configurez dans les paramètres d'environnement de travail est inférieure à la valeur `MessageRetentionPeriod` que vous définissez dans Amazon SQS, alors le démon supprime le message avant qu'Amazon SQS puisse le faire. Pour `VisibilityTimeout`, la valeur que vous configurez pour le démon dans les paramètres d'environnement de travail remplace la valeur Amazon SQS `VisibilityTimeout`. Assurez-vous que les messages sont supprimés correctement en comparant vos paramètres Elastic Beanstalk à vos paramètres Amazon SQS.

## Création de liens entre les environnements Elastic Beanstalk

A mesure que votre application se développe en taille et en complexité, vous pourrez souhaiter la diviser en composants ayant différents cycle de vie opérationnels et de développement. En exécutant des services plus petits qui interagissent les uns avec les autres via une interface bien définie, les équipes peuvent travailler de manière indépendante et réduire les risques lors des déploiements. AWS Elastic Beanstalk vous permet de lier vos environnements pour partager des informations entre les composants qui dépendent les uns des autres.

### Note

Elastic Beanstalk prend en charge actuellement des liens d'environnement pour toutes les plateformes, sauf Multicontainer Docker.

Les liens d'environnement vous permettent de spécifier les connexions entre les environnements de composants de votre application en tant que références désignées. Lorsque vous créez un environnement qui définit un lien, Elastic Beanstalk définit une variable d'environnement portant le même nom que le lien. La valeur de la variable est le point de terminaison que vous pouvez utiliser pour vous connecter à l'autre composant qui peut être un environnement de travail ou de serveur web.

Par exemple, si votre application est constituée d'un serveur frontal qui collecte des adresses e-mail et d'un serveur de travail qui envoie un e-mail de bienvenue aux adresses de messagerie collectées par le serveur frontal, vous pouvez créer un lien vers le serveur de travail dans votre serveur frontal et permettre à ce dernier de découvrir automatiquement le point de terminaison (URL de file d'attente) pour votre serveur de travail.

Définissez les liens vers d'autres environnements dans un [manifeste d'environnement \(p. 785\)](#), un fichier formaté YAML nommé `env.yaml` à la racine de la source de votre application. Le manifeste suivant définit un lien vers un serveur de travail nommé de l'environnement :

```
~/workspace/my-app/frontend/env.yaml
```

```
AWSConfigurationTemplateVersion: 1.1.0.0
EnvironmentLinks:
  "WORKERQUEUE": "worker"
```

Lorsque vous créez un environnement avec une version de l'application qui inclut le manifeste d'environnement ci-dessus, Elastic Beanstalk recherche un environnement nommé `worker` qui appartient à la même application. Si cet environnement existe, Elastic Beanstalk crée une propriété d'environnement nommée `WORKERQUEUE`. La valeur de `WORKERQUEUE` est l'URL de file d'attente Amazon SQS. L'application frontale peut lire cette propriété de la même manière qu'une variable d'environnement. Pour de plus amples informations, veuillez consulter [Manifeste d'environnement \(env.yaml\) \(p. 785\)](#).

Pour utiliser des liens d'environnement, ajoutez un manifeste d'environnement à la source de votre application et téléchargez-le avec l'interface de ligne de commande EB, l'AWS CLI ou un kit SDK. Si vous utilisez l'AWS CLI ou un SDK, définissez l'indicateur `process` lorsque vous appelez `CreateApplicationVersion` :

```
$ aws elasticbeanstalk create-application-version --process --application-name my-app
--version-label frontend-v1 --source-bundle S3Bucket="DOC-EXAMPLE-BUCKET",S3Key="front-
v1.zip"
```

Cette option indique à Elastic Beanstalk de valider les fichiers de configuration et le manifeste d'environnement de votre groupe source lorsque vous créez la version d'application. Les ensembles de l'interface de ligne de commande EB définissent automatiquement cet indicateur lorsque vous avez un manifeste d'environnement dans le répertoire de votre projet.

Créez vos environnements normalement à l'aide de n'importe quel client. Lorsque vous avez besoin de mettre fin à des environnements, résiliez l'environnement avec le lien en premier. Si un environnement est lié à un autre environnement, Elastic Beanstalk empêche l'arrêt de l'environnement lié. Pour remplacer cette protection, utilisez l'indicateur `ForceTerminate`. Ce paramètre est disponible dans l'AWS CLI en tant que `--force-terminate` :

```
$ aws elasticbeanstalk terminate-environment --force-terminate --environment-name worker
```

# Configuration d'environnements Elastic Beanstalk

AWS Elastic Beanstalk propose un large éventail d'options pour personnaliser les ressources de votre environnement, ainsi que le comportement d'Elastic Beanstalk et les paramètres de plateforme. Lorsque vous créez un environnement de serveur web, Elastic Beanstalk crée plusieurs ressources pour prendre en charge le fonctionnement de votre application.

- Instance EC2 – Machine virtuelle Amazon Elastic Compute Cloud (Amazon EC2) configurée pour exécuter des applications web sur la plateforme de votre choix.

Chaque plateforme exécute un ensemble spécifique de logiciels, de fichiers de configuration et de scripts pour prendre en charge une version de langage, une infrastructure ou un conteneur web spécifiques, ou une combinaison de ces éléments. La plupart des plateformes utilisent Apache ou nginx comme proxy inverse situé devant votre application web, qui lui transmet des demandes, traite des ressources statiques et génère des journaux d'accès et d'erreur.

## Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021.

Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification.

Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

- Groupe de sécurité de l'instance – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant de l'équilibreur de charge à atteindre l'instance EC2 qui exécute votre application web. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- ÉquilibrEUR de charge – ÉquilibrEUR de charge Elastic Load Balancing configuré pour répartir les demandes vers les instances exécutant votre application. De plus, l'équilibrEUR de charge vous évite d'exposer directement vos instances sur Internet.
- Groupe de sécurité de l'équilibrEUR de charge – Groupe de sécurité Amazon EC2 configuré pour autoriser le trafic entrant sur le port 80. Cette ressource autorise le trafic HTTP provenant d'Internet à atteindre l'équilibrEUR de charge. Par défaut, le trafic n'est pas autorisé sur les autres ports.
- Groupe Auto Scaling – Groupe Auto Scaling configuré pour remplacer une instance si elle est résiliée ou devient indisponible.
- Compartiment Amazon S3 – Emplacement de stockage pour votre code source, les journaux et autres artefacts qui sont créés lorsque vous utilisez Elastic Beanstalk.
- Alarmes Amazon CloudWatch – Deux alarmes CloudWatch qui contrôlent la charge sur les instances de votre environnement et se déclenchent si la charge est trop élevée ou trop faible. Lorsqu'une alarme est déclenchée, votre groupe Auto Scaling s'adapte en fonction, à la hausse ou à la baisse.

- Pile AWS CloudFormation – Elastic Beanstalk utilise AWS CloudFormation pour lancer les ressources dans votre environnement et pour propager les modifications de configuration. Les ressources sont définies dans un modèle, que vous pouvez afficher dans la [console AWS CloudFormation](#).
- Nom de domaine – Nom de domaine qui permet d'accéder à votre application web sous la forme **sous-domaine.région.elasticbeanstalk.com**.

Cette rubrique se concentre sur les options de configuration des ressources disponibles dans la console Elastic Beanstalk. Les rubriques suivantes montrent comment configurer votre environnement dans la console. Elles décrivent également les espaces de noms sous-jacents qui correspondent aux options de la console à utiliser avec les fichiers de configuration ou les options de configuration d'API. Pour en savoir plus sur les méthodes de configuration avancées, consultez [Configuration d'environnements \(niveau avancé\) \(p. 658\)](#).

#### Rubriques

- [Configuration d'un environnement avec la console Elastic Beanstalk. \(p. 533\)](#)
- [Instances Amazon EC2 de votre environnement Elastic Beanstalk \(p. 538\)](#)
- [Groupe Auto Scaling pour votre environnement Elastic Beanstalk \(p. 547\)](#)
- [Équilibrer de charge pour votre environnement Elastic Beanstalk \(p. 562\)](#)
- [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#)
- [Sécurité de votre environnement AWS Elastic Beanstalk \(p. 626\)](#)
- [Balisage des ressources dans vos environnements Elastic Beanstalk \(p. 629\)](#)
- [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#)
- [Notifications d'environnement Elastic Beanstalk avec Amazon SNS \(p. 644\)](#)
- [Configuration d'Amazon Virtual Private Cloud \(Amazon VPC\) avec Elastic Beanstalk \(p. 649\)](#)
- [Nom de domaine de votre environnement Elastic Beanstalk \(p. 656\)](#)

## Configuration d'un environnement avec la console Elastic Beanstalk.

Vous pouvez utiliser la console Elastic Beanstalk pour consulter et modifier de nombreuses [options de configuration \(p. 658\)](#) de votre environnement et ses ressources. Vous pouvez personnaliser le comportement de votre environnement au cours des déploiements, activer des fonctions supplémentaires et modifier le type d'instance ainsi que d'autres paramètres que vous avez choisis lors de la création de l'environnement.

Pour afficher un récapitulatif de la configuration de votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.

## Page de présentation de la configuration

La page Configuration overview (Présentation de la configuration) affiche un ensemble de catégories de configuration. Chaque catégorie de configuration récapitule l'état actuel d'un groupe d'options connexes.

Vous avez le choix entre deux vues pour cette page. Activez la Table View (Vue Table) pour afficher une liste d'options regroupées par catégorie.

Configuration overview		Cancel	Re
Category	Options		
Software	Environment properties: GRADLE_HOME, JAVA_HOME, M2, M2_HOME, PORT Log streaming: disabled Rotate logs: disabled X-Ray daemon: disabled		
Instances	EC2 security groups: awseb-e-m5yhyre5nuj-stack-AWSEBSecurityGroup-12122MOSK IMDSv1: disabled IOPS: container default Monitoring interval: 5 minute Root volume type: container default Size: container default Throughput: container default		
Capacity	AMI ID: ami-04d85997c65fd307d Availability Zones: Any Breach duration: 5 Environment type: load balancing, auto scaling Instance type: t2.micro Lower threshold: 200000 Max: 4 Metric: NetworkOut Min: 1 Period: 5 Placement: Scale down increment: -1 Scale up increment: 1 Scaling cooldown: 360 seconds Statistic: Average Unit: Bytes		

Vous pouvez rechercher une option par son nom ou sa valeur en saisissant les termes de recherche dans une zone de recherche. À mesure que vous saisissez du texte, la liste se réduit et affiche uniquement les options qui correspondent à vos termes de recherche.

The screenshot shows the 'Configuration overview' page in the AWS Elastic Beanstalk developer guide. At the top right are 'Cancel' and 'Review changes' buttons. A search bar contains the text 'log'. Below is a table with two columns: 'Category' and 'Options'.

Category	Options
Software	Lifecycle: Keep <b>logs</b> after terminating environment Rotate <b>logs</b> : disabled <b>Log</b> streaming: disabled
Monitoring	<b>Log</b> group: <b>logStream:group=/aws/elasticbeanstalk/Node-env/environment-health.log</b> <a href="#">target="_blank"&gt;/aws/elasticbeanstalk/Node-env/environment-health.log</a>

Activez la Table View (Vue en tableau) pour afficher chaque catégorie dans une section distincte (carte de configuration).

## Configuration overview

<h3>Software</h3> <p>Environment properties: GRADLE_HOME, JAVA_HOME, M2, M2_HOME, PORT Log streaming: disabled Rotate logs: disabled X-Ray daemon: disabled</p> <p><input type="button" value="Edit"/></p>	<h3>Instances</h3> <p>EC2 security groups: awseb-e-m5yhare5nuj-stack-AWSEBSecurityGroup-12122MOSKFTC4 IMDSv1: disabled IOPS: container default Monitoring interval: 5 minute Root volume type: container default Size: container default Throughput: container default</p> <p><input type="button" value="Edit"/></p>	<h3>Load balancer</h3> <p>Listeners: 1 Load balancer type: application Processes: 1 Rules: 0 Shared: false Store logs: disabled</p> <p><input type="button" value="Edit"/></p>	<h3>Rolling updates and deployments</h3> <p>Batch size: 100% Command timeout: 600 Deployment policy: All at once Healthy threshold: Ok Ignore health check: disabled Rolling updates: disabled</p> <p><input type="button" value="Edit"/></p>
--	---	--	---

Choisissez Edit (Modifier) dans une des catégories de configuration pour accéder à la page de configuration correspondante, depuis laquelle vous pouvez consulter toutes les valeurs d'option et apporter des modifications. Une fois que vous avez consulté et modifié les options, vous pouvez choisir l'une des actions suivantes :

- Cancel (Annuler) : cette option vous redirige vers le tableau de bord de l'environnement sans appliquer les modifications de configuration. Lorsque vous choisissez Annuler, toutes les modifications en attente apportées sur les catégories de configuration sont perdues.

Vous pouvez également annuler vos modifications de configuration en choisissant une autre page de la console, comme Tableau de bord ou Journaux. Dans ce cas, s'il existe des modifications de configuration en attente, la console vous invitera à confirmer que vous acceptez de les perdre.

- Review changes (Vérifier les modifications) : cette option vous permet d'obtenir un résumé de toutes les modifications en attente que vous avez effectuées dans les catégories de configuration. Pour plus d'informations, consultez [Page de vérification des modifications \(p. 537\)](#).
- Apply configuration (Appliquer une configuration) : cette option vous permet d'appliquer à votre environnement les modifications que vous avez effectuées dans les catégories de configuration. Dans certains cas, vous êtes invité à confirmer une conséquence de l'une de vos décisions de configuration.

## Page de vérification des modifications

La page Review Changes (Vérifier les modifications) affiche un tableau répertoriant toutes les modifications d'options que vous avez effectuées dans les catégories de configuration et que vous n'avez pas encore appliquées à l'environnement. Si vous avez supprimé des options (par exemple, une propriété d'un environnement personnalisé), un deuxième tableau indique les options supprimées.

Les deux tableaux répertorient chaque option en combinant les valeurs Namespace (Espace de noms) et Option Name (Nom de l'option) qu'Elastic Beanstalk utilise pour identifier les options en question. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

Voici quelques exemples de modifications de configuration que vous pouvez effectuer :

- Dans la catégorie Capacité, remplacez la valeur 1 de l'option Instances (Min) par 2 et la valeur 2 de l'option Instances (Max) par 4. Cette modification correspond à deux modifications dans l'espace de noms `aws:autoscaling:asg` de la liste des options modifiées.
- Dans la catégorie Logiciels :
  - Activez l'option Effectuer une rotation des journaux. Cette modification correspond à une modification dans l'espace de noms `aws:elasticbeanstalk:hostmanager` de la liste des options modifiées.
  - Supprimez la propriété d'environnement `MY_ENV_PROPERTY`. Cette modification correspond à une seule entrée pour l'espace de noms `aws:elasticbeanstalk:application:environment` de la liste des options supprimées.
- Dans la catégorie Mises à jour gérées, activez l'option Mises à jour gérées. À elle seule, cette modification de configuration correspond à trois modifications d'option sur deux espaces de noms. Ces modifications sont représentées par les trois derniers éléments affichés dans la liste Changed Options (Options modifiées).

L'image suivante illustre les listes des modifications de configuration sur la page Review Changes (Vérifier les modifications).

Review Changes

Continue

Changed Options

Namespace ▾	Option Name	Old Value	New Value
aws:autoscaling:asg	MaxSize	2	4
aws:autoscaling:asg	MinSize	1	2
aws:elasticbeanstalk:hostmanager	LogPublicationControl	false	true
aws:elasticbeanstalk:managedactions	ManagedActionsEnabled	false	true
aws:elasticbeanstalk:managedactions	PreferredStartTime		WED:12:00
aws:elasticbeanstalk:managedactions :platformupdate	UpdateLevel		minor

Removed Options

Namespace	Option Name
aws:elasticbeanstalk:application:environment	MY_ENV_PROPERTY

Une fois que vous avez vérifié les modifications, vous pouvez choisir l'une des actions suivantes :

- Continue (Continuer) : cette option vous redirige vers la page Configuration overview (Présentation de la configuration). Vous pouvez alors continuer à apporter des modifications ou appliquer celles qui sont en attente.
- Apply configuration (Appliquer une configuration) : cette option vous permet d'appliquer à votre environnement les modifications que vous avez effectuées dans les catégories de configuration. Dans certains cas, vous êtes invité à confirmer une conséquence de l'une de vos décisions de configuration.

## Instances Amazon EC2 de votre environnement Elastic Beanstalk

Lorsque vous créez un environnement de serveur web, AWS Elastic Beanstalk crée une ou plusieurs machines virtuelles Amazon Elastic Compute Cloud (Amazon EC2), appelées Instances.

Les instances de votre environnement sont configurées pour exécuter des applications Web sur la plateforme que vous choisissez. Vous pouvez apporter des modifications aux propriétés et aux comportements des instances de votre environnement lorsque vous créez votre environnement, ou même pendant son exécution. Vous pouvez également déjà effectuer ces modifications en modifiant le code source que vous déployez dans l'environnement. Pour plus d'informations, consultez [the section called "Options de configuration" \(p. 658\)](#).

#### Note

Le [groupe Auto Scaling \(p. 547\)](#) de votre environnement gère les instances Amazon EC2 qui exécutent votre application. Lorsque vous apportez les modifications de configuration décrites sur cette page, la configuration de lancement (un modèle de lancement Amazon EC2 ou une ressource de configuration de lancement de groupe Auto Scaling) change. Cette modification nécessite le [remplacement de toutes les instances \(p. 488\)](#) et déclenche une [mise à jour continue \(p. 489\)](#) ou [immuable \(p. 493\)](#), selon la configuration.

Lorsque vous créez un nouvel environnement, vous choisissez un type d'instance pour déterminer le matériel de l'ordinateur hôte utilisé pour exécuter vos instances. Elastic Beanstalk prend en charge les types d'instance compatibles après leur introduction par Amazon EC2. Pour plus d'informations sur les types d'instance disponibles, consultez [Types d'instance](#) dans le Guide de l'utilisateur Amazon EC2 pour instances Linux ou [Types d'instance](#) dans le Guide de l'utilisateur Amazon EC2 pour instances Windows.

#### Note

Elastic Beanstalk ne prend pas en charge les types d'instance Amazon EC2 Mac et Amazon EC2 Graviton (basées sur ARM) pour le moment.

Elastic Beanstalk prend en charge plusieurs [options d'achat d'instances](#)Amazon EC2 : instances à la demande, instances réservées et instances Spot. Une instance à la demande est une ressource utilisant un modèle de paiement à l'utilisation. Aucun engagement à long terme n'est requis lorsque vous l'utilisez. Une instance réservée est une remise de facturation prépayée appliquée automatiquement aux instances à la demande correspondantes dans votre environnement. Une instance Spot est une instance Amazon EC2 non utilisée qui est disponible à un prix inférieur au prix à la demande. Vous pouvez activer les instances Spot dans votre environnement en définissant une seule option. Vous pouvez configurer l'utilisation des instances Spot, y compris la combinaison d'instances à la demande et d'instances Spot, à l'aide d'options supplémentaires. Pour plus d'informations, consultez [Groupe Auto Scaling \(p. 547\)](#).

#### Sections

- [Configuration des instances Amazon EC2 de votre environnement \(p. 539\)](#)
- [Espace de noms aws:autoscaling:launchconfiguration \(p. 544\)](#)
- [Configuration du service de métadonnées d'instance sur les instances de votre environnement \(p. 545\)](#)

## Configuration des instances Amazon EC2 de votre environnement

Vous pouvez modifier la configuration d'instance Amazon EC2 de votre environnement Elastic Beanstalk dans la console Elastic Beanstalk.

Pour configurer des instances Amazon EC2 dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.

4. Dans la catégorie de configuration Instances, choisissez Edit (Modifier). Modifiez les paramètres de cette catégorie, puis choisissez Apply (Appliquer). Pour les descriptions de paramètre, consultez la section [the section called “Paramètres de catégorie d’instances” \(p. 540\)](#) de cette page.
5. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier). Modifiez les paramètres de cette catégorie, puis choisissez Continue (Continuer). Pour les descriptions de paramètre, consultez la section [the section called “Paramètres de catégorie de capacité” \(p. 543\)](#) de cette page.

## Paramètres de catégorie d'instances

Les paramètres suivants relatifs aux instances Amazon EC2 sont disponibles dans la catégorie de configuration Instances.

### Options

- [Intervalle de surveillance \(p. 542\)](#)
- [Volume racine \(périphérique de démarrage\) \(p. 542\)](#)
- [Service des métadonnées d’instance \(p. 542\)](#)
- [Groupes de sécurité \(p. 542\)](#)

## Modify instances

### Amazon CloudWatch monitoring

The time interval between when metrics are reported from the EC2 instances.

#### Monitoring interval

5 minute



### Root volume (boot device)

#### Root volume type

(Container default)



#### Size

The number of gigabytes of the root volume attached to each instance.

100 GB

#### IOPS

Input/output operations per second for a provisioned IOPS (SSD) volume.

100 IOPS

#### Throughput

The desired throughput to provision for the Amazon EBS root volume attached to your environment's EC2 instance

1 MiB/s

### Instance metadata service (IMDS)

Your environment's platform supports both IMDSv1 and IMDSv2. To enforce IMDSv2, disable IMDSv1. [Learn more](#)

#### Disable IMDSv1

With the current setting, the environment enables both IMDSv1 and IMDSv2.

Disabled

### EC2 security groups

Group name
<input type="checkbox"/> awseb-e-awppgphwta-stack-AWSEBLoadBalancerSecurityGroup-LUAOUHKL3SNI
<input type="checkbox"/> awseb-e-awppgphwta-stack-AWSEBSecurityGroup-109O5QLX6UCC
<input type="checkbox"/> awseb-e-awppgphwta-stack-AWSEBSecurityGroup-541

## Intervalle de surveillance

Par défaut, les instances dans votre environnement publient des [métriques d'état de base \(p. 834\)](#) à Amazon CloudWatch à des intervalles de 5 minutes sans coût supplémentaire.

Pour obtenir des rapports plus détaillés, vous pouvez définir Monitoring interval (Intervalle de surveillance) sur 1 minute afin d'augmenter la fréquence à laquelle les ressources dans votre environnement publient les [métriques d'état de base \(p. 836\)](#) sur CloudWatch. Les frais de service CloudWatch s'appliquent aux métriques d'intervalle d'une minute. Pour plus d'informations, consultez [Amazon CloudWatch](#).

## Volume racine (périphérique de démarrage)

Chaque instance dans votre environnement est configurée avec un volume racine. Le volume racine est le périphérique de stockage en mode bloc Amazon EBS attaché à l'instance pour stocker le système d'exploitation, les bibliothèques, les scripts et le code source de votre application. Par défaut, toutes les plateformes utilisent des périphériques de stockage en mode bloc à usage général (SSD) pour le stockage.

Vous pouvez modifier Type de volume racine pour utiliser les types de volumes d'IOPS provisionnées (SSD) ou de stockage magnétique et, si nécessaire, accroître la taille du volume. Pour les volumes d'IOPS provisionnées, vous devez également sélectionner le nombre d'IOPS à approvisionner. Débit est uniquement applicable aux types de volumes SSD gp3. Vous pouvez saisir le débit souhaité à approvisionner. Il peut varier entre 125 et 1 000 megiocets par seconde (MiB/s). Sélectionnez le type de volume qui répond à vos exigences de prix et de performance.

Pour de plus amples informations, veuillez consulter [Types de volume Amazon EBS](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux et [Détails du produit Amazon EBS](#).

## Service des métadonnées d'instance

Le service de métadonnées d'instance (IMDS) est un composant sur instance utilisé par le code sur l'instance pour accéder en toute sécurité aux métadonnées d'instance. Le code peut accéder aux métadonnées d'instance à partir d'une instance en cours d'exécution à l'aide de l'une des deux méthodes suivantes : Instance Metadata Service Version 1 (IMDSv1) ou Instance Metadata Service Version 2 (IMDSv2). IMDSv2 est plus sécurisé. Désactivez IMDSv1 pour appliquer IMDSv2. Pour de plus amples informations, veuillez consulter [the section called "IMDS" \(p. 545\)](#).

### Note

La section IMDS de cette page de configuration n'apparaît que pour les versions de plateforme prenant en charge IMDSv2.

## Groupes de sécurité

Les groupes de sécurité attachés à vos instances déterminent quel trafic est autorisé à atteindre les instances, et quel trafic est autorisé à quitter les instances. Elastic Beanstalk crée un groupe de sécurité qui autorise le trafic à partir de l'équilibrEUR de charge sur les ports standards pour HTTP (80) et HTTPS (443).

Vous pouvez spécifier des groupes de sécurité supplémentaires que vous avez créés pour autoriser le trafic sur les autres ports ou depuis d'autres sources. Par exemple, vous pouvez créer un groupe de sécurité pour l'accès SSH qui autorise le trafic entrant sur le port 22 à partir d'une plage d'adresses IP limitée ou, pour plus de sécurité, à partir d'un hôte bastion auquel vous seul avez accès.

### Note

Pour autoriser le trafic entre les instances de l'environnement A et celles de l'environnement B, vous pouvez ajouter une règle au groupe de sécurité qu'Elastic Beanstalk a attaché à l'environnement B. Puis, vous pouvez spécifier le groupe de sécurité qu'Elastic Beanstalk a attaché à l'environnement A. Ceci permet le trafic entrant dans/le trafic sortant depuis les instances de l'environnement A. Toutefois, cette opération crée une dépendance entre les deux groupes de sécurité. Si vous essayez de résilier ultérieurement l'environnement A,

Elastic Beanstalk ne sera pas en mesure de supprimer le groupe de sécurité de l'environnement, car le groupe de sécurité de l'environnement B en est dépendant.

Par conséquent, nous vous conseillons de plutôt créer un groupe de sécurité distinct, l'attacher à l'environnement A et le spécifier dans une règle du groupe de sécurité de l'environnement B.

Pour de plus amples informations sur les groupes de sécurité Amazon EC2, consultez [Groupes de sécurité Amazon EC2](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

## Paramètres de catégorie de capacité

Les paramètres suivants relatifs aux instances Amazon EC2 sont disponibles dans la catégorie de configuration Capacity (Capacité).

### Options

- [Type d'instance \(p. 543\)](#)
- [ID d'AMI \(p. 544\)](#)

The screenshot shows the 'Auto Scaling Group' configuration section. It includes a visual representation of 15 yellow squares representing instances, followed by dropdown menus for 'Instance type' (set to 't2.micro') and 'AMI ID' (set to 'ami-0c0f37f83da1542ca').

## Type d'instance

Le paramètre Instance type (Type d'instance) détermine le type d'instance Amazon EC2 lancée pour exécuter votre application. Choisissez une instance qui soit suffisamment puissante pour exécuter votre application en charge, mais pas puissante au point d'être inactive la plupart du temps. À des fins de développement, la famille t2 d'instances fournit une quantité modérée de puissance avec la possibilité de rafales sur de courtes périodes.

À grande échelle, les applications à haut niveau de disponibilité utilisent un groupe d'instances afin de garantir que cette capacité ne soit pas considérablement affectée si une seule instance s'arrête. Commencez par un type d'instance qui vous permet d'exécuter cinq instances sous des charges modérées pendant les heures normales. En cas de défaillance d'une instance, les instances restantes peuvent absorber le reste du trafic. Le tampon de capacité laisse également du temps pour que l'environnement s'adapte à mesure que le trafic commence à augmenter pendant les heures de pointe.

Elastic Beanstalk ne prend pas en charge les types d'instance Amazon EC2 Mac et Amazon EC2 Graviton (basées sur ARM) pour le moment. Pour de plus amples informations sur les familles d'instance et types d'instance Amazon EC2, consultez [Types d'instance](#) dans le Guide de l'utilisateur Amazon EC2 pour

les instances Linux ou [Types d'instance](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows. Pour déterminer les types d'instance qui répondent à vos exigences et leurs régions prises en charge, consultez [Types d'instance disponibles](#) dans le Guide de l'utilisateur Amazon EC2 pour instances Linux ou [Types d'instance disponibles](#) dans le Guide de l'utilisateur Amazon EC2 pour instances Windows.

Lorsque vous activez les demandes de parc d'instances Spot pour votre environnement, cette page de configuration affiche une liste de Instance types (Types d'instances) au lieu d'un paramètre unique. Vous pouvez sélectionner un ou plusieurs types d'instance pour vos instances Spot. Pour de plus amples informations, veuillez consulter [the section called "Prise en charge d'une instance Spot" \(p. 548\)](#).

## ID d'AMI

L'Amazon Machine Image (AMI) est l'image de machine Amazon Linux ou Windows Server qu'Elastic Beanstalk utilise pour lancer les instances Amazon EC2 dans votre environnement. Elastic Beanstalk fournit des images de machine contenant les outils et les ressources requis pour exécuter votre application.

Elastic Beanstalk sélectionne une valeur AMI par défaut pour votre environnement en fonction de la région, de la plateforme et du type d'instance que vous choisissez. Si vous avez créé une [AMI personnalisée \(p. 787\)](#), remplacez l'ID AMI par défaut par le vôtre.

## Espace de noms aws:autoscaling:launchconfiguration

Vous pouvez utiliser les [options de configuration \(p. 658\)](#) dans l'espace de noms [aws:autoscaling:launchconfiguration \(p. 681\)](#) afin de configurer vos instances d'environnement, y compris les options supplémentaires qui ne sont pas disponibles dans la console.

L'exemple de [fichier de configuration \(p. 737\)](#) suivant utilise les options de configuration de base affichées dans cette rubrique. Par exemple, il utilise l'option DisableIMDSv1, abordée dans [IMDS \(p. 545\)](#). Il utilise également les options EC2KeyName et IamInstanceProfile abordées dans [Sécurité \(p. 626\)](#), et l'option BlockDeviceMappings, qui n'est pas disponible dans la console.

```
option_settings:
  aws:autoscaling:launchconfiguration:
    InstanceType: m1.small
    SecurityGroups: my-securitygroup
    MonitoringInterval: "1 minute"
    DisableIMDSv1: false
    EC2KeyName: my-keypair
    IamInstanceProfile: "aws-elasticbeanstalk-ec2-role"
    BlockDeviceMappings: "/dev/sdj=:100,/dev/sdh=snap-51eef269,/dev/sdb=ephemeral0"
```

Vous pouvez utiliser BlockDeviceMappings pour configurer des périphériques de stockage en mode bloc supplémentaires pour vos instances. Pour de plus amples informations, veuillez consulter [Exemple de mappage de périphérique de stockage en mode bloc](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

### Note

L'option InstanceType est obsolète. Elle est remplacée par la nouvelle option InstanceTypes plus puissante dans l'espace de noms [aws:ec2:instances \(p. 693\)](#). Vous pouvez utiliser cette nouvelle option pour spécifier une liste d'un ou plusieurs types d'instance pour votre environnement. La première valeur de cette liste est équivalente à la valeur de l'option InstanceType incluse dans l'espace de noms [aws:autoscaling:launchconfiguration](#) décrit ici. Nous vous recommandons de spécifier les types d'instance en utilisant la nouvelle option. Si elle est spécifiée, la nouvelle option prévaut sur la précédente. Pour de plus amples informations, veuillez consulter [the section called "Espace de noms aws:ec2:instances" \(p. 555\)](#).

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser

des fichiers de configuration pour configurer la même chose. Consultez [Valeurs recommandées \(p. 660\)](#) pour plus de détails.

## Configuration du service de métadonnées d'instance sur les instances de votre environnement

Les métadonnées d'instance sont des données relatives à une instance Amazon Elastic Compute Cloud (Amazon EC2) que les applications peuvent utiliser pour configurer ou gérer l'instance en cours d'exécution. Le service de métadonnées d'instance (IMDS) est un composant sur instance utilisé par le code sur l'instance pour accéder en toute sécurité aux métadonnées d'instance. Ce code peut être du code de plateforme Elastic Beanstalk sur vos instances d'environnement, le kit AWS SDK que votre application peut utiliser et même le code de votre application. Pour de plus amples informations, veuillez consulter [Métadonnées d'instance et données utilisateur](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Le code peut accéder aux métadonnées d'instance à partir d'une instance en cours d'exécution à l'aide de l'une des deux méthodes suivantes : Instance Metadata Service Version 1 (IMDSv1) ou Instance Metadata Service Version 2 (IMDSv2). IMDSv2 utilise des requêtes orientées session et atténue plusieurs types de vulnérabilités qui pourraient être utilisées pour essayer d'accéder à l'IMDS. Pour de plus amples informations sur ces deux méthodes, consultez [Configuration du service de métadonnées d'instance](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

### Sections

- [Prise en charge de la plateforme pour IMDS \(p. 545\)](#)
- [Choisir des méthodes IMDS \(p. 545\)](#)
- [Configuration d'IMDS à l'aide de la console Elastic Beanstalk \(p. 546\)](#)
- [Espace de noms aws:autoscaling:launchconfiguration \(p. 546\)](#)

## Prise en charge de la plateforme pour IMDS

Les anciennes versions de plateforme Elastic Beanstalk prennent en charge IMDSv1. Les nouvelles versions de la plateforme Elastic Beanstalk (toutes les versions de [plateforme Amazon Linux 2 \(p. 508\)](#)) prennent en charge à la fois IMDSv1 et IMDSv2. Vous pouvez configurer votre environnement pour prendre en charge les deux méthodes (par défaut) ou désactiver IMDSv1.

### Note

La désactivation d'IMDsv1 nécessite l'utilisation de modèles de lancement Amazon EC2. Lorsque vous configurez cette fonction au cours de la création ou des mises à jour de l'environnement, Elastic Beanstalk tente de configurer votre environnement de sorte qu'il utilise des modèles de lancement Amazon EC2 (si l'environnement ne les utilise pas déjà). Dans ce cas, si votre stratégie utilisateur ne dispose pas des autorisations nécessaires, la création ou les mises à jour de l'environnement peuvent échouer. Par conséquent, nous vous recommandons d'utiliser notre stratégie d'utilisateur gérée ou d'ajouter les autorisations requises à vos stratégies personnalisées. Pour de plus amples informations sur les autorisations requises, veuillez consulter [the section called "Création d'une stratégie utilisateur personnalisée" \(p. 947\)](#).

## Choisir des méthodes IMDS

Lorsque vous prenez une décision concernant les méthodes IMDS que votre environnement doit prendre en charge, tenez compte des cas d'utilisation suivants :

- AWS SDK – Si votre application utilise un kit AWS SDK, assurez-vous d'utiliser la dernière version du SDK. Les kits AWS SDK font des appels IMDS, et les versions plus récentes du SDK utilisent IMDSv2

chaque fois que possible. Si vous désactivez IMDSv1 ou si votre application utilise une ancienne version du SDK, les appels IMDS peuvent échouer.

- Votre code d'application – Si votre application effectue des appels IMDS, envisagez d'utiliser le kit AWS SDK pour effectuer les appels au lieu de procéder à des requêtes HTTP directes. De cette façon, vous n'avez pas besoin de modifier le code pour basculer entre les méthodes IMDS. Le kit AWS SDK utilise IMDSv2 chaque fois que possible.
- Code de plateforme Elastic Beanstalk – Notre code effectue des appels IMDS via le kit SDK AWS, et utilise donc IMDSv2 sur toutes les versions de plateforme qui le prennent en charge. Si votre code utilise un kit AWS SDK à jour et effectue tous les appels IMDS via le SDK, vous pouvez désactiver IMDSv1 en toute sécurité.

## Configuration d'IMDS à l'aide de la console Elastic Beanstalk

Vous pouvez modifier la configuration d'instance Amazon EC2 de votre environnement Elastic Beanstalk dans la console Elastic Beanstalk.

Pour configurer IMDS sur vos instances Amazon EC2 dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).



5. Définissez Désactiver IMDsv1 pour appliquer IMDsv2. DÉCOchez Désactiver IMDsv1 pour activer IMDsv1 et IMDsv2.
6. Choisissez Apply.

## Espace de noms aws:autoscaling:launchconfiguration

Vous pouvez utiliser une [option de configuration \(p. 658\)](#) dans l'espace de noms `aws:autoscaling:launchconfiguration` ([p. 681](#)) pour configurer IMDS sur les instances de votre environnement.

L'exemple [de fichier de configuration \(p. 737\)](#) suivant désactive IMDSv1 à l'aide de l'option `DisableIMDSv1`.

```
option_settings:  
  aws:autoscaling:launchconfiguration:  
    DisableIMDSv1: true
```

## Groupe Auto Scaling pour votre environnement Elastic Beanstalk

Votre environnement AWS Elastic Beanstalk inclut un groupe Auto Scaling qui gère les [instances Amazon EC2 \(p. 538\)](#) de votre environnement. Dans un environnement à instance unique, le groupe Auto Scaling s'assure qu'il y a toujours une instance en cours d'exécution. Dans un environnement à charge équilibrée, vous configurez le groupe avec une gamme d'instances à exécuter et l'Auto Scaling ajoute ou supprime les instances en fonction de la charge requise par vos besoins.

Le groupe Auto Scaling applique également la configuration de lancement pour les instances de votre environnement. Vous pouvez [modifier la configuration de lancement \(p. 538\)](#) pour modifier le type d'instance, la paire de clés, le stockage Amazon Elastic Block Store (Amazon EBS) et autres paramètres qui ne peuvent être configurés que lorsque vous lancez une instance.

Le groupe Auto Scaling utilise deux alarmes Amazon CloudWatch pour déclencher des opérations de mise à l'échelle. Les déclencheurs par défaut évoluent quand le trafic réseau sortant moyen de chaque instance est supérieur à 6 Mio ou inférieur à 2 Mio sur une période de cinq minutes. Pour utiliser Auto Scaling de façon efficace, [configurez des déclencheurs \(p. 555\)](#) adaptés à votre application, au type d'instance et aux exigences du service. Vous pouvez mettre à l'échelle en fonction de plusieurs statistiques, y compris la latence, les E/S disque, l'utilisation de l'UC et le nombre de demandes.

Pour optimiser l'utilisation des instances Amazon EC2 de votre environnement à travers les périodes prévisibles de trafic de pointe, [configurez votre groupe Auto Scaling pour modifier son nombre d'instances sur un calendrier \(p. 558\)](#). Vous pouvez planifier les modifications apportées à la configuration de votre groupe qui se produisent de manière quotidienne ou hebdomadaire, ou planifier des modifications exceptionnelles pour préparer les événements marketing qui généreront un trafic important sur votre site.

En option, Elastic Beanstalk peut combiner des instances à la demande et des instances [Spot \(p. 548\)](#) pour votre environnement. Vous pouvez configurer Amazon EC2 Auto Scaling pour surveiller et réagir automatiquement aux modifications qui affectent la disponibilité de vos instances Spot en activant [Rééquilibrage de capacité](#).

Auto Scaling surveille l'intégrité de chaque instance Amazon EC2 qu'il lance. Si une instance est résiliée de façon inattendue, Auto Scaling détecte cette résiliation et lance une instance de remplacement. Pour configurer le groupe afin d'utiliser le mécanisme de vérification de l'état de l'équilibrage de charge, veuillez consulter [Paramètre de vérification de l'état Auto Scaling \(p. 561\)](#).

Vous pouvez configurer Auto Scaling pour votre environnement à l'aide de la [console Elastic Beanstalk \(p. 550\)](#), de l'[interface de ligne de commande EB \(p. 554\)](#), ou des [options de configuration \(p. 555\)](#).

### Rubriques

- [Prise en charge d'une instance Spot \(p. 548\)](#)
- [Configuration du groupe Auto Scaling à l'aide de la console Elastic Beanstalk \(p. 550\)](#)
- [Configuration du groupe Auto Scaling à l'aide de l'interface de ligne de commande EB \(p. 554\)](#)
- [Options de configuration \(p. 555\)](#)
- [Déclencheurs de mise à l'échelle automatique \(p. 555\)](#)

- [Actions planifiées Auto Scaling \(p. 558\)](#)
- [Paramètre de vérification de l'état Auto Scaling \(p. 561\)](#)

## Prise en charge d'une instance Spot

Pour tirer parti des [instances Spot](#) Amazon EC2, vous pouvez activer une option Spot pour votre environnement. Le groupe Auto Scaling de votre environnement combine ensuite des options d'achat Amazon EC2 et gère un mélange d'instances à la demande et d'instances Spot.

Cette rubrique décrit les méthodes suivantes pour activer les demandes d'instance Spot pour votre environnement :

- La console Elastic Beanstalk — Pour plus d'informations, consultez [Composition de la flotte dans the section called “Configuration du groupe Auto Scaling à l'aide de la console Elastic Beanstalk” \(p. 550\)](#).
- L'interface de ligne de commande EB – Pour plus d'informations, consultez [the section called “Configuration du groupe Auto Scaling à l'aide de l'interface de ligne de commande EB” \(p. 554\)](#).
- L'option de configuration de l'espace de noms `aws:ec2:instances` – Pour plus d'informations, consultez [the section called “Options de configuration” \(p. 555\)](#).

### Important

La demande d'instances Spot peut varier considérablement d'un instant à l'autre, de même que la disponibilité d'instances Spot peut aussi considérablement fluctuer selon le nombre d'instances Amazon EC2 inutilisées disponibles. Par ailleurs, votre instance Spot est susceptible d'être interrompue.

Pour réduire l'impact de ces interruptions sur votre application, vous pouvez activer l'option Rééquilibrage de capacité offerte avec Amazon EC2 Auto Scaling. Lorsque cette fonction est activée, EC2 tente automatiquement de remplacer les instances Spot d'un groupe Auto Scaling avant qu'elles ne soient interrompues. Pour activer cette fonction, [configurez le groupe Auto Scaling \(p. 550\)](#) dans la console Elastic Beanstalk. Vous pouvez également définir la valeur de l'[option de configuration \(p. 555\)](#) `EnableCapacityRebalancing` Elastic Beanstalk sur `true` dans l'espace de noms `aws:autoscaling:asg` (p. 680).

Pour plus d'informations, consultez [Rééquilibrage de capacité](#) dans le Guide de l'utilisateur Amazon EC2 Auto Scaling et [Interruptions d'instance Spot](#) dans le Guide de l'utilisateur Amazon EC2 pour instances Linux.

Elastic Beanstalk propose plusieurs options de configuration pour la prise en charge de la fonction Spot. Ces options sont abordées dans les sections suivantes qui expliquent la configuration de votre groupe Auto Scaling.

Deux de ces options, dans l'espace de noms [aws:ec2:instances \(p. 693\)](#), méritent une attention toute particulière :

- `SpotFleetOnDemandBase`
- `SpotFleetOnDemandAboveBasePercentage`

Ces deux options sont en corrélation avec l'option `MinSize` de l'espace de nom [aws:autoscaling:asg???](#) (p. 680) :

- Seule `MinSize` détermine la capacité initiale de votre environnement, c'est-à-dire le nombre d'instances que vous souhaitez exécuter au minimum.
- `SpotFleetOnDemandBase` n'affecte pas la capacité initiale. Si Spot est activé, cette option ne détermine que le nombre d'instances à la demande allouées avant la prise en compte des instances Spot.

- Imaginez quand `SpotFleetOnDemandBase` est inférieur à `MinSize`. Vous aurez toujours exactement `MinSize` instances comme capacité initiale. Au moins `SpotFleetOnDemandBase` d'entre elles doivent être des instances à la demande.
- Imaginez que `SpotFleetOnDemandBase` est supérieur à `MinSize`. À mesure de la mise à l'échelle de votre environnement, vous êtes assuré d'obtenir au minimum une quantité d'instances supplémentaires égale à la différence entre les deux valeurs. Autrement dit, vous êtes assuré d'obtenir au moins  $(\text{SpotFleetOnDemandBase} - \text{MinSize})$  instances supplémentaires qui sont à la demande avant de satisfaire à l'exigence `SpotFleetOnDemandBase`.

Dans les environnements de production, les instances Spot sont particulièrement utiles dans le cadre d'un environnement scalable et à charge équilibrée. Nous ne recommandons pas d'utiliser Spot dans un environnement à instance unique. Si les instances Spot ne sont pas disponibles, vous risquez de perdre toute la capacité (une instance unique) de votre environnement. Vous pouvez toujours souhaiter utiliser une instance Spot dans un environnement à instance unique pour le développement ou le test. Dans ce cas, assurez-vous de définir `SpotFleetOnDemandBase` et `SpotFleetOnDemandAboveBasePercentage` à zéro. Tout autre paramétrage donne lieu à une instance à la demande.

#### Notes

- Certains comptes AWS plus anciens peuvent fournir à Elastic Beanstalk des types d'instance par défaut qui ne prennent pas en charge les instances Spot (par exemple, t1.micro). Si vous activez les demandes d'instances Spot et que l'erreur `None of the instance types you specified supports Spot (Aucun des types d'instance spécifiés ne prend en charge les instances Spot)` s'affiche, assurez-vous de configurer des types d'instance qui prennent en charge les instances Spot. Pour choisir des types d'instance Spot, utilisez [Spot Instance Advisor](#).
- L'activation des requêtes d'instance Spot nécessite l'utilisation de modèles de lancement Amazon EC2. Lorsque vous configurez cette fonction au cours de la création ou des mises à jour de l'environnement, Elastic Beanstalk tente de configurer votre environnement de sorte qu'il utilise des modèles de lancement Amazon EC2 (si l'environnement ne les utilise pas déjà). Dans ce cas, si votre stratégie utilisateur ne dispose pas des autorisations nécessaires, la création ou les mises à jour de l'environnement peuvent échouer. Par conséquent, nous vous recommandons d'utiliser notre stratégie d'utilisateur gérée ou d'ajouter les autorisations requises à vos stratégies personnalisées. Pour de plus amples informations sur les autorisations requises, veuillez consulter [the section called "Création d'une stratégie utilisateur personnalisée" \(p. 947\)](#).

Les exemples suivants illustrent différents scénarios de définition des différentes options de mise à l'échelle. Tous les exemples supposent qu'un environnement à charge équilibrée avec les demandes d'instances Spot activées a été défini.

#### Example 1 : Capacité initiale avec instances à la demande et instances Spot

##### Paramètres d'option

Option	NAMESPACE	Valeur
<code>MinSize</code>	<code>aws:autoscaling:asg</code>	10
<code>MaxSize</code>	<code>aws:autoscaling:asg</code>	24
<code>SpotFleetOnDemandBase</code>	<code>aws:ec2:instances</code>	4
<code>SpotFleetOnDemandAboveBasePercentage</code>	<code>aws:ec2:instances</code>	50

Dans cet exemple, l'environnement commence avec dix instances, dont sept instances à la demande (quatre de base et 50 % des six autres au-delà) et trois instances Spot. L'environnement peut accepter

jusqu'à 24 instances. Au fur et à mesure du dimensionnement, la proportion d'instances à la demande de la partie du parc au-delà des quatre instances à la demande de base est maintenue à 50 %, jusqu'à un maximum de 24 instances au total, dont 14 sont des instances à la demande (quatre de base et 50 % des 20 autres au-delà de la base) et 10 sont des instances Spot.

Example 2 : Capacité initiale avec uniquement des instances à la demande

Paramètres d'option

Option	Namespace	Valeur
MinSize	aws:autoscaling:asg	4
MaxSize	aws:autoscaling:asg	24
SpotFleetOnDemandBase	aws:ec2:instances	4
SpotFleetOnDemandAboveBasePercentage	aws:ec2:instances	50

Dans cet exemple, l'environnement commence avec quatre instances à la demande. L'environnement peut accepter jusqu'à 24 instances. Au fur et à mesure du dimensionnement, la proportion d'instances à la demande de la partie du parc au-delà des quatre instances à la demande de base est maintenue à 50 %, jusqu'à un maximum de 24 instances au total, dont 14 sont des instances à la demande (quatre de base et 50 % des 20 autres au-delà de la base) et 10 sont des instances Spot.

Example 3 : Instances à la demande de base supplémentaires au-delà de la capacité initiale

Paramètres d'option

Option	Namespace	Valeur
MinSize	aws:autoscaling:asg	3
MaxSize	aws:autoscaling:asg	24
SpotFleetOnDemandBase	aws:ec2:instances	4
SpotFleetOnDemandAboveBasePercentage	aws:ec2:instances	50

Dans cet exemple, l'environnement commence avec trois instances à la demande. L'environnement peut accepter jusqu'à 24 instances. La première instance supplémentaire au-delà des trois instances initiales est une instance à la demande, pour compléter les quatre instances à la demande de base. Au fur et à mesure du dimensionnement, la proportion d'instances à la demande de la partie du parc au-delà des quatre instances à la demande de base est maintenue à 50 %, jusqu'à un maximum de 24 instances au total, dont 14 sont des instances à la demande (quatre de base et 50 % des 20 autres au-delà de la base) et 10 sont des instances Spot.

## Configuration du groupe Auto Scaling à l'aide de la console Elastic Beanstalk

Vous pouvez configurer le fonctionnement d'Auto Scaling en modifiant Capacity (Capacité) sur la page Configuration de l'environnement dans la [console Elastic Beanstalk \(p. 429\)](#).

Pour configurer le groupe Auto Scaling dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Auto Scaling group (Groupe Auto Scaling), configurez les paramètres suivants.
  - Environment type (Type d'environnement) – Sélectionnez Load balanced (Charge équilibrée).
  - Min instances (Nombre minimal d'instances) – Nombre minimal d'instances EC2 que le groupe doit contenir à tout moment. Le groupe démarre avec le nombre minimal et ajoute les instances quand la condition de déclenchement d'une augmentation est remplie.
  - Max instances (Nombre maximal d'instances) – Nombre maximal d'instances EC2 que le groupe doit contenir à tout moment.

Note

Si vous utilisez des mises à jour propagées, assurez-vous que le nombre maximal d'instances est supérieur au paramètre [Instances minimums en service \(p. 490\)](#) pour les mises à jour propagées.

- Composition de la flotte — La valeur par défaut est Instances à la demande. Pour activer les demandes d'instance Spot, sélectionnez Combined purchase options and instance types (Options d'achat et types d'instance combinés).

Les options suivantes sont activées si vous choisissez d'activer les demandes d'instance Spot :

- Maximum Spot price (Prix Spot maximal) – Pour des recommandations sur les options tarifaires maximales pour les instances Spot, consultez [Historique de tarification d'instances Spot](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
- On-Demand base (À la demande de base) – Nombre minimal d'instances à la demande que votre groupe Auto Scaling alloue avant de considérer les instances Spot à mesure que votre environnement grandit.
- On-Demand above base (À la demande au-dessus de la base) – Pourcentage d'instances à la demande dans le cadre de la capacité supplémentaire que votre groupe Auto Scaling alloue au-delà des instances à la demande de base.

Note

Les options On-Demand base (À la demande de base) et On-Demand above base (À la demande au-dessus de la base) sont en corrélation avec les options Min et Max Instances répertoriées plus tôt. Pour de plus amples informations sur ces options, consultez [the section called “Prise en charge d'une instance Spot” \(p. 548\)](#).

- Enable Capacity Rebalancing (Activer le rééquilibrage de charge) — Cette option n'est pertinente que lorsque votre groupe Auto Scaling comprend au moins une instance Spot. Lorsque cette fonction est activée, EC2 tente automatiquement de remplacer les instances Spot du groupe Auto Scaling avant qu'elles ne soient interrompues, réduisant ainsi les interruptions d'instances Spot pour vos applications. Pour plus d'informations, consultez [Rééquilibrage de la capacité](#) dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.
- Instance type (Type d'instance) – Type d'instance Amazon EC2 lancée pour exécuter votre application. Pour plus d'informations, consultez [the section called “Type d'instance” \(p. 543\)](#).
- AMI ID (ID d'AMI) – Image machine utilisée par Elastic Beanstalk pour lancer des instances Amazon EC2 dans votre environnement. Pour plus d'informations, consultez [the section called “ID d'AMI” \(p. 544\)](#).

- Availability Zones (Zones de disponibilité) – Choisissez le nombre de zones de disponibilité entre lesquelles répartir les instances de votre environnement. Par défaut, le groupe Auto Scaling lance les instances de manière uniforme sur l'ensemble des zones utilisables. Pour concentrer vos instances en moins de zones, choisissez le nombre de zones à utiliser. Pour les environnements de production, utilisez au moins deux zones afin de vous assurer que votre application est disponible en cas de panne de l'une des zones de disponibilité.
- Placement (facultatif) – Choisissez les zones de disponibilité à utiliser. Utilisez ce paramètre si vos instances ont besoin de se connecter aux ressources de zones spécifiques ou si vous avez acheté [des instances réservées](#), qui sont spécifiques à une zone. Si vous définissez la stabilisation sur « 720 », les zones de disponibilité personnalisées sur « us-west-2a,us-west-2b » et la taille maximale sur « 4 », définissez le nombre de zones, vous devez choisir au moins autant de zones personnalisées.

Si vous lancez votre environnement dans un VPC personnalisé, vous ne pouvez pas configurer cette option. Dans un VPC personnalisé, vous choisissez les zones de disponibilité des sous-réseaux que vous affectez à votre environnement.

- Scaling cooldown (Stabilisation de la mise à l'échelle) – Délai d'attente (en secondes) avant que les instances se lancent ou se terminent après la mise à l'échelle, avant de continuer à analyser les déclencheurs. Pour de plus amples informations, veuillez consulter [Stabilisation de la mise à l'échelle](#).

## Modify capacity

Configure the compute capacity of your environment and Auto Scaling settings to optimize the number of instances used.

### Auto Scaling Group

#### Environment type

Load balanced

#### Instances

Min

1

Max

4

#### Fleet composition

Choose a mix of On-Demand and Spot Instances with multiple instance types. Spot Instances are automatically launched at the lowest available price. [Learn more](#)

- On-Demand instances
- Combine purchase options and instances

#### Maximum spot price

The maximum price per instance-hour, in USD, that you're willing to pay for a Spot Instance. Setting a custom price limits your chances to fulfill your target capacity using

- Default - the On-Demand price for each instance type (recommended)
- Set your maximum price

#### On-Demand base

The minimum number of On-Demand Instances that your Auto Scaling group provisions before considering Spot Instances as your environment scales out.

0

#### On-Demand above base

The percentage of On-Demand Instances as part of any additional capacity that your Auto Scaling group provisions beyond the On-Demand base instances.

70

%

#### Enable Capacity Rebalancing

Specifies whether to enable the Capacity Rebalancing feature for Spot Instances in your Auto Scaling Group. This option is only relevant when EnableSpot is true in the `aws elasticbeanstalk create-environment` command.

- Enabled

#### Instance types

Add acceptable instance types for your fleet. Change their order to set the launch priority of On-Demand Instances. This order doesn't affect Spot Instances. We recommend using a mix of instance types.

-- Choose Instance Types --

t2.micro (1vCPUs, 1GiB) X

t2.small (1vCPUs, 2GiB) X

#### AMI ID

ami-9999999999999999

#### Availability Zones

Number of Availability Zones (AZs) to use.

Any

#### Placement

Specify Availability Zones (AZs) to use.

-- Choose Availability Zones (AZs) --

553

#### Scaling cooldown

360

seconds

6. Choisissez Apply.

## Configuration du groupe Auto Scaling à l'aide de l'interface de ligne de commande EB

Lorsque vous créez un environnement à l'aide de la commande [eb create \(p. 1078\)](#), vous pouvez spécifier quelques options liées au groupe Auto Scaling de votre environnement. Voici quelques-unes des options qui vous aident à contrôler la capacité de votre environnement.

**--single**

Crée l'environnement avec une instance Amazon EC2 et sans équilibrer de charge. Si vous n'utilisez pas cette option, un équilibrer de charge est ajouté à l'environnement qui a été créé.

**--enable-spot**

Active les demandes d'instances Spot pour votre environnement.

Les options suivantes de la commande [eb create \(p. 1078\)](#) ne peuvent être utilisées qu'avec **--enable-spot**.

**--instance-types**

Liste les types d'instances Amazon EC2 que vous souhaitez que votre environnement utilise.

**--spot-max-price**

Prix maximum par heure d'unité, en dollars américains, que vous êtes prêt à payer pour une instance Spot. Pour des recommandations sur les options tarifaires maximales pour les instances Spot, consultez [Historique de tarification d'instances Spot](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

**--on-demand-base-capacity**

Nombre minimal d'instances à la demande que votre groupe Auto Scaling alloue avant de considérer les instances Spot à mesure que votre environnement grandit.

**--on-demand-above-base-capacity**

Pourcentage d'instances à la demande dans le cadre de la capacité supplémentaire allouée par votre groupe Auto Scaling supérieur au nombre d'instances qui est spécifié par l'option **--on-demand-base-capacity**.

L'exemple suivant crée un environnement et configure le groupe Auto Scaling de manière à activer les demandes d'instances Spot pour le nouvel environnement. Pour cet exemple, trois types d'instances possibles peuvent être utilisés.

```
$ eb create --enable-spot --instance-types "t2.micro,t3.micro,t3.small"
```

### Important

Il existe une autre option nommée de la même manière qui s'appelle **--instance-type** (pas de « s »), que l'interface de ligne de commande (CLI) EB reconnaît uniquement lors du traitement des instances à la demande. N'utilisez pas **--instance-type** (pas de « s ») avec l'option **--enable-spot**. Si vous le faites, l'interface de ligne de commande (CLI) EB l'ignore. Utilisez plutôt **--instance-types** (avec « s ») avec l'option **--enable-spot**.

## Options de configuration

Elastic Beanstalk fournit des [options de configuration \(p. 658\)](#) pour les paramètres Auto Scaling dans deux espaces de noms : [aws:autoscaling:asg \(p. 680\)](#) et [aws:ec2:instances \(p. 693\)](#).

### Espace de noms aws:autoscaling:asg

L'espace de noms [aws:autoscaling:asg \(p. 680\)](#) fournit des options pour le dimensionnement et la disponibilité d'ensemble.

L'exemple de [fichier de configuration \(p. 737\)](#) suivant configure le groupe Auto Scaling de manière à ce qu'il utilise de deux à quatre instances, des zones de disponibilité spécifiques et un temps de stabilisation de 12 minutes (720 secondes). Rééquilibrage de capacité pour les instances Spot est activé. Cette dernière option n'entre en vigueur que si `EnableSpot` a la valeur `true` dans l'espace de noms [aws:ec2:instances \(p. 693\)](#), comme illustré dans l'exemple de fichier de configuration qui suit.

```
option_settings:  
  aws:autoscaling:asg:  
    Availability Zones: Any  
    Cooldown: '720'  
    Custom Availability Zones: 'us-west-2a,us-west-2b'  
    MaxSize: '4'  
    MinSize: '2'  
    EnableCapacityRebalancing: true
```

### Espace de noms aws:ec2:instances

L'espace de noms [aws:ec2:instances \(p. 693\)](#) fournit des options liées aux instances de votre environnement, y compris la gestion d'instances Spot. Il complète [aws:autoscaling:launchconfiguration \(p. 681\)](#) et [aws:autoscaling:asg \(p. 680\)](#).

Lorsque vous mettez à jour la configuration de votre environnement et supprimez un ou plusieurs types d'instance de l'option `InstanceTypes`, Elastic Beanstalk résilie toutes les instances Amazon EC2 exécutées sur les types d'instance supprimés. Le groupe Auto Scaling de votre environnement lance ensuite de nouvelles instances, si nécessaire pour compléter la capacité souhaitée, en utilisant les types d'instance spécifiés actuels.

L'exemple de [fichier de configuration \(p. 737\)](#) suivant configure le groupe Auto Scaling pour activer les demandes d'instances Spot pour votre environnement. Trois types d'instances possibles peuvent être utilisés. Au moins une instance à la demande est utilisée pour la capacité de base, et 33 % soutenus des instances à la demande sont utilisés pour toute capacité supplémentaire.

```
option_settings:  
  aws:ec2:instances:  
    EnableSpot: true  
    InstanceTypes: 't2.micro,t3.micro,t3.small'  
    SpotFleetOnDemandBase: '1'  
    SpotFleetOnDemandAboveBasePercentage: '33'
```

Pour choisir des types d'instance Spot, utilisez [Spot Instance Advisor](#).

## Déclencheurs de mise à l'échelle automatique

Le groupe Auto Scaling de votre environnement Elastic Beanstalk utilise deux alarmes Amazon CloudWatch pour déclencher des opérations de mise à l'échelle. Les déclencheurs par défaut évoluent

quand le trafic réseau sortant moyen de chaque instance est supérieur à 6 Mo ou inférieur à 2 Mo sur une période de cinq minutes. Pour utiliser Amazon EC2 Auto Scaling de façon efficace, configurez des déclencheurs adaptés à votre application, au type d'instance et aux exigences du service. Vous pouvez mettre à l'échelle en fonction de plusieurs statistiques, y compris la latence, les E/S disque, l'utilisation de l'UC et le nombre de demandes.

Pour de plus amples informations sur les métriques et alarmes CloudWatch, veuillez consulter [Amazon CloudWatch Concepts](#) dans le Guide de l'utilisateur d'Amazon CloudWatch.

## Configuration des déclencheurs Auto Scaling

Vous pouvez configurer les déclencheurs qui ajustent le nombre d'instances du groupe Auto Scaling de votre environnement dans la console Elastic Beanstalk.

Pour configurer des déclencheurs dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Déclencheurs de dimensionnement, configurez les paramètres suivants :
  - Métrique – Mesure utilisée pour votre déclencheur Auto Scaling.
  - Statistique – Calcul de statistiques que le déclencheur doit utiliser, comme Average.
  - Unité – Unité pour la métrique du déclencheur, comme Octets.
  - Période – Spécifie la fréquence à laquelle Amazon CloudWatch mesure les métriques pour votre déclencheur.
  - Breach duration (Durée de la faille) – Durée, en minutes, pendant laquelle une métrique peut se trouver en dehors des seuils supérieur et inférieur avant de déclencher une opération de mise à l'échelle.
  - Seuil supérieur – Si la métrique dépasse ce nombre pendant la durée de l'utilisation hors limites, une opération de mise à l'échelle est déclenchée.
  - Incrémentation d'augmentation – Nombre d'instances Amazon EC2 à ajouter, dans le cadre d'une activité de mise à l'échelle.
  - Seuil inférieur – Si la métrique tombe en-dessous de ce nombre pendant la durée de l'utilisation hors limites, une opération de mise à l'échelle est déclenchée.
  - Incrémentation de diminution – Le nombre d'instances Amazon EC2 à supprimer, dans le cadre d'une activité de mise à l'échelle.

### Scaling triggers

**Metric**  
Change the metric that is monitored to determine if the environment's capacity is too low or too high.

NetworkOut

**Statistic**  
Choose how the metric is interpreted.

Average

**Unit**

Bytes

**Period**  
The period between metric evaluations.

5 Min

**Breach duration**  
The amount of time a metric can exceed a threshold before triggering a scaling operation.

5 Min

**Upper threshold**

6000000 Bytes

**Scale up increment**

1 EC2 instances

**Lower threshold**

2000000 Bytes

**Scale down increment**

-1 EC2 instances

6. Choisissez Apply.

## Espace de noms aws:autoscaling:trigger

Elastic Beanstalk fournit des [options de configuration \(p. 658\)](#) pour les paramètres Auto Scaling dans l'espace de noms [aws:autoscaling:trigger \(p. 688\)](#). Les paramètres de cet espace de noms sont organisés par la ressource à laquelle ils s'appliquent.

```
option_settings:  
    AWSEBAutoScalingScaleDownPolicy.aws:autoscaling:trigger:  
        LowerBreachScaleIncrement: '-1'  
    AWSEBAutoScalingScaleUpPolicy.aws:autoscaling:trigger:  
        UpperBreachScaleIncrement: '1'  
    AWSEBCloudwatchAlarmHigh.aws:autoscaling:trigger:  
        UpperThreshold: '6000000'  
    AWSEBCloudwatchAlarmLow.aws:autoscaling:trigger:  
        BreachDuration: '5'  
        EvaluationPeriods: '1'  
        LowerThreshold: '2000000'  
        MeasureName: NetworkOut  
        Period: '5'  
        Statistic: Average  
        Unit: Bytes
```

## Actions planifiées Auto Scaling

Pour optimiser l'utilisation des instances Amazon EC2 de votre environnement à travers les périodes prévisibles de trafic de pointe, configurez votre groupe Amazon EC2 Auto Scaling pour modifier son nombre d'instances sur un calendrier. Vous pouvez configurer votre environnement avec une action récurrente pour augmenter la capacité le matin et la réduire le soir, lorsque le trafic est faible. Par exemple, si une raison telle qu'un événement marketing amène du trafic vers votre site pendant une période limitée, vous pouvez planifier un événement unique pour augmenter la capacité au début de cet événement, et un autre pour le réduire à la fin.

Vous pouvez définir jusqu'à 120 actions planifiées actives par environnement. Elastic Beanstalk conserve également jusqu'à 150 actions planifiées expirées, que vous pouvez réutiliser en mettant à jour leurs paramètres.

## Configuration d'actions planifiées

Vous pouvez créer des actions planifiées pour le groupe Auto Scaling de votre environnement dans la console Elastic Beanstalk.

Pour configurer des actions planifiées dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Dans la section Time-based scaling (Dimensionnement basé sur la durée), choisissez Add scheduled action (Ajouter une action planifiée).

The screenshot shows the 'Time-based Scaling' section of the AWS Elastic Beanstalk developer manual. It includes a 'Current status' summary, a 'Time zone' selection (UTC is chosen), and a table for managing scheduled actions. The table has columns for Name, Min, Max, Desired, and Next occurrence. A note at the bottom states 'No scheduled actions'.

6. Renseignez les paramètres des actions planifiées suivants :

- Nom – Indiquez un nom unique de 255 caractères alphanumériques au maximum et sans espace.
- Instances – Choisissez le nombre minimum et maximum d'instances à appliquer au groupe Auto Scaling.
- Capacité souhaitée (facultatif) – Définissez la capacité souhaitée initiale pour le groupe Auto Scaling. Lorsque l'action planifiée est appliquée, des déclencheurs ajustent la capacité souhaitée en fonction de leurs paramètres.
- Occurrence – Choisissez Récurrent pour répéter l'action de mise à l'échelle selon un calendrier spécifique.
- Heure de début – Pour des actions ponctuelles, choisissez la date et l'heure d'exécution de l'action.

Pour des actions récurrentes, l'heure de début est facultative. Spécifiez-le pour choisir le moment le plus proche où l'action est exécutée. Après cette période, l'action se répète selon l'expression de la Référence.

- Référence – Utilisez une expression [Cron](#) pour spécifier la fréquence à laquelle vous souhaitez que l'action planifiée se produise. Par exemple, 30 6 \* \* 2 exécute l'action tous les mardis à 6h30 UTC.
- Heure de fin (facultatif) – Facultatif pour les actions récurrentes. Si elle est spécifiée, l'action se répète selon l'expression de la Référence et n'est plus exécutée après cette période.

Lorsqu'une action planifiée se termine, Auto Scaling ne revient pas automatiquement à ses paramètres précédents. Configurez une seconde action planifiée pour rétablir les paramètres d'origine d'Auto Scaling selon vos besoins.

7. Choisissez Ajouter.
8. Choisissez Apply.

Note

Les actions planifiées ne sont pas enregistrées tant qu'elles ne sont pas appliquées.

## Espace de noms aws:autoscaling:scheduledaction

Si vous devez configurer un grand nombre d'actions planifiées, vous pouvez utiliser les [fichiers de configuration \(p. 737\)](#) ou [l'API Elastic Beanstalk \(p. 678\)](#) pour appliquer les modifications d'option de configuration à partir d'un fichier YAML ou JSON. Ces méthodes vous permettent également d'accéder à l'[option Suspend \(p. 687\)](#) pour désactiver temporairement une action planifiée récurrente.

### Note

Lorsque vous utilisez des options de configuration d'action planifiée en dehors de la console, utilisez le format d'heure ISO 8601 pour spécifier les heures de début et de fin en UTC. Par exemple, 2015-04-28T04:07:02Z. Pour plus d'informations sur le format horaire ISO 8601, accédez à [Date and Time Formats](#). Les dates doivent être uniques dans toutes les actions planifiées.

Elastic Beanstalk fournit des options de configuration pour les paramètres d'action planifiée dans l'espace de noms [aws:autoscaling:scheduledaction \(p. 687\)](#). Utilisez le champ `resource_name` pour spécifier le nom de l'action planifiée.

### Example Scheduled-scale-up-specific-time-long.config

Ce fichier de configuration indique à Elastic Beanstalk de monter en puissance de 5 à 10 instances à cette date : 2015-12-12T00:00:00Z.

```
option_settings:
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: MinSize
    value: '5'
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: MaxSize
    value: '10'
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: DesiredCapacity
    value: '5'
  - namespace: aws:autoscaling:scheduledaction
    resource_name: ScheduledScaleUpSpecificTime
    option_name: StartTime
    value: '2015-12-12T00:00:00Z'
```

### Example Scheduled-scale-up-specific-time.config

Pour utiliser la syntaxe abrégée avec la CLI EB ou les fichiers de configuration, ajoutez le nom de la ressource au début de l'espace de noms.

```
option_settings:
  ScheduledScaleUpSpecificTime.aws:autoscaling:scheduledaction:
    MinSize: '5'
    MaxSize: '10'
    DesiredCapacity: '5'
    StartTime: '2015-12-12T00:00:00Z'
```

### Example Scheduled-scale-down-specific-time.config

Ce fichier de configuration indique à Elastic Beanstalk de diminuer en puissance à 2015-12-12T 07:00:00 Z.

```
option_settings:  
  ScheduledScaleDownSpecificTime.aws:autoscaling:scheduledaction:  
    MinSize: '1'  
    MaxSize: '1'  
    DesiredCapacity: '1'  
    StartTime: '2015-12-12T07:00:00Z'
```

#### Example Scheduled-periodic-scale-up.config

Ce fichier de configuration indique à Elastic Beanstalk de monter en puissance tous les jours à 9h du matin. L'action doit commencer le 14 mai 2015 et se terminer le 12 janvier 2016.

```
option_settings:  
  ScheduledPeriodicScaleUp.aws:autoscaling:scheduledaction:  
    MinSize: '5'  
    MaxSize: '10'  
    DesiredCapacity: '5'  
    StartTime: '2015-05-14T07:00:00Z'  
    EndTime: '2016-01-12T07:00:00Z'  
    Recurrence: 0 9 * * *
```

#### Example Scheduled-periodic-scale-down.config

Ce fichier de configuration indique à Elastic Beanstalk de diminuer en puissance afin de pas exécuter d'instance chaque jour à 18 h. Si vous savez que votre application est surtout inactive en dehors des heures d'ouverture, vous pouvez créer une action planifiée similaire. Si votre application doit être arrêtée en dehors des heures d'ouverture, définissez MaxSize sur 0.

```
option_settings:  
  ScheduledPeriodicScaleDown.aws:autoscaling:scheduledaction:  
    MinSize: '0'  
    MaxSize: '1'  
    DesiredCapacity: '0'  
    StartTime: '2015-05-14T07:00:00Z'  
    EndTime: '2016-01-12T07:00:00Z'  
    Recurrence: 0 18 * * *
```

#### Example Scheduled-weekend-scale-down.config

Ce fichier de configuration indique à Elastic Beanstalk de mettre à l'échelle tous les vendredis à 18h. Si vous savez que votre application ne reçoit pas autant de trafic au cours du week-end, vous pouvez créer une action planifiée similaire.

```
option_settings:  
  ScheduledWeekendScaleDown.aws:autoscaling:scheduledaction:  
    MinSize: '1'  
    MaxSize: '4'  
    DesiredCapacity: '1'  
    StartTime: '2015-12-12T07:00:00Z'  
    EndTime: '2016-01-12T07:00:00Z'  
    Recurrence: 0 18 * * 5
```

## Paramètre de vérification de l'état Auto Scaling

Amazon EC2 Auto Scaling surveille l'état de chaque instance Amazon Elastic Compute Cloud (Amazon EC2) lancée. Si une instance est résiliée de façon inattendue, Auto Scaling détecte cette résiliation et lance

une instance de remplacement. Par défaut, le groupe Auto Scaling créé pour votre environnement utilise les [contrôles d'état Amazon EC2](#). Si une instance de votre environnement échoue lors de la vérification de l'état Amazon EC2, elle est désactivée et remplacée par Auto Scaling.

Les vérifications de l'état Amazon EC2 ne portent que sur l'état d'une instance, et non sur celui de votre application, de votre serveur ou des conteneurs Docker exécutés sur l'instance. Si votre application se bloque, mais que l'instance sur laquelle elle s'exécute reste saine, elle peut être exclue de l'équilibrer de charge, mais elle ne sera pas automatiquement remplacée par Auto Scaling. Le comportement par défaut convient pour la résolution des problèmes. Si Auto Scaling a remplacé l'instance dès le blocage de l'application, il est possible que vous ne vous rendiez pas compte du problème, même si le blocage s'est produit peu de temps après le démarrage.

Si vous souhaitez qu'Auto Scaling remplace les instances dont l'application a cessé de répondre, vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour configurer le groupe Auto Scaling afin qu'il utilise les vérifications de l'état d'Elastic Load Balancing. L'exemple suivant définit le groupe pour utiliser les vérifications de l'état de l'équilibrer de charge, en plus du contrôle de l'état Amazon EC2, pour déterminer l'état d'une instance.

Example .ebextensions/autoscaling.config

```
Resources:  
  AWSEBAutoScalingGroup:  
    Type: "AWS::AutoScaling::AutoScalingGroup"  
    Properties:  
      HealthCheckType: ELB  
      HealthCheckGracePeriod: 300
```

Pour de plus amples informations sur les propriétés `HealthCheckType` et `HealthCheckGracePeriod`, veuillez consulter [AWS::AutoScaling::AutoScalingGroup](#) dans le Guide de l'utilisateur AWS CloudFormation et [Vérifications de l'état pour les instances d'Auto Scaling](#) dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Par défaut, la vérification de l'état Elastic Load Balancing est configurée pour tenter une connexion TCP vers votre instance via le port 80. Cela confirme que le serveur web s'exécutant sur l'instance accepte les connexions. Cependant, il se peut que vous souhaitiez [personnaliser la vérification de l'état de l'équilibrer de charge \(p. 562\)](#) afin de vous assurer que votre application, et non pas seulement le serveur web, se trouve dans un état correct. Le paramètre de période de grâce définit le laps de temps (en secondes) pendant lequel une instance peut échouer aux vérifications de l'état sans être suspendue ou remplacée. Comme les instances peuvent être restaurées après avoir été exclues de l'équilibrer de charge, veuillez attribuer à votre instance un laps de temps adapté à votre application.

## Équilibrer de charge pour votre environnement Elastic Beanstalk

L'équilibrer de charges répartit le trafic entre les instances de votre environnement. Lorsque vous activez [l'équilibrage de charge \(p. 520\)](#), AWS Elastic Beanstalk crée un équilibrer de charge [Elastic Load Balancing](#) dédié à votre environnement. Elastic Beanstalk gère entièrement cet équilibrer de charge, en prenant soin des paramètres de sécurité et en veillant à résilier l'équilibrer de charge lorsque vous arrêtez votre environnement.

Vous pouvez également choisir de partager un équilibrer de charge entre plusieurs environnements Elastic Beanstalk. Avec un équilibrer de charge partagé, vous économisez sur les coûts opérationnels en évitant d'utiliser un équilibrer de charge dédié pour chaque environnement. Vous assumez également davantage la responsabilité de gestion de l'équilibrer de charge partagé que vos environnements utilisent.

Les types d'équilibrer de charge d'Elastic Load Balancing sont les suivants :

- [Classic Load Balancer](#) – Équilibrer de charge de la génération précédente. Achemine le trafic de demandes HTTP, HTTPS ou TCP vers différents ports sur des instances d'environnement.
- [Application Load Balancer](#) – Équilibrer de charge de couche d'application. Achemine le trafic de demandes HTTP ou HTTPS vers différents ports sur des instances d'environnement en fonction du chemin d'accès de la demande.
- [Network Load Balancer](#) – Équilibrer de charge de couche réseau. Achemine le trafic de demandes TCP vers différents ports sur des instances d'environnement. Prend en charge les vérifications de l'état actives et passives.

Elastic Beanstalk prend en charge les trois types d'équilibrer de charge. Le tableau suivant indique les types que vous pouvez utiliser avec les deux modèles d'utilisation :

Nouveau type d'équilibrer de charge	Dédié	Partagé
Équilibrer de charge classique	✓ Oui	X Non
Application Load Balancer	✓ Oui	✓ Oui
Équilibrer de charge du réseau	✓ Oui	X Non

Par défaut, Elastic Beanstalk crée un équilibrer de charge Application Load Balancer pour votre environnement lorsque vous activez l'équilibrage de charge avec la console Elastic Beanstalk ou l'interface de ligne de commande EB. Il configure l'équilibrer de charge pour écouter le trafic HTTP sur le port 80 et transmettre ce trafic aux instances sur le même port. Vous pouvez choisir le type d'équilibrer de charge que votre environnement utilise uniquement lors de la création de l'environnement. Ultérieurement, vous pourrez modifier les paramètres pour gérer le comportement de l'équilibrer de charge de votre environnement d'exécution, mais pas en changer le type.

#### Note

Votre environnement doit se trouver dans un VPC avec des sous-réseaux dans au moins deux zones de disponibilité pour créer un équilibrer de charge Application Load Balancer. Tous les nouveaux comptes AWS incluent les VPC par défaut qui répondent à cette exigence. Si votre environnement est dans un VPC avec des sous-réseaux dans une seule zone de disponibilité, la valeur par défaut correspond à un équilibrer de charge Classic Load Balancer. Si vous n'avez aucun sous-réseau, vous ne pouvez pas activer l'équilibrage de charge.

Vous pouvez créer et gérer des environnements avec tous les types d'équilibrer de charge à l'aide de la console Elastic Beanstalk, de la commande [eb create \(p. 1078\)](#) de l'interface de ligne de commande EB ou des API Elastic Beanstalk.

Consultez les rubriques suivantes pour en savoir plus sur chaque type d'équilibrer de charge pris en charge par Elastic Beanstalk, sur ses fonctionnalités et sur la façon de le configurer et de le gérer dans un environnement Elastic Beanstalk, ainsi que pour découvrir comment configurer un équilibrer de charge pour qu'il [charge les journaux d'accès \(p. 617\)](#) dans Amazon S3.

#### Rubriques

- [Configuration d'un équilibrer de charge classique \(p. 564\)](#)
- [Configuration d'un Application Load Balancer \(p. 574\)](#)
- [Configuration d'un équilibrer de charge Application Load Balancer partagé \(p. 591\)](#)
- [Configuration d'un équilibrer de charge réseau \(p. 607\)](#)

- Configuration des journaux d'accès (p. 617)

## Configuration d'un équilibrer de charge classique

Lorsque vous activez l'[équilibrage de charge \(p. 520\)](#), votre environnement AWS Elastic Beanstalk est équipé d'un équilibrer de charge Elastic Load Balancing qui permet de répartir le trafic entre les instances de votre environnement. Elastic Load Balancing prend en charge plusieurs types d'équilibrer de charge. Pour en savoir plus, consultez le [Guide de l'utilisateur Elastic Load Balancing](#). Elastic Beanstalk peut créer un équilibrer de charge pour vous, ou vous permettre de spécifier un équilibrer de charge partagé que vous avez créé.

Cette rubrique décrit la configuration d'un équilibrer de charge [Classic Load Balancer](#) créé par Elastic Beanstalk et dédié à votre environnement. Pour de plus amples informations sur la configuration de tous les types d'équilibrer de charge pris en charge par Elastic Beanstalk, veuillez consulter [Équilibrer de charge pour votre environnement Elastic Beanstalk \(p. 562\)](#).

### Note

Vous pouvez choisir le type d'équilibrer de charge que votre environnement utilise uniquement lors de la création de l'environnement. Ultérieurement, vous pourrez modifier les paramètres pour gérer le comportement de l'équilibrer de charge de votre environnement d'exécution, mais pas en changer le type.

## Introduction

Un équilibrer de charge [Classic Load Balancer](#) est l'équilibrer de charge Elastic Load Balancing de génération précédente. Il prend en charge l'acheminement du trafic des demandes HTTP, HTTPS ou TCP vers différents ports sur des instances d'environnement.

Lorsque votre environnement utilise un équilibrer de charge Classic Load Balancer, Elastic Beanstalk le configure par défaut pour [écouter](#) le trafic HTTP sur le port 80 et le transférer aux instances du même port. Pour prendre en charge les connexions sécurisées, vous pouvez configurer votre équilibrer de charge avec un écouteur sur le port 443 et un certificat TLS.

L'équilibrer de charge utilise une [vérification de l'état](#) pour déterminer si les instances Amazon EC2 exécutant votre application sont saines. La vérification de l'état envoie une demande à une URL spécifiée selon un intervalle défini. Si l'URL renvoie un message d'erreur ou ne répond pas dans un délai spécifique, la vérification de l'état échoue.

Si votre application enregistre de meilleures performances en traitant plusieurs demandes à partir du même client sur un seul serveur, vous pouvez configurer votre équilibrer de charge pour utiliser des [sessions permanentes](#). Avec les sessions permanentes, l'équilibrer de charge ajoute un cookie aux réponses HTTP, qui identifie l'instance Amazon EC2 ayant traité la demande. Lorsqu'une demande ultérieure est reçue du même client, l'équilibrer de charge utilise le cookie pour envoyer la demande à la même instance.

Avec l'[équilibrage de charge entre zones](#), chaque nœud de l'équilibrer de charge pour votre équilibrer de charge Classic Load Balancer répartit les demandes uniformément entre les instances enregistrées dans toutes les zones de disponibilité activées. Si l'équilibrage de charge entre zones est désactivé, chaque nœud de l'équilibrer de charge répartit les demandes uniformément entre les instances enregistrées dans sa zone de disponibilité uniquement.

Lorsqu'une instance est supprimée de l'équilibrer de charge parce qu'elle n'est plus saine ou parce que la capacité de l'environnement est diminuée, [Connection Draining](#) laisse à l'instance le temps de terminer les demandes avant de fermer la connexion entre l'instance et l'équilibrer de charge. Vous pouvez modifier le laps de temps accordé aux instances pour l'envoi d'une réponse ou désactiver totalement le drainage de la connexion.

#### Note

Connection Draining est activé par défaut lorsque vous créez un environnement via la console Elastic Beanstalk ou l'interface de ligne de commande EB. Pour les autres clients, vous pouvez l'activer via les [options de configuration \(p. 573\)](#).

Vous pouvez utiliser les paramètres avancés de l'équilibrer de charge pour configurer des écouteurs sur des ports arbitraires, modifier des paramètres de session permanente supplémentaires et configurer l'équilibrer de charge pour vous connecter en toute sécurité aux instances EC2. Ces paramètres sont disponibles via les [options de configuration \(p. 573\)](#) que vous pouvez définir avec des fichiers de configuration dans votre code source, ou directement sur un environnement via l'API Elastic Beanstalk. Un grand nombre de ces paramètres sont également disponibles dans la console Elastic Beanstalk. Vous pouvez également configurer un équilibrer de charge pour [charger les journaux d'accès \(p. 617\)](#) dans Amazon S3.

## Configuration d'un équilibrer de charge Classic Load Balancer à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer le port, le certificat HTTPS et d'autres paramètres d'un équilibrer Classic Load Balancer lors de la création de l'environnement ou plus tard, lorsque votre environnement est en cours d'exécution.

Pour configurer un équilibrer de charge Classic Load Balancer dans la console Elastic Beanstalk lors de la création de l'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements).
3. Choisissez [Create a new environment \(Créer un nouvel environnement\) \(p. 442\)](#) pour commencer à créer votre environnement.
4. Sur la page principale de l'assistant, avant de choisir Créer un environnement, choisissez Configurer plus d'options.
5. Choisissez le préréglage de configuration Haute disponibilité.

Sinon, dans la catégorie de configuration Capacité, configurez un type d'environnement avec un Équilibrage de charge. Pour plus d'informations, consultez [Capacity \(p. 450\)](#).

6. Dans la catégorie de configuration Load balancer (Équilibrer de charge), choisissez Edit (Modifier).
7. Sélectionnez l'option Classic Load Balancer si elle n'est pas déjà sélectionnée.

The screenshot shows the 'Modify load balancer' configuration screen. At the top, there is a breadcrumb trail: 'Elastic Beanstalk > Applications > getting-started-app'. Below the title 'Modify load balancer', there is a list of three options:

- Application Load Balancer  
Application layer load balancer—routing HTTP and HTTPS traffic based on protocol, port, and route to environment processes.
- Classic Load Balancer  
Previous generation — HTTP, HTTPS, and TCP
- Network Load Balancer  
Ultra-high performance and static IP addresses for your application.

8. Effectuez toutes les modifications de configuration de l'équilibrer de charge Classic Load Balancer exigées par votre environnement.
9. Choisissez Enregistrer, puis effectuez toutes les autres modifications de configuration exigées par votre environnement.
10. Choisissez Create environment.

Pour configurer un équilibrer de charge Classic Load Balancer d'un environnement en cours d'exécution dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Load balancer (Équilibrer de charge), choisissez Edit (Modifier).

#### Note

Si la catégorie de configuration Load balancer (Équilibrer de charge) ne dispose pas du bouton Edit (Modifier), cela signifie que votre environnement ne dispose pas d'un équilibrer de charge. Pour apprendre à en configurer un, consultez [Changement de type d'environnement \(p. 520\)](#).

5. Effectuez les modifications de configuration de l'équilibrer de charge Classic Load Balancer exigées par votre environnement.
6. Choisissez Apply.

Paramètres de l'équilibrer de charge Classic Load Balancer

- [Écouteurs \(p. 566\)](#)
- [Sessions \(p. 570\)](#)
- [Equilibrage de charge entre zones \(p. 570\)](#)
- [Drainage de la connexion \(p. 570\)](#)
- [Vérification de l'état \(p. 571\)](#)

## Écouteurs

Utilisez cette liste pour spécifier plusieurs écouteurs pour votre équilibrer de charge. Chaque écouteur achemine le trafic client entrant sur un port spécifié à l'aide d'un protocole spécifié vers vos instances. Au départ, la liste affiche l'écouteur par défaut, lequel achemine le trafic HTTP entrant sur le port 80 vers les serveurs d'instance de votre environnement qui écoutent le trafic HTTP sur le port 80.

## Classic Load Balancer

You can specify listeners for your load balancer. Each listener routes incoming client traffic on a specified port using a specific protocol to one or more instances. By default, we've configured your load balancer with a standard web server on port 80.

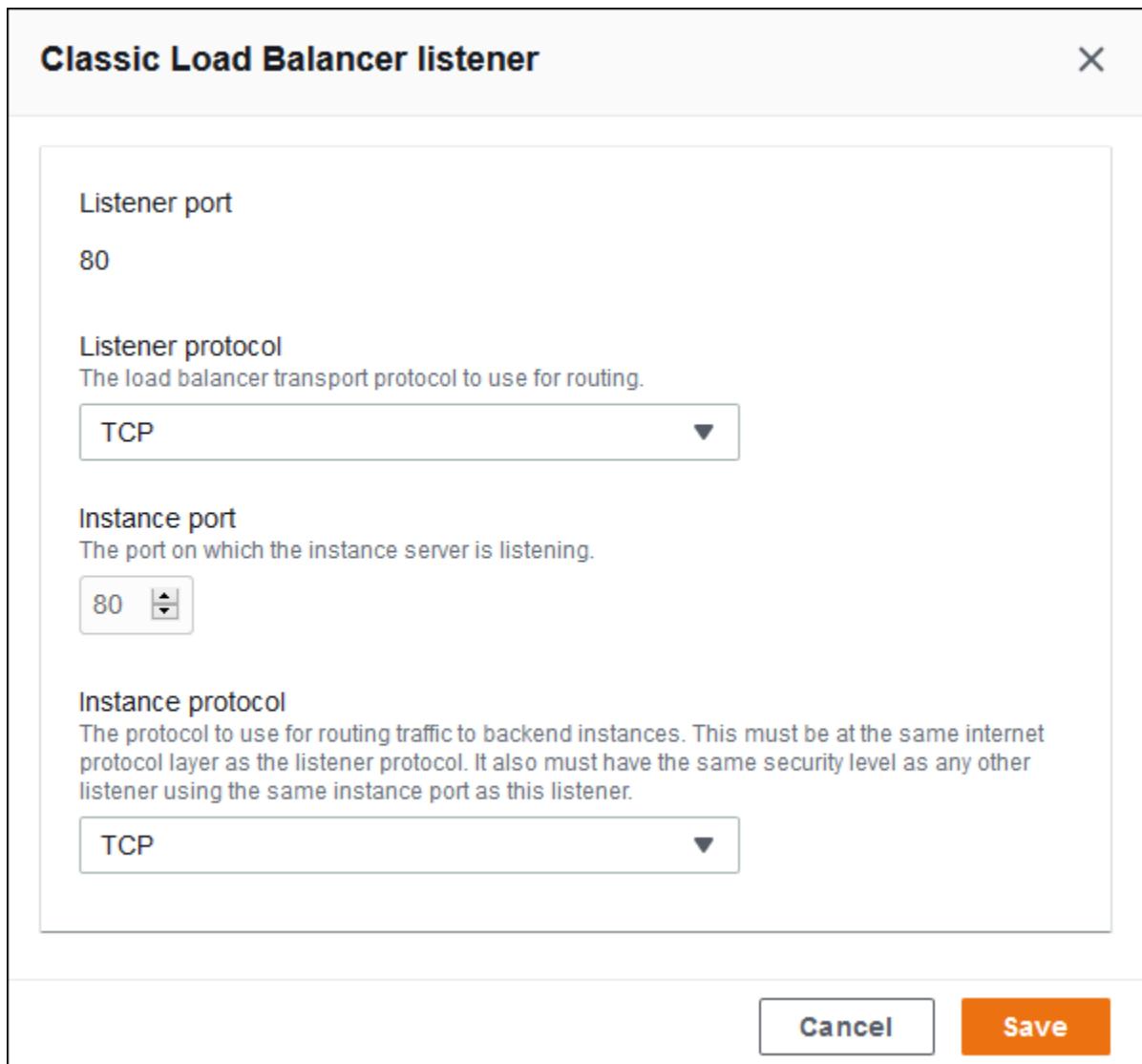
Action

<input type="checkbox"/>	Port	Protocol	Instance port	Instance protocol	SSL cert
<input type="checkbox"/>	80	HTTP	80	HTTP	--

Pour configurer un écouteur existant

1. Cochez la case en regard de son entrée de table, puis choisissez Actions et sélectionnez l'action de votre choix.
2. Si vous choisissez Modifier, utilisez la boîte de dialogue Écouteur Classic Load Balancer pour modifier les paramètres, puis choisissez Enregistrer.

Par exemple, vous pouvez modifier l'écouteur par défaut et remplacer la valeur du paramètre Protocole, HTTP, par TCP si vous souhaitez que l'équilibrer de charge transfère la demande en l'état. Ce paramètre évite que l'équilibrer de charge réécrire les en-têtes (y compris X-Forwarded-For). La technique ne fonctionne pas avec les sessions permanentes.



#### Pour ajouter un écouteur

1. Choisissez Ajouter un écouteur.
2. Dans la boîte de dialogue Écouteur Classic Load Balancer, configurez les paramètres de votre choix, puis sélectionnez Ajouter.

L'ajout d'un écouteur sécurisé est une cas d'utilisation courant. L'exemple présenté dans l'image qui suit illustre l'ajout d'un écouteur pour le trafic HTTPS sur le port 443. Cet écouteur achemine le trafic entrant vers les serveurs d'instances de l'environnement qui écoutent le trafic HTTPS sur le port 443.

Pour pouvoir configurer un écouteur HTTPS, assurez-vous que vous disposez d'un certificat SSL valide. Effectuez l'une des actions suivantes :

- Si AWS Certificate Manager (ACM) est [disponible dans votre région AWS](#), créez ou importez un certificat avec ACM. Pour plus d'informations sur une demande de certificat ACM, consultez [Demande de certificat](#) dans le Guide de l'utilisateur AWS Certificate Manager. Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez [Importation de certificats](#) dans le Guide de l'utilisateur AWS Certificate Manager.

- Si ACM n'est pas [disponible dans votre région AWS](#), chargez votre clé et votre certificat existants dans IAM. Pour de plus amples informations sur la création et le chargement des certificats dans IAM, veuillez consulter [Utilisation des certificats de serveur](#) dans le Guide de l'utilisateur IAM.

Pour de plus amples informations sur la configuration HTTPS et l'utilisation des certificats dans Elastic Beanstalk, veuillez consulter [Configuration de HTTPS pour votre environnement Elastic Beanstalk \(p. 792\)](#).

Pour Certificat SSL, choisissez l'ARN de votre certificat SSL. Par exemple, `arn:aws:iam::123456789012:server-certificate/abc/certs/build` ou `arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678`.

**Classic Load Balancer listener**

**Listener port**  
443 ▲▼

**Listener protocol**  
The load balancer transport protocol to use for routing.  
HTTPS ▼

**Instance port**  
The port on which the instance server is listening.  
443 ▲▼

**Instance protocol**  
The protocol to use for routing traffic to backend instances. This must be at the same internet protocol layer as the listener protocol. It also must have the same security level as any other listener using the same instance port as this listener.  
HTTPS ▼

**SSL certificate**  
`arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678` ▼

**Add**

Pour de plus amples informations sur la configuration HTTPS et l'utilisation des certificats dans Elastic Beanstalk, veuillez consulter [Configuration de HTTPS pour votre environnement Elastic Beanstalk \(p. 792\)](#).

## Sessions

Activez ou désactivez la case Permanence de la session activée pour activer ou désactiver les sessions permanentes. Utilisez l'option Durée du cookie pour configurer la durée d'une session permanente, pouvant aller jusqu'à **1000000** secondes. Dans la liste Load balancer ports (Ports de l'équilibrer de charge), sélectionnez les ports écouteurs auxquels la stratégie par défaut (**AWSEB-ELB-StickinessPolicy**) s'applique.

### Sessions

The following settings let you control whether the load balancer routes requests for the same session to the Amazon EC2 instance consistently to the same instance.

Session stickiness enabled

#### Cookie duration

Lifetime of the sticky session cookie between an Amazon EC2 instance and the load balancer.

0  seconds

#### Load balancer ports

List of the listener ports that the default policy (**AWSEB-ELB-StickinessPolicy**) applies to.

*Choose load balancer ports*

80

443

## Équilibrage de charge entre zones

Cochez ou décochez Équilibrage de charge entre plusieurs zones de disponibilité activé pour activer ou désactiver l'équilibrage de charge entre zones.

### Cross-zone load balancing

Load balancing across multiple Availability Zones enabled

## Drainage de la connexion

Activez ou désactivez la case Drainage de la session activée pour activer ou désactiver le drainage de la connexion. Définissez le délai de drainage, pouvant aller jusqu'à **3600** secondes.

The screenshot shows the 'Connection draining' configuration section. It includes a checkbox labeled 'Connection draining enabled', which is unchecked. Below it is a 'Draining timeout' field with the value '20 seconds'. A note states: 'Maximum time that the load balancer maintains connections to an Amazon EC2 instance before forcibly closing connections.'

## Vérification de l'état

Utilisez les paramètres suivants pour configurer les vérifications de l'état de l'équilibrEUR de charge :

- Health check path (Chemin de vérification de l'état) – Chemin d'accès vers lequel l'équilibrEUR de charge envoie les demandes de vérification de l'état. Si ce chemin n'est pas défini, l'équilibrEUR de charge tente d'établir une connexion TCP sur le port 80 pour vérifier l'intégrité.
- Timeout (Délai) – Durée, en secondes, d'attente d'une réponse de la vérification de l'état.
- Interval (Intervalle) – Durée, en secondes, entre les vérifications de l'état d'une instance. L'intervalle doit être supérieur au délai.
- Unhealthy threshold (Seuil de défectuosité), Healthy threshold (Seuil de bonne santé) – Nombre de vérifications de l'état qui doivent échouer ou réussir avant qu'Elastic Load Balancing modifie l'état de santé d'une instance.

The screenshot shows the 'Health check' configuration section of the AWS Elastic Beanstalk developer guide. It includes fields for 'Health check path', 'Timeout', 'Interval', 'Unhealthy threshold', and 'Healthy threshold'. Each field has a value (e.g., 5 seconds, 10 seconds) followed by a dropdown menu.

**Health check**

**Health check path**  
Path to which ELB sends an HTTP GET request to verify instance health.

**Timeout**  
Amount of time to wait for a health check response.  
5  seconds

**Interval**  
Amount of time between health checks of an individual instance. The interval must be greater than the timeout.  
10  seconds

**Unhealthy threshold**  
The number of consecutive health check failures required to designate the instance as unhealthy.  
5  requests

**Healthy threshold**  
The number of consecutive successful health checks required to designate the instance as healthy.  
3  requests

#### Note

La vérification de l'état Elastic Load Balancing n'a pas d'incidence sur le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Les instances dont la vérification de l'état Elastic Load Balancing échoue ne sont pas automatiquement remplacées par Amazon EC2 Auto Scaling, sauf si vous configurez manuellement Amazon EC2 Auto Scaling pour le faire. Pour de plus amples informations, veuillez consulter [Paramètre de vérification de l'état Auto Scaling \(p. 561\)](#).

Pour de plus amples informations sur les vérifications de l'état et leur influence sur l'état global de votre environnement, veuillez consulter [Création de rapports d'intégrité de base \(p. 834\)](#).

## Configuration d'un équilibrer de charge Classic Load Balancer à l'aide de l'interface de ligne de commande EB

L'interface de ligne de commande EB vous invite à choisir un type d'équilibrer de charge lorsque vous exécutez la commande [eb create \(p. 1078\)](#).

```
$ eb create
```

```
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 1):
```

Appuyez sur Entrée pour sélectionner l'option `classic`.

Vous pouvez également spécifier un type d'équilibrer de charge à l'aide de l'option `--elb-type`.

```
$ eb create test-env --elb-type classic
```

## Espaces de noms pour la configuration d'un équilibrer de charge Classic Load Balancer

Vous trouverez les paramètres liés aux équilibreurs Classic Load Balancer dans les espaces de noms suivants :

- [aws:elb:healthcheck \(p. 712\)](#) – Configurez les seuils, l'intervalle de vérification et le délai d'attente pour les vérifications de l'état de l'équilibrer de charge.
- [aws:elasticbeanstalk:application \(p. 697\)](#) – Configurez l'URL de la vérification de l'état.
- [aws:elb:loadbalancer \(p. 713\)](#) – Activez l'équilibrage de charge entre zones. Affectez des groupes de sécurité à l'équilibrer de charge et remplacez le groupe de sécurité par défaut créé par Elastic Beanstalk. Cet espace de noms inclut également des options obsolètes pour la configuration des écouteurs standard et sécurisés, qui ont été remplacées par les options de l'espace de noms `aws:elb:listener`.
- [aws:elb:listener \(p. 714\)](#) – Configurez l'écouteur par défaut sur le port 80, un écouteur sécurisé sur le port 443 ou des écouteurs supplémentaires pour n'importe quel protocole sur n'importe quel port. Si vous spécifiez l'espace de noms `aws:elb:listener`, les paramètres s'appliquent à l'écouteur par défaut sur le port 80. Si vous spécifiez un port (par exemple, `aws:elb:listener:443`), un écouteur est configuré sur ce port.
- [aws:elb:policies \(p. 716\)](#) – Configurez des paramètres supplémentaires pour votre équilibrer de charge. Utilisez les options de cet espace de noms pour configurer des écouteurs sur des ports arbitraires, modifier des paramètres de session permanente supplémentaires et configurer l'équilibrer de charge pour vous connecter en toute sécurité aux instances Amazon EC2.

L'interface de ligne de commande EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Pour de plus amples informations, veuillez consulter [Valeurs recommandées \(p. 660\)](#).

Example .ebextensions/loadbalancer-terminatehttps.config

L'exemple de fichier de configuration suivant crée un écouteur HTTPS sur le port 443, affecte un certificat que l'équilibrer de charge utilise pour mettre la connexion sécurisée hors service, et désactive l'écouteur par défaut sur le port 80. L'équilibrer de charge transmet les demandes déchiffrées aux instances EC2 de votre environnement sur le port HTTP:80.

```
option_settings:
  aws:elb:listener:443:
```

```
ListenerProtocol: HTTPS
SSLCertificateId: arn:aws:acm:us-
east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678
InstancePort: 80
InstanceProtocol: HTTP
aws:elb:listener:
ListenerEnabled: false
```

## Configuration d'un Application Load Balancer

Lorsque vous activez l'[équilibrage de charge \(p. 520\)](#), votre environnement AWS Elastic Beanstalk est équipé d'un équilibrEUR de charge Elastic Load Balancing qui permet de répartir le trafic entre les instances de votre environnement. Elastic Load Balancing prend en charge plusieurs types d'équilibrEUR de charge. Pour en savoir plus, consultez le [Guide de l'utilisateur Elastic Load Balancing](#). Elastic Beanstalk peut créer un équilibrEUR de charge pour vous, ou vous permettre de spécifier un équilibrEUR de charge partagé que vous avez créé.

Cette rubrique décrit la configuration d'un équilibrEUR de charge Application Load Balancer créé par Elastic Beanstalk et dédié à votre environnement. Voir aussi [the section called "ÉquilibrEUR de charge Application Load Balancer partagé" \(p. 591\)](#). Pour de plus amples informations sur la configuration de tous les types d'équilibrEUR de charge pris en charge par Elastic Beanstalk, veuillez consulter [the section called "ÉquilibrEUR de charge" \(p. 562\)](#).

### Note

Vous pouvez choisir le type d'équilibrEUR de charge que votre environnement utilise uniquement lors de la création de l'environnement. Vous pouvez modifier les paramètres pour gérer le comportement de l'équilibrEUR de charge de votre environnement d'exécution, mais pas en changer le type. Vous ne pouvez pas non plus passer d'un équilibrEUR de charge dédié à un équilibrEUR de charge partagé ou inversement.

## Introduction

Un équilibrEUR de charge Application Load Balancer inspecte le trafic au niveau de la couche protocole réseau de l'application pour identifier le chemin d'accès des demandes afin de les diriger vers différentes destinations pour divers chemins d'accès.

Lorsque votre environnement utilise un équilibrEUR de charge Application Load Balancer, Elastic Beanstalk le configure par défaut pour exécuter la même fonction qu'un équilibrEUR de charge Classic Load Balancer. L'écouteur par défaut accepte les demandes HTTP sur le port 80 et les distribue aux instances de votre environnement. Vous pouvez ajouter un écouteur sécurisé sur le port 443 avec un certificat permettant de déchiffrer le trafic HTTPS, configurer le fonctionnement de la vérification de l'état et acheminer les journaux d'accès de l'équilibrEUR de charge vers un compartiment Amazon Simple Storage Service (Amazon S3).

### Note

Contrairement à un équilibrEUR de charge Classic Load Balancer ou Network Load Balancer, un équilibrEUR de charge Application Load Balancer ne peut pas avoir d'écouteurs TCP ou SSL/TLS de couche de transport (couche 4). Il ne prend en charge que les écouteurs HTTP et HTTPS. En outre, il ne peut pas utiliser l'authentification backend pour authentifier les connexions HTTPS entre l'équilibrEUR de charge et les instances backend.

Dans un environnement Elastic Beanstalk, vous pouvez utiliser un équilibrEUR de charge Application Load Balancer pour diriger le trafic de certains chemins d'accès vers un autre processus sur vos instances de serveur web. Avec un équilibrEUR de charge Classic Load Balancer, l'ensemble du trafic vers un écouteur est acheminé vers un seul processus sur les instances de backend. Un équilibrEUR de charge Application Load Balancer vous permet de configurer plusieurs règles sur l'écouteur afin d'acheminer les demandes de

certains chemins d'accès vers d'autres processus de backend. Vous configurez chaque processus avec le port sur lequel le processus écoute.

Par exemple, vous pouvez exécuter un processus de connexion séparément de votre application principale. Tandis que l'application principale sur les instances de votre environnement accepte la plupart des demandes et écoute sur le port 80, votre processus de connexion écoute sur le port 5000 et accepte les demandes du chemin d'accès `/login`. Toutes les demandes entrantes provenant de clients arrivent sur le port 80. Avec un équilibrEUR de charge Application Load Balancer, vous pouvez configurer un seul écouteur pour le trafic entrant sur le port 80, avec deux règles qui acheminent le trafic vers deux processus distincts, selon le chemin d'accès de la requête. Vous ajoutez une règle personnalisée qui achemine le trafic vers `/login` pour le processus de connexion écoutant sur le port 5000. La règle par défaut achemine tout le reste du trafic vers le processus d'application principale à l'écoute sur le port 80.

Une règle d'équilibrEUR de charge Application Load Balancer fait correspondre une demande à un groupe cible. Dans Elastic Beanstalk, un groupe cible est représenté par un processus. Vous pouvez configurer un processus avec un protocole, le un et des paramètres de vérification de l'état. Le processus représente le processus en cours d'exécution sur les instances de votre environnement. Le processus par défaut est un écouteur sur le port 80 du proxy inverse (nginx ou Apache) qui s'exécute devant votre application.

#### Note

En dehors d'Elastic Beanstalk, un groupe cible est mappé à un groupe d'instances. Un écouteur peut utiliser des règles et des groupes cibles pour acheminer le trafic vers différentes instances en fonction du chemin d'accès. Dans Elastic Beanstalk, toutes les instances de votre environnement sont identiques et la distinction s'applique donc aux processus écoutant sur des ports différents.

Un équilibrEUR de charge Classic Load Balancer utilise un chemin de vérification de l'état unique pour l'ensemble de l'environnement. Avec un équilibrEUR de charge Application Load Balancer, chaque processus a un chemin de vérification de l'état distinct, surveillé par l'équilibrEUR de charge et la surveillance améliorée de l'état Elastic Beanstalk.

Pour utiliser un équilibrEUR de charge Application Load Balancer, votre environnement doit se trouver dans un VPC par défaut ou personnalisé et doit avoir un rôle de service avec un ensemble standard d'autorisations. Si vous avez un rôle de service plus ancien, il sera peut-être nécessaire de [mettre à jour ses autorisations \(p. 925\)](#) afin d'inclure `elasticloadbalancing:DescribeTargetHealth` et `elasticloadbalancing:DescribeLoadBalancers`. Pour de plus amples informations sur les équilibrEURS de charge Application Load Balancer, veuillez consulter [Qu'est-ce qu'un équilibrEUR de charge Application Load Balancer ?](#)

#### Note

La vérification de l'état de l'équilibrEUR de charge Application Load Balancer n'utilise pas le chemin d'accès à la vérification de l'état Elastic Beanstalk. Elle utilise le chemin spécifique configuré pour chaque processus séparément.

## Configuration d'un équilibrEUR de charge Application Load Balancer à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer les écouteurs, les processus et les règles d'un équilibrEUR de charge Application Load Balancer lors de la création de l'environnement ou plus tard alors que votre environnement est en cours d'exécution.

Pour configurer un équilibrEUR de charge Application Load Balancer dans la console Elastic Beanstalk lors de la création de l'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements).
3. Choisissez [Create a new environment \(Créer un nouvel environnement\)](#) (p. 442) pour commencer à créer votre environnement.
4. Sur la page principale de l'assistant, avant de choisir Créer un environnement, choisissez Configurer plus d'options.
5. Choisissez le prérglage de configuration Haute disponibilité.  
Sinon, dans la catégorie de configuration Capacité, configurez un type d'environnement avec un Équilibrage de charge. Pour plus d'informations, consultez [Capacity \(p. 450\)](#).
6. Dans la catégorie de configuration Load balancer (ÉquilibrEUR de charge), choisissez Edit (Modifier).
7. Sélectionnez les options Application Load Balancer (ÉquilibrEUR de charge d'application) et Dedicated (Dédieré) si elles ne sont pas déjà sélectionnées.

Elastic Beanstalk > Applications > getting-started-app

## Modify load balancer

### Load balancer type

Application Load Balancer  
Application layer load balancer—routing HTTP and HTTPS traffic based on protocol, port, and route to environment processes.

Classic Load Balancer  
Previous generation — HTTP, HTTPS, and TCP

Network Load Balancer  
Ultra-high performance and static IP addresses for your application.

Dedicated  
Use a load balancer that Elastic Beanstalk creates exclusively for this environment.

Shared  
Use a load balancer that someone in your account can be shared among multiple Elastic Beanstalk environments.

8. Effectuez toutes les modifications de configuration de l'équilibrEUR de charge Application Load Balancer exigées par votre environnement.
9. Choisissez Enregistrer, puis effectuez toutes les autres modifications de configuration exigées par votre environnement.
10. Choisissez Create environment.

Pour configurer un équilibrEUR de charge Application Load Balancer d'un environnement en cours d'exécution dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Load balancer (Équilibrer de charge), choisissez Edit (Modifier).

### Note

Si la catégorie de configuration Load balancer (Équilibrer de charge) ne dispose pas du bouton Edit (Modifier), cela signifie que votre environnement ne dispose pas d'un équilibrer de charge. Pour apprendre à en configurer un, consultez [Changement de type d'environnement \(p. 520\)](#).

5. Effectuez les modifications de configuration de l'équilibrer de charge Application Load Balancer exigées par votre environnement.
6. Choisissez Apply.

### Déclencheurs des équilibrés de charge Application Load Balancer

- [Écouteurs \(p. 577\)](#)
- [Processus \(p. 579\)](#)
- [Règles \(p. 582\)](#)
- [Capture des journaux d'accès \(p. 584\)](#)

## Écouteurs

Utilisez cette liste pour spécifier plusieurs écouteurs pour votre équilibrer de charge. Chaque écouteur achemine le trafic client entrant sur un port spécifié à l'aide d'un protocole spécifié vers un ou plusieurs processus sur vos instances. Initialement, la liste indique l'écouteur par défaut, qui achemine le trafic HTTP entrant sur le port 80 pour un processus nommé default.

The screenshot shows the 'Listeners' section of the AWS Elastic Beanstalk Application Load Balancer configuration. It includes a descriptive text about listeners, a table with one row, and an 'Actions' button.

	Port	Protocol	SSL certificate	Default process
<input type="checkbox"/>	80	HTTP	--	default

### Pour configurer un écouteur existant

1. Cochez la case en regard de son entrée de table, puis choisissez Actions, Modifier.
2. Utilisez la boîte de dialogue Écouteur de l'Application Load Balancer pour modifier les paramètres, puis choisissez Enregistrer.

Pour ajouter un écouteur

1. Choisissez Ajouter un écouteur.
2. Dans la boîte de dialogue Écouteur de l'équilibrer de charge d'application, configurez les paramètres désirés, puis choisissez Ajouter.

Utilisez les paramètres de la boîte de dialogue Écouteur de l'Application Load Balancer pour choisir le port et le protocole sur lesquels l'écouteur écoute le trafic, ainsi que le processus vers lequel acheminer le trafic. Si vous choisissez le protocole HTTPS, configurez les paramètres SSL.

The screenshot shows the 'Application Load Balancer listener' configuration dialog. It has fields for 'Port' (set to 80), 'Protocol' (set to HTTP), and 'Default process' (set to default). At the bottom are 'Cancel' and 'Save' buttons.

**Application Load Balancer listener**

Port  
80

Protocol  
The transport protocol that the load balancer uses for routing incoming traffic from clients.  
HTTP

Default process  
The process to which the listener routes traffic by default, when the message path doesn't match any custom listener rule.  
default

Cancel Save

Pour pouvoir configurer un écouteur HTTPS, assurez-vous que vous disposez d'un certificat SSL valide. Effectuez l'une des actions suivantes :

- Si AWS Certificate Manager (ACM) est [disponible dans votre région AWS](#), créez ou importez un certificat avec ACM. Pour plus d'informations sur une demande de certificat ACM, consultez [Demande de certificat](#) dans le Guide de l'utilisateur AWS Certificate Manager. Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez [Importation de certificats](#) dans le Guide de l'utilisateur AWS Certificate Manager.
- Si ACM n'est pas [disponible dans votre région AWS](#), chargez votre clé et votre certificat existants dans IAM. Pour de plus amples informations sur la création et le chargement des certificats dans IAM, veuillez consulter [Utilisation des certificats de serveur](#) dans le Guide de l'utilisateur IAM.

Pour de plus amples informations sur la configuration HTTPS et l'utilisation des certificats dans Elastic Beanstalk, veuillez consulter [Configuration de HTTPS pour votre environnement Elastic Beanstalk \(p. 792\)](#).

## Processus

Utilisez cette liste pour spécifier les processus pour votre équilibrEUR de charge. Un processus est une cible pour le routage du trafic par les écouteurs. Chaque écouteur achemine le trafic client entrant sur un port spécifié à l'aide d'un protocole spécifié vers un ou plusieurs processus sur vos instances. Initialement, la liste indique le processus par défaut, qui écoute le trafic HTTP entrant sur le port 80.

The screenshot shows a table titled 'Processes' with the following columns: Name, Port, Protocol, and Health check path. There is also an 'Actions' button at the top right. A single row is present in the table, labeled 'default'.

<input type="checkbox"/>	Name	Port	Protocol	HTTP code	Health check path	Actions ▾
<input type="checkbox"/>	default	80	HTTP		/	

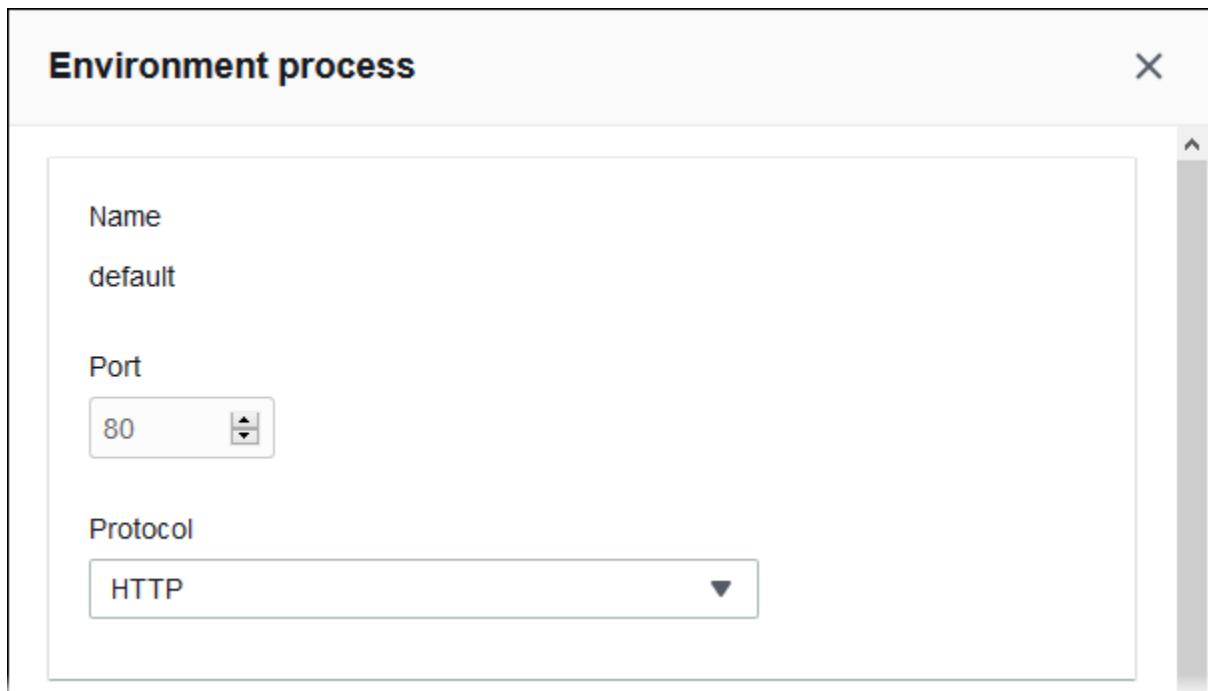
Vous pouvez modifier les paramètres d'un processus existant ou ajouter un nouveau processus. Pour commencer la modification d'un processus de la liste ou l'ajout d'un processus, utilisez la procédure définie pour la [liste d'écouteurs \(p. 577\)](#). La boîte de dialogue Processus d'environnement s'ouvre.

Paramètres de la boîte de dialogue de processus d'environnement de l'équilibrEUR de charge Application Load Balancer

- [Définition \(p. 579\)](#)
- [Vérification de l'état \(p. 580\)](#)
- [Sessions \(p. 582\)](#)

### Définition

Utilisez ces paramètres pour définir le processus : son nom et le port et le protocole sur lesquels il écoute les requêtes.



### Vérification de l'état

Utilisez les paramètres suivants pour configurer les processus des vérifications de l'état :

- HTTP code (Code HTTP) – Code d'état HTTP désignant un processus sain.
- Path (Chemin) – Chemin d'accès de la demande de vérification de l'état du processus.
- Timeout (Délai) – Durée, en secondes, d'attente d'une réponse de la vérification de l'état.
- Interval (Intervalle) – Durée, en secondes, entre les vérifications de l'état d'une instance. L'intervalle doit être supérieur au délai.
- Unhealthy threshold (Seuil de défectuosité), Healthy threshold (Seuil de bonne santé) – Nombre de vérifications de l'état qui doivent échouer ou réussir avant qu'Elastic Load Balancing modifie l'état de santé d'une instance.
- Deregistration delay (Retard d'annulation d'enregistrement) – Délai, en secondes, d'attente de la fin des requêtes actives avant l'annulation de l'enregistrement d'une instance.

## Health check

### HTTP code

HTTP status code of a healthy instance in your environment.

### Path

Path to which the load balancer sends HTTP health check requests.

 /

### Timeout

Amount of time to wait for a health check response.

 5  seconds

### Interval

Amount of time between health checks of an individual instance. The interval must be greater than the timeout.

 15  seconds

### Unhealthy threshold

The number of consecutive health check failures required to designate the instance as unhealthy.

 5  requests

### Healthy threshold

The number of consecutive successful health checks required to designate the instance as healthy.

 3  requests

### Deregistration delay

Amount of time to wait for active requests to complete before deregistering.

 20  seconds

## Note

La vérification de l'état Elastic Load Balancing n'a pas d'incidence sur le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Les instances dont la vérification de l'état Elastic Load Balancing échoue ne sont pas automatiquement remplacées par Amazon EC2 Auto Scaling, sauf si vous configurez manuellement Amazon EC2 Auto Scaling pour le faire. Pour de plus amples informations, veuillez consulter [Paramètre de vérification de l'état Auto Scaling \(p. 561\)](#).

Pour de plus amples informations sur les vérifications de l'état et leur influence sur l'état global de votre environnement, veuillez consulter [Création de rapports d'intégrité de base \(p. 834\)](#).

## Sessions

Activez ou désactivez la case Stratégie de permanence activée pour activer ou désactiver les sessions permanentes. Utilisez l'option Durée du cookie pour configurer la durée d'une session permanente, pouvant aller jusqu'à **604800** secondes.

The screenshot shows the 'Sessions' configuration page. At the top, a section titled 'Sessions' contains the text: 'The following settings let you control whether the load balancer routes requests for the same session to the Amazon EC2 instance with the smallest load, or consistently to the same instance.' Below this, there is a checkbox labeled 'Stickiness policy enabled'. To its right is a numeric input field set to '86400' with up and down arrows for adjustment. At the bottom of the dialog are two buttons: 'Cancel' and 'Save'.

## Règles

Utilisez cette liste pour spécifier des règles d'écouteur personnalisées pour votre équilibrage de charge. Une règle mappe les requêtes que l'écouteur reçoit sur un modèle de chemin spécifique à un processus cible. Chaque écouteur peut avoir plusieurs règles acheminant les requêtes sur différents chemins à différents processus sur vos instances.

Les règles ont des priorités numériques qui déterminent la priorité selon laquelle elles sont appliquées aux demandes entrantes. Pour chaque nouvel écouteur que vous ajoutez, Elastic Beanstalk ajoute une règle par défaut qui achemine l'ensemble du trafic de l'écouteur au processus par défaut. La priorité de la règle par défaut est la plus basse ; elle n'est appliquée que si aucune autre règle pour le même écouteur ne

correspond à la requête entrante. Initialement, si vous n'avez pas ajouté de règles personnalisées, la liste est vide. Les règles par défaut de tous les écouteurs ne sont pas affichées.

The screenshot shows the 'Rules' section of the AWS Elastic Beanstalk Application Load Balancer configuration. At the top, a note states: 'Your load balancer routes requests to environment processes based on rules. Rules are evaluated by priority in ascending numerical order. Elastic Beanstalk configures a default rule for each listener. Each default rule routes all traffic to the default process associated with each listener. The default rule has the highest priority among all rules of that listener. If a request doesn't match the conditions for any other rule, a default rule routes the request to the default process.' Below this is a table header with columns: Name, Listener port, Priority, Host headers, and Path patterns. A message below the table says: 'No additional listener rules are currently configured. Choose Add rule to add a listener rule.' An 'Actions' button is visible at the top right of the page.

Vous pouvez modifier les paramètres d'une règle existante ou ajouter une nouvelle règle. Pour commencer la modification d'une règle de la liste ou l'ajout d'une règle, utilisez la procédure définie pour la [liste d'écouteurs \(p. 577\)](#). La boîte de dialogue Règle d'écouteur s'ouvre, avec les paramètres suivants :

- Name (Nom) – Nom de la règle.
- Listener port (Port d'écoute) – Port d'écoute auquel la règle s'applique.
- Priority (Priorité) – Priorité de la règle. Une valeur de priorité plus faible a une priorité plus élevée. Les priorités des règles d'un écouteur doivent être uniques.
- Match conditions (Conditions de correspondance) – Liste des conditions d'URL de requête auxquelles la règle s'applique. Il existe deux types de conditions : HostHeader (la partie domaine de l'URL) et PathPattern (la partie chemin de l'URL). Vous pouvez ajouter jusqu'à cinq conditions. Chaque valeur de condition comporte jusqu'à 128 caractères et peut inclure des caractères génériques.
- Process (Processus) – Processus auquel l'équilibrage de charge achemine les demandes qui correspondent à la règle.

Lorsque vous modifiez une règle existante, vous ne pouvez pas changer son Nom ni son Port d'écoute.

### Listener rule

Name: images

Listener port: 80

Priority: 1

Match conditions:

Type: PathPattern Value: /images/\* Remove

Add condition

Process: images

Cancel Add

## Capture des journaux d'accès

Utilisez ces paramètres pour configurer Elastic Load Balancing afin qu'il capture les journaux avec des informations détaillées sur les demandes envoyées à votre équilibrEUR de charge Application Load Balancer. La capture des journaux d'accès est désactivée par défaut. Lorsque l'option Store logs (Stocker des journaux) est activée, Elastic Load Balancing stocke les journaux dans le compartiment S3 que vous configurez. Le paramètre Prefix (Préfixe) spécifie un dossier de niveau supérieur dans le compartiment pour les journaux. Elastic Load Balancing place les journaux dans un dossier nommé AWSLogs sous votre préfixe. Si vous ne spécifiez pas de préfixe, Elastic Load Balancing place son dossier au niveau racine du compartiment.

### Note

Si le compartiment Amazon S3 que vous configurez pour la capture des journaux d'accès n'est pas le compartiment créé par Elastic Beanstalk pour votre compte, veillez à ajouter une

stratégie utilisateur avec les autorisations appropriées à vos utilisateurs AWS Identity and Access Management Identity and Access Management (IAM). Les [stratégies utilisateur gérées \(p. 946\)](#) fournies par Elastic Beanstalk ne couvrent que les autorisations pour les ressources gérées par Elastic Beanstalk.

Pour plus d'informations sur les journaux d'accès, y compris les autorisations et d'autres exigences, consultez la section [Journaux d'accès pour votre Équilibrer de charge d'application](#).

The screenshot shows the 'Access log files' configuration page. It includes a description of capturing logs with detailed information about requests sent to the Load Balancer, a 'more' link, and sections for 'Store logs' (with a note about standard Amazon S3 charges), 'S3 bucket' (with a dropdown menu showing '-- Choose an Amazon S3 bucket --' and a 'Choose a bucket' button), and 'Prefix' (with a note about logical hierarchy and a text input field).

## Exemple : Équilibrer de charge Application Load Balancer avec un écouteur sécurisé et deux processus

Dans cet exemple, votre application exige le chiffrement du trafic de bout en bout et un processus distinct pour le traitement des requêtes administratives.

Pour configurer l'équilibrer de charge Application Load Balancer de votre environnement afin qu'il réponde à ces exigences, vous devez supprimer l'écouteur par défaut, ajouter un écouteur HTTPS, indiquer le processus par défaut qui écoute le port 443 sur HTTPS et ajouter un processus et une règle d'écouteur pour le trafic d'administration sur un autre chemin.

Pour configurer l'équilibrer de charge de cet exemple

1. Ajoutez un écouteur sécurisé. Pour Port, entrez **443**. Pour Protocole, sélectionnez **HTTPS**. Pour Certificat SSL, choisissez l'ARN de votre certificat SSL. Par exemple, **arn:aws:iam::123456789012:server-certificate/abc/certs/build** ou **arn:aws:acm:useast-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678**.

Pour Processus par défaut, gardez **default** sélectionné.

### Application Load Balancer listener

Port  
443

Protocol  
The transport protocol that the load balancer uses for routing incoming traffic from clients.  
HTTPS

SSL certificate  
arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678 C

SSL policy  
The Secure Sockets Layer (SSL) negotiation configuration, known as a security policy, that this load balancer uses to negotiate SSL connections with clients.  
ELBSecurityPolicy-2016-08

Default process  
The process to which the listener routes traffic by default, when the message path doesn't match any custom listener rule.  
default

**Add**

Vous voyez désormais votre écouteur supplémentaires dans la liste.

<input type="checkbox"/>	Port	Protocol	SSL certificate
<input type="checkbox"/>	80	HTTP	--
<input type="checkbox"/>	443	Pending create	arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678

2. Désactivez l'écouteur HTTP du port 80 par défaut. Pour l'écouteur par défaut, désactivez l'option Activé.

	Port	Protocol	SSL certificate
<input type="checkbox"/>	80	HTTP	--
<input type="checkbox"/>	443	Pending create	HTTPS arn:aws:acm:us-east-2:123456789012:certificate/12345678-12ab-34cd-56ef-12345678

3. Configurez le processus par défaut sur HTTPS. Sélectionnez le processus par défaut, puis, pour Actions, choisissez Edit (Modifier). Pour Port, entrez **443**. Pour Protocole, sélectionnez **HTTPS**.

### Environment process

Name  
default

Port  
443

Protocol  
HTTPS

4. Ajoutez un processus d'administration. Pour Nom, entrez **admin**. Pour Port, entrez **443**. Pour Protocole, sélectionnez **HTTPS**. Sous Vérification de l'état, pour Chemin, tapez **/admin**.

### Environment process

Name: admin

Port: 443

Protocol: HTTPS

### Health check

HTTP code: 200

Path: /admin

5. Ajoutez une règle pour le trafic d'administration. Pour Nom, entrez **admin**. Pour Port d'écoute, entrez **443**. Pour les conditions de correspondance, ajoutez un PathPattern avec la valeur **/admin/\***. Pour Processus, sélectionnez **admin**.

**Listener rule**

**Name**  
admin

**Listener port**  
443 ▾

**Priority**  
Evaluated in ascending numerical order. Must be unique across all rules.  
1 ▾

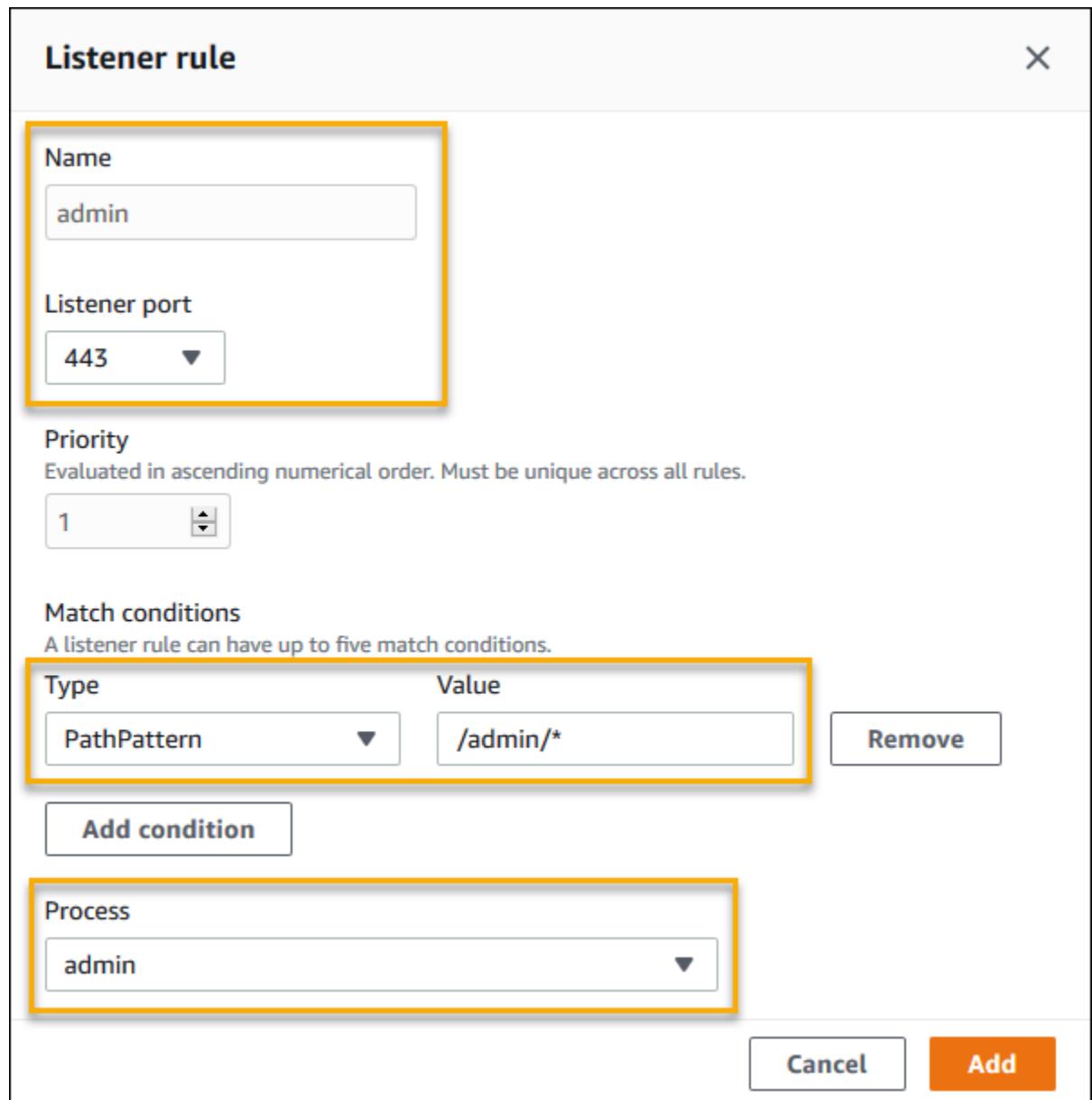
**Match conditions**  
A listener rule can have up to five match conditions.

Type	Value	Remove
PathPattern	/admin/*	Remove

Add condition

**Process**  
admin

Cancel Add



## Configuration d'un équilibrEUR de charge Application Load Balancer à l'aide de l'interface de ligne de commande EB

L'interface de ligne de commande EB vous invite à choisir un type d'équilibrEUR de charge lorsque vous exécutez la commande [eb create \(p. 1078\)](#).

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF
Select a load balancer type
```

```
1) classic
2) application
3) network
(default is 2):
```

Vous pouvez également spécifier un type d'équilibrage de charge à l'aide de l'option `--elb-type`.

```
$ eb create test-env --elb-type application
```

## Espaces de noms Application Load Balancer

Vous trouverez les paramètres liés aux équilibreurs Application Load Balancers dans les espaces de noms suivants :

- [aws:elasticbeanstalk:environment \(p. 700\)](#) – Choisissez le type d'équilibrage de charge pour votre environnement. La valeur d'un équilibrage de charge Application Load Balancer est `application`.  
Vous ne pouvez pas définir cette option dans les fichiers de configuration ([Ebextensions \(p. 737\)](#)).
- [aws:elbv2:loadbalancer \(p. 723\)](#) – Configurez les journaux d'accès et autres paramètres s'appliquant à l'équilibrage de charge Application Load Balancer dans son ensemble.
- [aws:elbv2:listener \(p. 720\)](#) – Configurez les écouteurs sur l'équilibrage de charge Application Load Balancer. Ces paramètres sont mappés aux paramètres de `aws:elb:listener` pour les équilibreurs de charge Classic Load Balancers.
- [aws:elbv2:listenerrule \(p. 721\)](#) – Configurez les règles qui acheminent le trafic vers différents processus, selon le chemin de la demande. Les règles sont spécifiques aux équilibreurs de charge Application Load Balancer.
- [aws:elasticbeanstalk:environment:process \(p. 701\)](#) – Configurez les vérifications de l'état et spécifiez le port et le protocole des processus qui s'exécutent sur les instances de votre environnement. Les paramètres de port et de protocole sont mappés sur ceux de l'instance dans `aws:elb:listener` pour un écouteur sur un équilibrage de charge Classic Load Balancer. Les paramètres de vérification de l'état sont mappés sur les paramètres des espaces de noms `aws:elb:healthcheck` et `aws:elasticbeanstalk:application`.

### Example .ebextensions/alb-access-logs.config

Le fichier de configuration suivant permet le chargement de journaux d'accès pour un environnement comportant un équilibrage de charge Application Load Balancer.

```
option_settings:
  aws:elbv2:loadbalancer:
    AccessLogsS3Bucket: DOC-EXAMPLE-BUCKET
    AccessLogsS3Enabled: 'true'
    AccessLogsS3Prefix: beanstalk-alb
```

### Example .ebextensions/alb-default-process.config

Le fichier de configuration suivant modifie les paramètres de vérification de l'état et de permanence sur le processus par défaut.

```
option_settings:
  aws:elasticbeanstalk:environment:process:default:
    DeregistrationDelay: '20'
    HealthCheckInterval: '15'
    HealthCheckPath: /
    HealthCheckTimeout: '5'
    HealthyThresholdCount: '3'
```

```
UnhealthyThresholdCount: '5'  
Port: '80'  
Protocol: HTTP  
StickinessEnabled: 'true'  
StickinessLBCookieDuration: '43200'
```

#### Example .ebextensions/alb-secure-listener.config

Le fichier de configuration suivant ajoute un écouteur sécurisé et un processus correspondant sur le port 443.

```
option_settings:  
aws:elbv2:listener:443:  
    DefaultProcess: https  
    ListenerEnabled: 'true'  
    Protocol: HTTPS  
    SSLCertificateArns: arn:aws:acm:us-east-2:123456789012:certificate/21324896-0fa4-412b-  
bf6f-f362d6eb6dd7  
aws:elasticbeanstalk:environment:process:https:  
    Port: '443'  
    Protocol: HTTPS
```

#### Example .ebextensions/alb-admin-rule.config

Le fichier de configuration suivant ajoute un écouteur sécurisé avec une règle qui achemine le trafic dont le chemin de la demande est /admin vers un processus nommé admin qui écoute sur le port 4443.

```
option_settings:  
aws:elbv2:listener:443:  
    DefaultProcess: https  
    ListenerEnabled: 'true'  
    Protocol: HTTPS  
    Rules: admin  
    SSLCertificateArns: arn:aws:acm:us-east-2:123456789012:certificate/21324896-0fa4-412b-  
bf6f-f362d6eb6dd7  
aws:elasticbeanstalk:environment:process:https:  
    Port: '443'  
    Protocol: HTTPS  
aws:elasticbeanstalk:environment:process:admin:  
    HealthCheckPath: /admin  
    Port: '4443'  
    Protocol: HTTPS  
aws:elbv2:listenerrule:admin:  
    PathPatterns: /admin/*  
    Priority: 1  
    Process: admin
```

## Configuration d'un équilibrer de charge Application Load Balancer partagé

Lorsque vous activez l'[équilibrage de charge \(p. 520\)](#), votre environnement AWS Elastic Beanstalk est équipé d'un équilibrer de charge Elastic Load Balancing qui permet de répartir le trafic entre les instances de votre environnement. Elastic Load Balancing prend en charge plusieurs types d'équilibrer de charge. Pour en savoir plus, consultez le [Guide de l'utilisateur Elastic Load Balancing](#). Elastic Beanstalk peut créer un équilibrer de charge pour vous, ou vous permettre de spécifier un équilibrer de charge partagé que vous avez créé.

Cette rubrique décrit la configuration d'un équilibrer de charge [Application Load Balancer](#) partagé que vous créez et associez à votre environnement. Voir aussi [the section called "Application Load](#)

**Balancer” (p. 574).** Pour de plus amples informations sur la configuration de tous les types d'équilibrer de charge pris en charge par Elastic Beanstalk, veuillez consulter [Équilibrer de charge pour votre environnement Elastic Beanstalk \(p. 562\)](#).

#### Note

Vous pouvez choisir le type d'équilibrer de charge que votre environnement utilise uniquement lors de la création de l'environnement. Vous pouvez modifier les paramètres pour gérer le comportement de l'équilibrer de charge de votre environnement d'exécution, mais pas en changer le type. Vous ne pouvez pas non plus passer d'un équilibrer de charge dédié à un équilibrer de charge partagé ou inversement.

## Introduction

Un équilibrer de charge partagé est un équilibrer de charge que vous créez et gérez vous-même à l'aide du service Amazon Elastic Compute Cloud (Amazon EC2), puis que vous utilisez dans plusieurs environnements Elastic Beanstalk.

Lorsque vous créez un environnement de mise à l'échelle à charge équilibrée et que vous choisissez d'utiliser un équilibrer de charge Application Load Balancer, Elastic Beanstalk crée un équilibrer de charge dédié à votre environnement par défaut. Pour comprendre ce qu'est un équilibrer de charge Application Load Balancer et comment il fonctionne dans un environnement Elastic Beanstalk, veuillez consulter l'[introduction \(p. 574\)](#) à la configuration d'un équilibrer de charge Application Load Balancer pour Elastic Beanstalk.

Dans certaines situations, vous pouvez économiser le coût d'avoir plusieurs équilibreurs de charge dédiés. Cela peut être utile lorsque vous avez plusieurs environnements, par exemple, si votre application est une suite de microservices au lieu d'un service monolithique. Dans de tels cas, vous pouvez choisir d'utiliser un équilibrer de charge partagé.

Pour utiliser un équilibrer de charge partagé, commencez par le créer dans Amazon EC2, puis ajoutez un ou plusieurs écouteurs. Lors de la création d'un environnement Elastic Beanstalk, vous fournissez ensuite l'équilibrer de charge et choisissez un port d'écoute. Elastic Beanstalk associe l'écouteur au processus par défaut dans votre environnement. Vous pouvez ajouter des règles d'écouteur personnalisées pour acheminer le trafic à partir d'en-têtes et de chemins d'accès d'hôte spécifiques vers d'autres processus d'environnement.

Elastic Beanstalk ajoute une balise à l'équilibrer de charge partagé. Le nom de la balise est `elasticbeanstalk:shared-elb-environment-count`, et sa valeur est le nombre d'environnements partageant cet équilibrer de charge.

L'utilisation d'un équilibrer de charge partagé est différente de l'utilisation d'un équilibrer dédié de plusieurs façons.

Regarding	Équilibrer de charge Application Load Balancer dédié	Équilibrer de charge Application Load Balancer partagé
Gestion	Elastic Beanstalk crée et gère les écouteurs, les règles d'écoute et les processus (groupes cibles) de l'équilibrer de charge. Elastic Beanstalk les supprime également lorsque vous arrêtez votre environnement. Elastic Beanstalk peut définir la capture des journaux d'accès de l'équilibrer de charge, si vous choisissez cette option.	Vous créez et gérez l'équilibrer de charge et les écouteurs en dehors d'Elastic Beanstalk. Elastic Beanstalk crée et gère une règle par défaut et un processus par défaut, et vous pouvez ajouter des règles et des processus. Elastic Beanstalk supprime les règles et processus d'écoute ajoutés lors de la création de l'environnement.

Regarding	Équilibrer de charge Application Load Balancer dédié	Équilibrer de charge Application Load Balancer partagé
Règles d'un écouteur	Elastic Beanstalk crée une règle par défaut pour chaque écouteur, pour acheminer l'ensemble du trafic vers le processus par défaut de l'écouteur.	<p>Elastic Beanstalk associe une règle par défaut uniquement à un écouteur de port 80, le cas échéant. Si vous choisissez un autre port d'écoute par défaut, vous devez lui associer la règle par défaut (la console Elastic Beanstalk et l'interface de ligne de commande EB le font pour vous).</p> <p>Pour résoudre les conflits de condition de règle d'écoute entre les environnements partageant l'équilibrer de charge, Elastic Beanstalk ajoute le CNAME de l'environnement à la règle d'écoute en tant que condition d'en-tête d'hôte.</p> <p>Elastic Beanstalk traite les paramètres de priorité de règle comme relatifs dans les environnements partageant l'équilibrer de charge et les mappe aux priorités absolues lors de la création.</p>
Groupes de sécurité	Elastic Beanstalk crée un groupe de sécurité par défaut et l'attache à l'équilibrer de charge.	<p>Vous pouvez configurer un ou plusieurs groupes de sécurité à utiliser pour l'équilibrer de charge. Si vous ne le faites pas, Elastic Beanstalk vérifie si un groupe de sécurité existant géré par Elastic Beanstalk est déjà attaché à l'équilibrer de charge. Si ce n'est pas le cas, Elastic Beanstalk crée un groupe de sécurité et l'attache à l'équilibrer de charge. Elastic Beanstalk supprime ce groupe de sécurité lorsque le dernier environnement partageant l'équilibrer de charge s'arrête.</p>
Mises à jour	<p>Vous pouvez mettre à jour votre équilibrer de charge Application Load Balancer après la création de l'environnement. Vous pouvez modifier les écouteurs, les règles des écouteurs et les processus. Vous pouvez configurer la capture du journal d'accès de l'équilibrer de charge.</p>	<p>Vous ne pouvez pas utiliser Elastic Beanstalk pour configurer la capture du journal d'accès dans votre équilibrer de charge Application Load Balancer, et vous ne pouvez pas mettre à jour les écouteurs et les règles d'écoute après la création de l'environnement. Vous ne pouvez mettre à jour que les processus (groupes cibles). Pour configurer la capture du journal d'accès et mettre à jour les écouteurs et les règles d'écoute, utilisez Amazon EC2.</p>

## Configuration d'un équilibrer de charge Application Load Balancer partagé à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer un équilibrer de charge Application Load Balancer partagé lors de la création de l'environnement. Vous pouvez sélectionner l'un des équilibreurs de charge partageables de votre compte à utiliser dans l'environnement, sélectionner le port de l'écouteur par défaut et configurer d'autres processus et règles de l'écouteur.

Vous ne pouvez pas modifier la configuration de votre équilibrer de charge Application Load Balancer partagé dans la console Application Load Balancer une fois votre environnement créé. Pour configurer les écouteurs, les règles d'écoute, les processus (groupes cibles) et la capture du journal d'accès, utilisez Amazon EC2.

Pour configurer un équilibrer de charge Application Load Balancer dans la console Elastic Beanstalk lors de la création de l'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements).
3. Choisissez [Create a new environment \(Créer un nouvel environnement\) \(p. 442\)](#) pour commencer à créer votre environnement.
4. Sur la page principale de l'assistant, avant de choisir Crée un environnement, choisissez Configurer plus d'options.
5. Choisissez le prérglage de configuration Haute disponibilité.  
Sinon, dans la catégorie de configuration Capacité, configurez un type d'environnement avec un Équilibrage de charge. Pour plus d'informations, consultez [Capacity \(p. 450\)](#).
6. Dans la catégorie de configuration Load balancer (Équilibrer de charge), choisissez Edit (Modifier).
7. Sélectionnez l'option Application Load Balancer (Équilibrer de charge d'application), si elle n'est pas déjà sélectionnée, puis sélectionnez l'option Shared (Partagé).

Elastic Beanstalk > Applications > getting-started-app

## Modify load balancer

### Load balancer type

#### Application Load Balancer

Application layer load balancer—routing HTTP and HTTPS traffic based on protocol, port, and route to environment processes.

#### Classic Load Balancer

*Previous generation* — HTTP, HTTPS, and TCP

#### Network Load Balancer

Ultra-high performance and static IP addresses for your application.

#### Dedicated

Use a load balancer that Elastic Beanstalk creates exclusively for this environment.

#### Shared

Use a load balancer that someone in your account can share among multiple Elastic Beanstalk environments.

8. Effectuez toutes les modifications de configuration de l'équilibrer de charge Application Load Balancer partagé exigées par votre environnement.
9. Choisissez Enregistrer, puis effectuez toutes les autres modifications de configuration exigées par votre environnement.
10. Choisissez Create environment.

Déclencheurs des équilibrer de charge Application Load Balancer partagé

- [Équilibrer de charge Application Load Balancer partagé \(p. 595\)](#)

- [Processus \(p. 595\)](#)
- [Règles \(p. 599\)](#)

## Équilibrer de charge Application Load Balancer partagé

Utilisez cette section pour choisir un équilibrer de charge Application Load Balancer partagé pour votre environnement et configurer le routage du trafic par défaut.

Avant de pouvoir configurer un équilibrer de charge Application Load Balancer partagé ici, utilisez Amazon EC2 pour définir au moins un équilibrer de charge Application Load Balancer pour le partage, avec au moins un écouteur, dans votre compte. Si vous ne l'avez pas déjà fait, vous pouvez choisir Manage load balancers (Gérer les équilibreurs de charge). Elastic Beanstalk ouvre la console Amazon EC2 dans un nouvel onglet de navigateur.

Une fois que vous avez terminé de configurer des équilibreurs de charge partagés en dehors d'Elastic Beanstalk, configurez les paramètres suivants dans cette section de la console :

- Load balancer ARN (ARN de l'équilibrer de charge) – Équilibrer de charge partagé à utiliser dans cet environnement. Sélectionnez dans une liste d'équilibreurs de charge ou entrez un nom de ressource Amazon (ARN) d'équilibrage de charge.
- Default listener port (Port d'écoute par défaut) – Port du processus d'écoute que l'équilibrer de charge partagé écoute. Faites votre choix dans une liste de ports d'écouteurs existants. Le trafic de cet écouteur avec le CNAME de l'environnement dans l'en-tête hôte est routé vers un processus par défaut dans cet environnement.

The screenshot shows the 'Shared Application Load Balancer' configuration page. It includes a 'Manage load balancers' button, a search bar for 'Load balancer ARN' containing 'arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/example-sh...', and a dropdown for 'Default listener' set to '80 (HTTP)'.

## Processus

Utilisez cette liste pour spécifier les processus de votre équilibrer de charge. Un processus est une cible pour le routage du trafic par les écouteurs. Initialement, la liste affiche le processus par défaut, qui reçoit le trafic de l'écouteur par défaut.

Processes					
For each environment process, you can specify the protocol and port that the load balancer uses to route requests to the process. You the load balancer performs process health checks.					
<b>Actions ▾</b>					
	Name	Port	Protocol	HTTP code	Health check path
<input type="checkbox"/>	default	80	HTTP		/

Pour configurer un processus existant

1. Cochez la case en regard de son entrée de table, puis choisissez Actions, Modifier.
2. Utilisez la boîte de dialogue du processus Environnement pour modifier les paramètres, puis choisissez Enregistrer.

Pour ajouter un processus

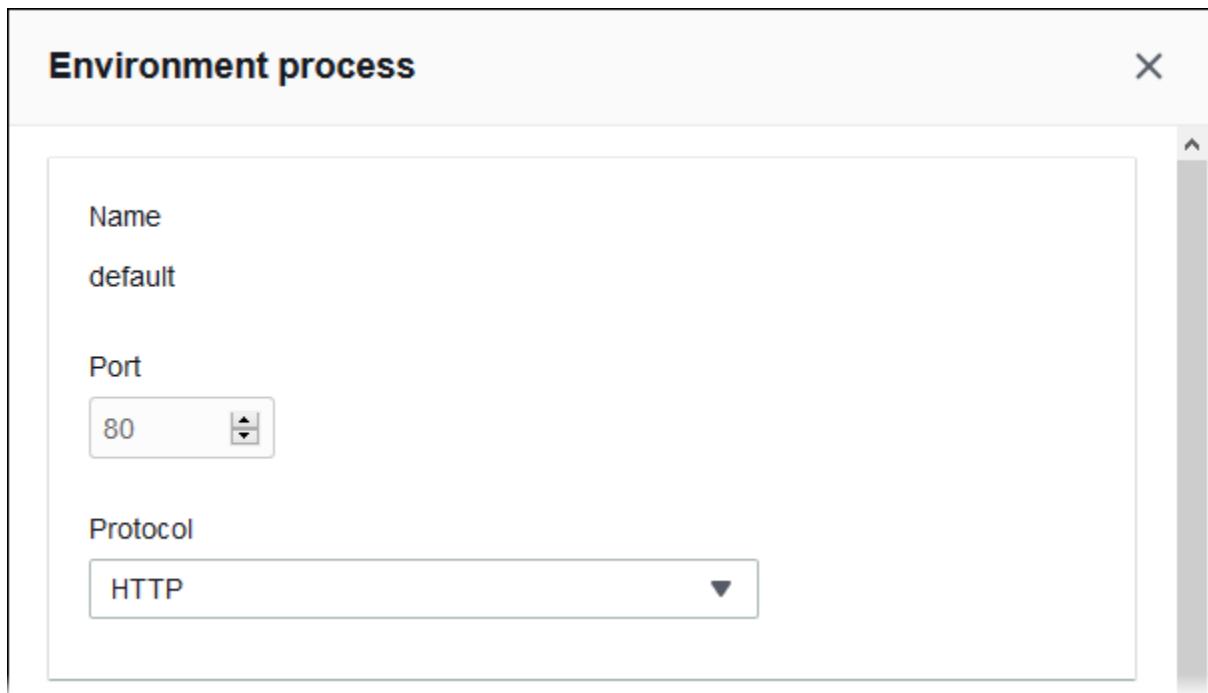
1. Choisissez Ajouter un processus.
2. Dans la boîte de dialogue Processus d'environnement, configurez les paramètres souhaités, puis choisissez Ajouter.

Paramètres de la boîte de dialogue de processus d'environnement de l'équilibrer de charge Application Load Balancer

- [Définition \(p. 579\)](#)
- [Vérification de l'état \(p. 597\)](#)
- [Sessions \(p. 599\)](#)

## Définition

Utilisez ces paramètres pour définir le processus : son nom et le port et le protocole sur lesquels il écoute les requêtes.



### Vérification de l'état

Utilisez les paramètres suivants pour configurer les processus des vérifications de l'état :

- HTTP code (Code HTTP) – Code d'état HTTP désignant un processus sain.
- Path (Chemin) – Chemin d'accès de la demande de vérification de l'état du processus.
- Timeout (Délai) – Durée, en secondes, d'attente d'une réponse de la vérification de l'état.
- Interval (Intervalle) – Durée, en secondes, entre les vérifications de l'état d'une instance. L'intervalle doit être supérieur au délai.
- Unhealthy threshold (Seuil de défectuosité), Healthy threshold (Seuil de bonne santé) – Nombre de vérifications de l'état qui doivent échouer ou réussir avant qu'Elastic Load Balancing modifie l'état de santé d'une instance.
- Deregistration delay (Retard d'annulation d'enregistrement) – Délai, en secondes, d'attente de la fin des requêtes actives avant l'annulation de l'enregistrement d'une instance.

## Health check

### HTTP code

HTTP status code of a healthy instance in your environment.

### Path

Path to which the load balancer sends HTTP health check requests.

 /

### Timeout

Amount of time to wait for a health check response.

 5  seconds

### Interval

Amount of time between health checks of an individual instance. The interval must be greater than the timeout.

 15  seconds

### Unhealthy threshold

The number of consecutive health check failures required to designate the instance as unhealthy.

 5  requests

### Healthy threshold

The number of consecutive successful health checks required to designate the instance as healthy.

 3  requests

### Deregistration delay

Amount of time to wait for active requests to complete before deregistering.

 20  seconds

## Note

La vérification de l'état Elastic Load Balancing n'a pas d'incidence sur le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Les instances dont la vérification de l'état Elastic Load Balancing échoue ne sont pas automatiquement remplacées par Amazon EC2 Auto Scaling, sauf si vous configurez manuellement Amazon EC2 Auto Scaling pour le faire. Pour de plus amples informations, veuillez consulter [Paramètre de vérification de l'état Auto Scaling \(p. 561\)](#).

Pour de plus amples informations sur les vérifications de l'état et leur influence sur l'état global de votre environnement, veuillez consulter [Création de rapports d'intégrité de base \(p. 834\)](#).

## Sessions

Activez ou désactivez la case Stratégie de permanence activée pour activer ou désactiver les sessions permanentes. Utilisez l'option Durée du cookie pour configurer la durée d'une session permanente, pouvant aller jusqu'à **604800** secondes.

The screenshot shows the 'Sessions' configuration page. At the top, there is a heading 'Sessions' followed by a descriptive text: 'The following settings let you control whether the load balancer routes requests for the same session to the Amazon EC2 instance with the smallest load, or consistently to the same instance.' Below this, there is a section titled 'Stickiness policy enabled' with a checkbox labeled 'Stickiness policy enabled'. Underneath it, there is a section titled 'Cookie duration' with a text input field containing '86400' and a dropdown arrow. At the bottom right of the dialog box are two buttons: 'Cancel' and 'Save'.

## Règles

Utilisez cette liste pour spécifier des règles d'écouteur personnalisées pour votre équilibrer de charge. Une règle mappe les requêtes que l'écouteur reçoit sur un modèle de chemin spécifique à un processus cible. Chaque écouteur peut avoir plusieurs règles, acheminer les demandes sur différents chemins vers différents processus sur des instances des différents environnements partageant l'écouteur.

Les règles ont des priorités numériques qui déterminent la priorité selon laquelle elles sont appliquées aux demandes entrantes. Elastic Beanstalk ajoute une règle par défaut qui achemine l'ensemble du trafic de l'écouteur par défaut vers le processus par défaut de votre nouvel environnement. La priorité de la règle par défaut est la plus basse ; elle n'est appliquée que si aucune autre règle pour le même écouteur ne

correspond à la requête entrante. Initialement, si vous n'avez pas ajouté de règles personnalisées, la liste est vide. La règle par défaut n'est pas affichée.

The screenshot shows the 'Rules' section of the AWS Elastic Beanstalk configuration interface. At the top, there is a note about shared load balancer environment rules. Below this, a table lists listener rules. The table has columns for Name, Listener port, Priority, Host headers, and Path patterns. A message indicates that no additional listener rules are currently configured, and a button allows adding a new rule.

**Rules**

Your load balancer routes requests to environment processes based on rules. Rules are evaluated by priority in ascending numerical order. If your environment has existing rules configured, this environment's rules are adjusted to have lower priority than existing rules. You can manage rules for environments in the EC2 console.

Elastic Beanstalk configures a default rule for this environment. This rule routes all traffic from the default listener on port 80 to the default process. The default rule has the highest priority among all rules of this environment. If a request doesn't match the conditions for any other rule, the default rule routes it to the default process.

**Shared load balancer environment rules**

After environment creation, you can't add or edit rules for this environment using Elastic Beanstalk. When you delete the environment, listener rules created outside of Elastic Beanstalk aren't automatically removed by Elastic Beanstalk.

Name	Listener port	Priority	Host headers	Path patterns
No additional listener rules are currently configured.				
Choose Add rule to add a listener rule.				

Vous pouvez modifier les paramètres d'une règle existante ou ajouter une nouvelle règle. Pour commencer la modification d'une règle de la liste ou l'ajout d'une règle, utilisez la procédure définie pour la [liste d'écouteurs \(p. 595\)](#). La boîte de dialogue Règle d'écouteur s'ouvre, avec les paramètres suivants :

- Name (Nom) – Nom de la règle.
- Listener port (Port d'écoute) – Port d'écoute auquel la règle s'applique.
- Priority (Priorité) – Priorité de la règle. Une valeur de priorité plus faible a une priorité plus élevée. Les priorités des règles d'un écouteur doivent être uniques. Elastic Beanstalk traite les priorités des règles comme relatives dans les environnements de partage et les mappe aux priorités absolues lors de la création.
- Match conditions (Conditions de correspondance) – Liste des conditions d'URL de requête auxquelles la règle s'applique. Il existe deux types de conditions : HostHeader (la partie domaine de l'URL) et PathPattern (la partie chemin de l'URL). Une condition est réservée au sous-domaine d'environnement et vous pouvez ajouter jusqu'à quatre conditions. Chaque valeur de condition a jusqu'à 128 caractères et peut inclure des caractères génériques.
- Process (Processus) – Processus auquel l'équilibrer de charge achemine les demandes qui correspondent à la règle.

### Listener rule

Name: images

Listener port: 80

Priority: 1

Match conditions:

A listener rule can have up to five match conditions.

Type	Value	Remove
PathPattern	/images/*	Remove

Add condition

Process: images

Cancel Add

## Exemple : utilisation d'un équilibrer de charge Application Load Balancer partagé pour une application sécurisée basée sur des microservices

Dans cet exemple, votre application se compose de plusieurs microservices, chacun implémenté en tant qu'environnement Elastic Beanstalk. En outre, vous avez besoin d'un chiffrement du trafic de bout en bout. Nous allons démontrer l'un des environnements de microservices, qui a un processus principal pour les demandes des utilisateurs et un processus distinct pour le traitement des demandes administratives.

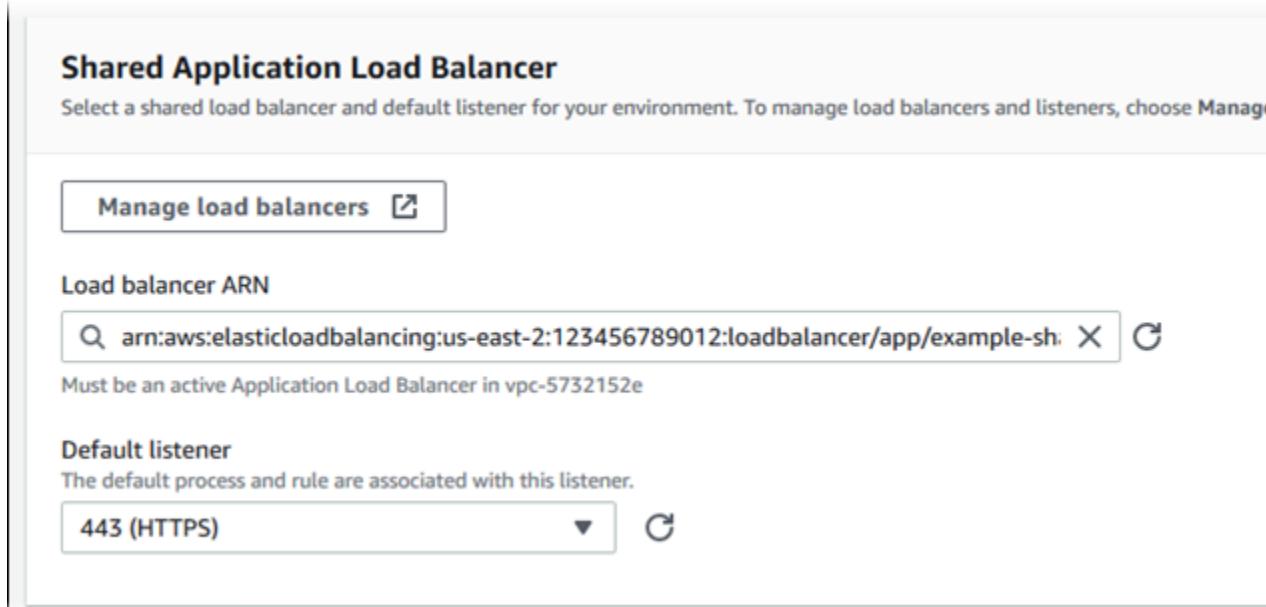
Pour répondre à ces exigences, utilisez Amazon EC2 pour créer un équilibrer de charge Application Load Balancer que vous partagerez entre vos microservices. Ajoutez un écouteur sécurisé sur le port 443 et le protocole HTTPS. Ajoutez ensuite plusieurs certificats SSL à l'écouteur, un par domaine de microservice. Pour de plus amples informations sur la création de l'équilibrer de charge Application Load Balancer et

de l'écouteur sécurisé, veuillez consulter [Création d'un équilibrer de charge Application Load Balancer](#) et [Création d'un écouteur HTTPS pour votre équilibrer de charge Application Load Balancer](#) dans le Guide de l'utilisateur des équilibreurs de charge Application Load Balancer.

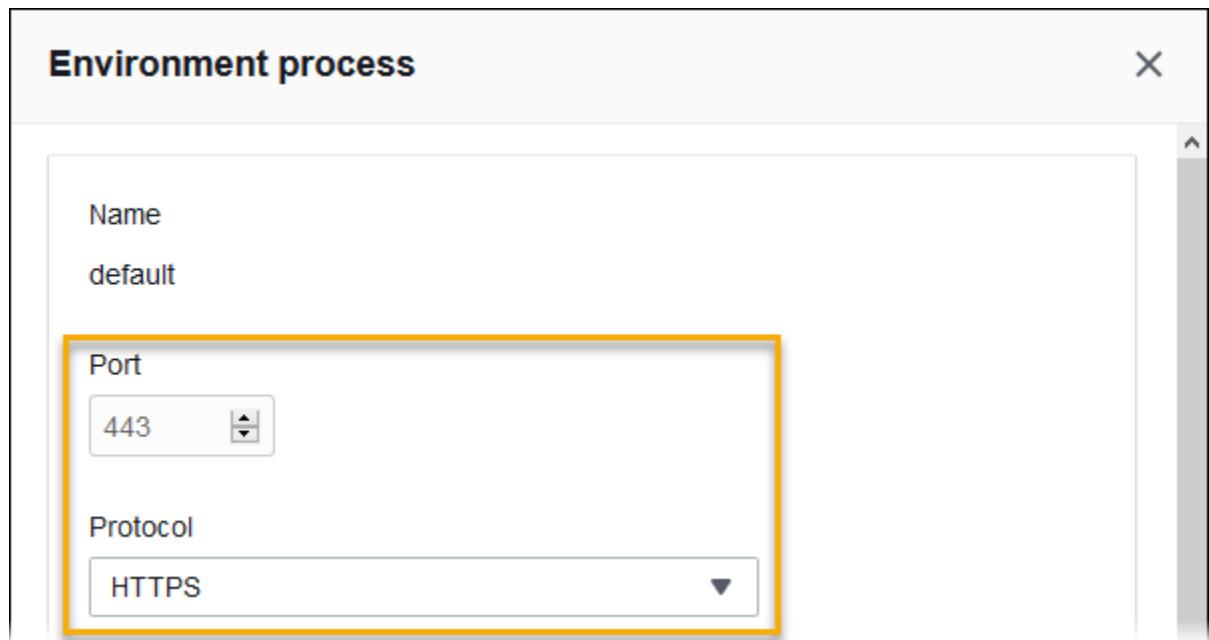
Dans Elastic Beanstalk, configurez chaque environnement de microservice pour utiliser l'équilibrer de charge Application Load Balancer partagé et définissez le port de l'écouteur par défaut sur 443. Dans le cas de l'environnement présenté ici, indiquez que le processus par défaut écoute le port 443 sur HTTPS, et ajoutez un processus et une règle d'écouteur pour le trafic admin sur un chemin différent.

Pour configurer l'équilibrer de charge partagé de cet exemple

1. Dans la section Shared Application Load Balancer (Équilibrer de charge Application Load Balancer partagé), sélectionnez votre équilibrer de charge, puis, pour Default listener port (Port d'écoute par défaut), sélectionnez **443**. Le port de l'écouteur doit déjà être sélectionné s'il s'agit du seul écouteur que possède l'équilibrer de charge.



2. Configurez le processus par défaut sur HTTPS. Sélectionnez le processus par défaut, puis, pour Actions, choisissez Edit (Modifier). Pour Port, entrez **443**. Pour Protocole, sélectionnez **HTTPS**.



3. Ajoutez un processus d'administration. Pour Nom, entrez **admin**. Pour Port, entrez **443**. Pour Protocole, sélectionnez **HTTPS**. Sous Vérification de l'état, pour Chemin, tapez **/admin**.

### Environment process

Name: admin

Port: 443

Protocol: HTTPS

### Health check

HTTP code: 200

Path: /admin

4. Ajoutez une règle pour le trafic d'administration. Pour Nom, entrez **admin**. Pour Port d'écouteur, entrez **443**. Pour les conditions Correspondance, ajoutez un PathPattern avec la valeur **/admin/\***. Pour Processus, sélectionnez **admin**.

**Listener rule**

**Name**  
admin

**Listener port**  
443 ▾

**Priority**  
Evaluated in ascending numerical order. Must be unique across all rules.  
1 ▾

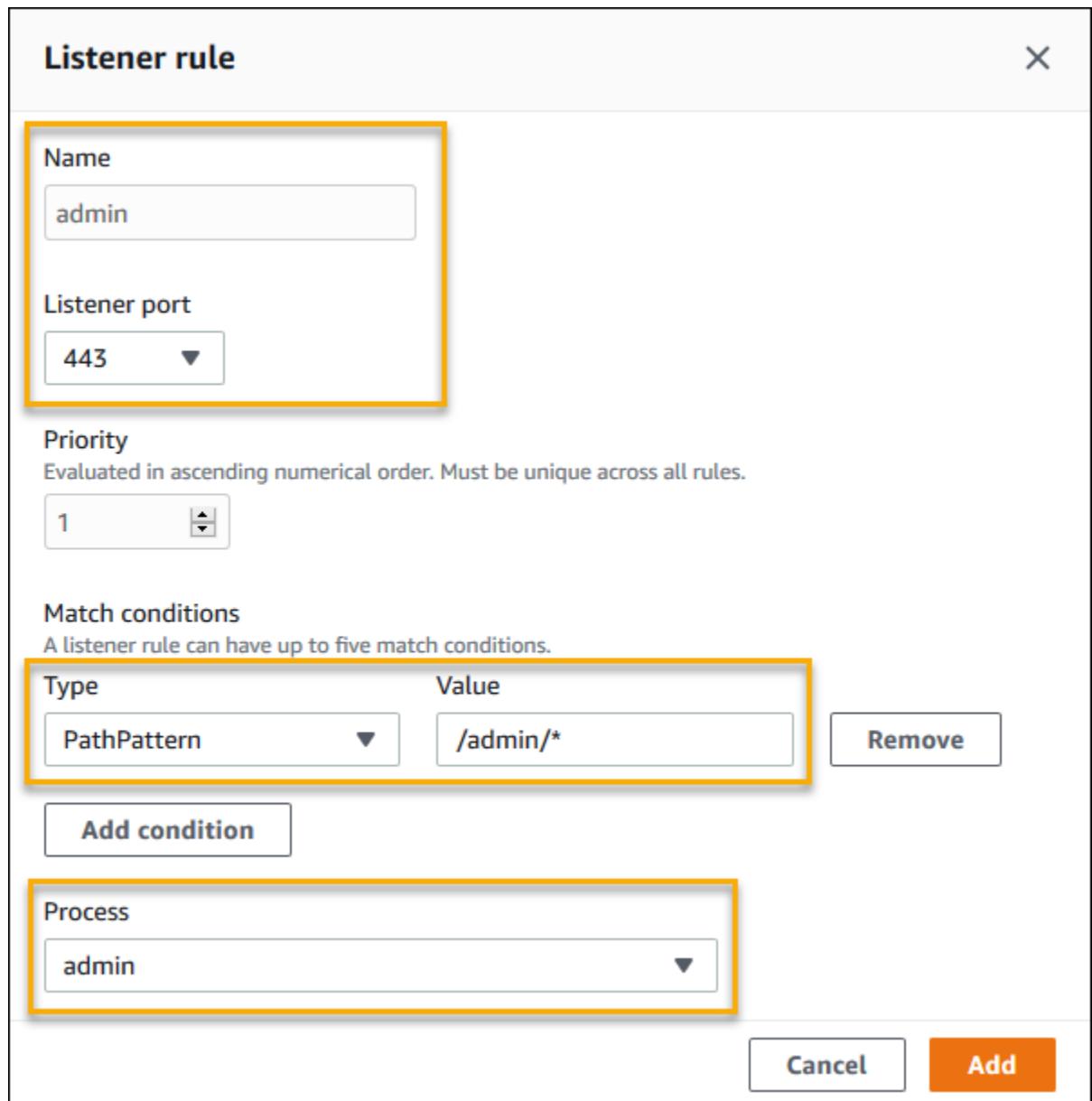
**Match conditions**  
A listener rule can have up to five match conditions.

Type	Value	Remove
PathPattern	/admin/*	Remove

Add condition

**Process**  
admin

Cancel Add



## Configuration d'un équilibrer de charge Application Load Balancer partagé à l'aide de l'interface de ligne de commande EB

L'interface de ligne de commande EB vous invite à choisir un type d'équilibrer de charge lorsque vous exécutez la commande `eb create` (p. 1078). Si vous choisissez application (valeur par défaut) et si votre compte dispose d'au moins un équilibrer de charge Application Load Balancer partageable, l'interface de ligne de commande EB vous demande également si vous souhaitez utiliser un équilibrer de charge Application Load Balancer partagé. Si vous répondez `y`, vous êtes également invité à sélectionner l'équilibrer de charge et le port par défaut.

```
$ eb create
Enter Environment Name
(default is my-app): test-env
```

```
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 2):

Your account has one or more sharable load balancers. Would you like your new environment
to use a shared load balancer?(y/N) y

Select a shared load balancer
1)MySharedALB1 - arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/
MySharedALB1/6d69caa75b15d46e
2)MySharedALB2 - arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/
MySharedALB2/e574ea4c37ad2ec8
(default is 1): 2

Select a listener port for your shared load balancer
1) 80
2) 100
3) 443
(default is 1): 3
```

Vous pouvez également spécifier un équilibrer de charge partagé à l'aide des options de commande.

```
$ eb create test-env --elb-type application --shared-lb MySharedALB2 --shared-lb-port 443
```

## Espaces de noms d'équilibreurs de charge Application Load Balancer partagés

Vous trouverez les paramètres liés aux équilibreurs Application Load Balancers partagés dans les espaces de noms suivants :

- [aws:elasticbeanstalk:environment \(p. 700\)](#) – Choisissez le type d'équilibreur de charge pour l'environnement et indiquez à Elastic Beanstalk que vous allez utiliser un équilibreur de charge partagé.

Vous ne pouvez pas définir ces deux options dans les fichiers de configuration ([.Ebextensions \(p. 737\)](#)).

- [aws:elbv2:loadbalancer \(p. 723\)](#) – Configurez l'ARN et les groupes de sécurité partagés de l'équilibreur de charge Application Load Balancer.
- [aws:elbv2:listener \(p. 720\)](#) – Associez les écouteurs de l'équilibreur de charge Application Load Balancer partagé avec l'environnement en listant les règles d'écoute.
- [aws:elbv2:listenerrule \(p. 721\)](#) – Configurez les règles d'écoute qui acheminent le trafic vers différents processus, selon le chemin de la demande. Les règles sont spécifiques aux équilibreurs de charge Application Load Balancer, qu'ils soient dédiés ou partagés.
- [aws:elasticbeanstalk:environment:process \(p. 701\)](#) – Configurez les vérifications de l'état et spécifiez le port et le protocole des processus qui s'exécutent sur les instances de votre environnement.

Example .ebextensions/application-load-balancer-shared.config

Pour commencer avec un équilibreur de charge Application Load Balancer partagé, utilisez la console Elastic Beanstalk, l'interface de ligne de commande EB ou l'API pour définir le type d'équilibreur de charge sur `application` et choisissez d'utiliser un équilibreur de charge partagé. Utilisez un [fichier de configuration \(p. 737\)](#) pour configurer l'équilibreur de charge partagé.

```
option_settings:  
  aws:elbv2:loadbalancer:  
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/  
app/MySharedALB2/e574ea4c37ad2ec8
```

#### Note

Vous ne pouvez configurer cette option que lors de la création de l'environnement.

#### Example .ebextensions/alb-shared-secure-listener.config

Le fichier de configuration suivant sélectionne un écouteur sécurisé par défaut sur le port 443 pour l'équilibrer de charge partagé et définit le processus par défaut pour écouter le port 443.

```
option_settings:  
  aws:elbv2:loadbalancer:  
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/  
app/MySharedALB2/e574ea4c37ad2ec8  
  aws:elbv2:listener:443:  
    rules: default  
  aws:elasticbeanstalk:environment:process:default:  
    Port: '443'  
    Protocol: HTTPS
```

#### Example .ebextensions/alb-shared-admin-rule.config

Le fichier de configuration suivant s'appuie sur l'exemple précédent et ajoute une règle qui achemine le trafic avec un chemin de requête de /admin vers un processus nommé admin qui écoute sur le port 4443.

```
option_settings:  
  aws:elbv2:loadbalancer:  
    SharedLoadBalancer: arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/  
app/MySharedALB2/e574ea4c37ad2ec8  
  aws:elbv2:listener:443:  
    rules: default,admin  
  aws:elasticbeanstalk:environment:process:default:  
    Port: '443'  
    Protocol: HTTPS  
  aws:elasticbeanstalk:environment:process:admin:  
    HealthCheckPath: /admin  
    Port: '4443'  
    Protocol: HTTPS  
  aws:elbv2:listenerrule:admin:  
    PathPatterns: /admin/*  
    Priority: 1  
    Process: admin
```

## Configuration d'un équilibrer de charge réseau

Lorsque vous activez l'[équilibrage de charge \(p. 520\)](#), votre environnement AWS Elastic Beanstalk est équipé d'un équilibrer de charge Elastic Load Balancing qui permet de répartir le trafic entre les instances de votre environnement. Elastic Load Balancing prend en charge plusieurs types d'équilibrer de charge. Pour en savoir plus, consultez le [Guide de l'utilisateur Elastic Load Balancing](#). Elastic Beanstalk peut créer un équilibrer de charge pour vous, ou vous permettre de spécifier un équilibrer de charge partagé que vous avez créé.

Cette rubrique décrit la configuration d'un équilibrer de charge [Network Load Balancer](#) créé par Elastic Beanstalk et dédié à votre environnement. Pour de plus amples informations sur la configuration de tous les types d'équilibrer de charge pris en charge par Elastic Beanstalk, veuillez consulter [Équilibrer de charge pour votre environnement Elastic Beanstalk \(p. 562\)](#).

#### Note

Vous pouvez choisir le type d'équilibrer de charge que votre environnement utilise uniquement lors de la création de l'environnement. Vous pouvez modifier les paramètres pour gérer le comportement de l'équilibrer de charge de votre environnement d'exécution, mais pas en changer le type.

## Introduction

Avec un équilibrer de charge Network Load Balancer, l'écouteur par défaut accepte les demandes TCP sur le port 80 et les distribue aux instances de votre environnement. Vous pouvez configurer le comportement de la vérification de l'état, configurer le port d'écoute ou ajouter un écouteur sur un autre port.

#### Note

Contrairement à un équilibrer de charge Classic Load Balancer ou Application Load Balancer, un équilibrer de charge Network Load Balancer ne peut pas avoir d'écouteurs HTTP or HTTPS de couche de transport (couche 7). Il ne prend en charge que les écouteurs TCP de la couche de transport (couche 4). Le trafic HTTP et HTTPS peut être acheminé vers votre environnement via TCP. Pour établir des connexions HTTPS sécurisées entre les clients web et votre environnement, installez un [certificat auto-signé \(p. 794\)](#) sur les instances de l'environnement, et configurez les instances pour écouter sur le port approprié (généralement, le port 443) et arrêtez les connexions HTTPS. La configuration varie selon la plateforme. Pour obtenir des instructions, veuillez consulter [Configuration de votre application pour suspendre des connexions HTTPS sur l'instance \(p. 799\)](#). Ensuite, configurez votre équilibrer de charge Network Load Balancer pour ajouter un écouteur qui est mappé à un processus à l'écoute sur le port approprié.

Un équilibrer de charge Network Load Balancer prend en charge les vérifications de l'état actives. Ces vérifications sont basées sur les messages transmis au chemin d'accès racine (/). Un équilibrer de charge Network Load Balancer prend également en charge les vérifications de l'état passives. Il détecte automatiquement les instances backend défectueuses et achemine uniquement le trafic vers des instances saines.

## Configuration d'un équilibrer de charge Network Load Balancer à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer les écouteurs et les processus d'un équilibrer de charge Network Load Balancer lors de la création de l'environnement ou plus tard alors que votre environnement est en cours d'exécution.

Pour configurer un équilibrer de charge Network Load Balancer dans la console Elastic Beanstalk lors de la création de l'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements).
3. Choisissez [Create a new environment \(Créer un nouvel environnement\) \(p. 442\)](#) pour commencer à créer votre environnement.
4. Sur la page principale de l'assistant, avant de choisir Crée un environnement, choisissez Configurer plus d'options.
5. Choisissez le prérglage de configuration Haute disponibilité.

Sinon, dans la catégorie de configuration Capacité, configuez un type d'environnement avec un Équilibrage de charge. Pour plus d'informations, consultez [Capacity \(p. 450\)](#).

6. Dans la catégorie de configuration Load balancer (Équilibrer de charge), choisissez Edit (Modifier).

7. Sélectionnez l'option Network Load Balancer (Équilibrer de charge du réseau) si elle n'est pas déjà sélectionnée.

Elastic Beanstalk > Applications > getting-started-app

## Modify load balancer

### Application Load Balancer

Application layer load balancer—routing HTTP and HTTPS traffic based on protocol, port, and route to environment processes.

### Classic Load Balancer

*Previous generation* — HTTP, HTTPS, and TCP

### Network Load Balancer

Ultra-high performance and static IP addresses for your application.

8. Effectuez toutes les modifications de configuration de l'équilibrer de charge Network Load Balancer exigées par votre environnement.
9. Choisissez Enregistrer, puis effectuez toutes les autres modifications de configuration exigées par votre environnement.
10. Choisissez Create environment.

Pour configurer un équilibrer de charge Network Load Balancer d'un environnement en cours d'exécution dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Load balancer (Équilibrer de charge), choisissez Edit (Modifier).

#### Note

Si la catégorie de configuration Load balancer (Équilibrer de charge) ne dispose pas du bouton Edit (Modifier), cela signifie que votre environnement ne dispose pas d'un équilibrer de charge. Pour apprendre à en configurer un, consultez [Changement de type d'environnement \(p. 520\)](#).

5. Effectuez les modifications de configuration de l'équilibrer de charge Network Load Balancer exigées par votre environnement.
6. Choisissez Apply.

Paramètres de l'équilibrer de charge Network Load Balancer

- [Écouteurs \(p. 610\)](#)
- [Processus \(p. 611\)](#)

## Écouteurs

Utilisez cette liste pour spécifier plusieurs écouteurs pour votre équilibrer de charge. Chaque écouteur achemine le trafic client entrant sur un port spécifié vers un processus spécifié sur vos instances. Initialement, la liste indique l'écouteur par défaut, qui achemine le trafic HTTP entrant sur le port 80 pour un processus nommé default, qui écoute le port HTTP 80.

**Network Load Balancer**

You can specify listeners for your load balancer. Each listener routes incoming client traffic on a specified port using TCP to an environment variable (the port that the process listens on). By default, we've configured your load balancer with a listener on port 80 that routes traffic to a process named default on port 80.

**Actions ▾**

	<b>Listener port</b>	<b>Process port</b>	<b>Protocol</b>	<b>Enabled</b>
<input type="checkbox"/>	80	80	TCP	<input checked="" type="checkbox"/>

Pour configurer un écouteur existant

1. Cochez la case en regard de son entrée de table, puis choisissez Actions, Modifier.
2. Utilisez la boîte de dialogue Network Load Balancer listener (Écouteur Network Load Balancer) pour modifier les paramètres, puis choisissez Save (Enregistrer).

Pour ajouter un écouteur

1. Choisissez Ajouter un écouteur.
2. Dans la boîte de dialogue Network Load Balancer listener (Écouteur Network Load Balancer), configurez les paramètres requis, puis sélectionnez Ajouter.

Utilisez la boîte de dialogue Network Load Balancer listener (Écouteur Network Load Balancer) pour configurer le port sur lequel l'écouteur écoute le trafic, et choisir le processus vers lequel vous souhaitez acheminer le trafic (spécifié par le port que le processus écoute).

### Network Load Balancer listener

**Listener port**  
80

**Protocol**  
The transport protocol that the load balancer uses for routing incoming traffic from clients.  
**TCP**

**Process port**  
The port to which this listener routes traffic. It determines the environment process that receives traffic from the listener.  
**80**

**Cancel** **Save**

## Processus

Utilisez cette liste pour spécifier les processus pour votre équilibrer de charge. Un processus est une cible pour le routage du trafic par les écouteurs. Chaque écouteur achemine le trafic client entrant sur un port spécifié vers un processus spécifié sur vos instances. Initialement, la liste indique le processus par défaut, qui écoute le trafic entrant sur le port 80.

## Processes

For each environment process, you can specify the port that the load balancer uses to route requests to the process. You can also specify the interval and healthy threshold for the process health checks.

	Process name	Process port	Interval	Healthy threshold	Unhealthy threshold
<input type="checkbox"/>	default	80	10	5	5

**Actions ▾**

Vous pouvez modifier les paramètres d'un processus existant ou ajouter un nouveau processus. Pour commencer la modification d'un processus de la liste ou l'ajout d'un processus, utilisez la procédure définie pour la [liste d'écouteurs \(p. 577\)](#). La boîte de dialogue Processus d'environnement s'ouvre.

Paramètres de la boîte de dialogue de processus d'environnement de l'équilibrer de charge Network Load Balancer

- [Définition \(p. 612\)](#)
- [Vérification de l'état \(p. 613\)](#)

### Définition

Utilisez ces paramètres pour définir le processus : son nom et le port de processus sur lequel il écoute les requêtes.

### Environment process

**Name**  
default

**Process port**  
80

## Vérification de l'état

Utilisez les paramètres suivants pour configurer les processus des vérifications de l'état :

- **Interval (Intervalle)** – Durée, en secondes, entre les vérifications de l'état d'une instance.
- **Healthy threshold (Seuil de bonne santé)** – Nombre de vérifications de l'état qui doivent réussir avant qu'Elastic Load Balancing modifie l'état de santé d'une instance. (Pour l'équilibreur de charge Network Load Balancer, l'option **Unhealthy threshold (Seuil de défectuosité)** est un paramètre en lecture seule qui est toujours égal à la valeur du seuil de bonne santé.)
- **Deregistration delay (Retard d'annulation d'enregistrement)** – Délai, en secondes, d'attente de la fin des requêtes actives avant l'annulation de l'enregistrement d'une instance.

### Health check

**Interval**  
Amount of time between health checks of an individual instance.

10 ▾  
seconds

**Healthy threshold**  
The number of consecutive successful health checks required to designate the instance as healthy.

5 ▲▼ requests

**Unhealthy threshold**  
The number of consecutive health check failures required to designate the instance as unhealthy.

5 ▲▼ requests

**Deregistration delay**  
Amount of time to wait for active requests to complete before deregistering.

20 ▲▼ seconds

**Cancel** **Save**

#### Note

La vérification de l'état Elastic Load Balancing n'a pas d'incidence sur le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Les instances dont la vérification de l'état Elastic Load Balancing échoue ne sont pas automatiquement remplacées par Amazon EC2 Auto Scaling, sauf si vous configurez manuellement Amazon EC2 Auto Scaling pour le faire. Pour de plus amples informations, veuillez consulter [Paramètre de vérification de l'état Auto Scaling \(p. 561\)](#).

Pour de plus amples informations sur les vérifications de l'état et leur influence sur l'état global de votre environnement, veuillez consulter [Création de rapports d'intégrité de base \(p. 834\)](#).

## Exemple : équilibrer de charge Network Load Balancer pour un environnement avec un chiffrement de bout en bout

Dans cet exemple, votre application requiert le chiffrement du trafic de bout en bout. Pour configurer l'équilibrer de charge Network Load Balancer de votre environnement afin de répondre à ces exigences, configurez le processus par défaut pour écouter sur le port 443, ajoutez un écouteur sur le port 443 qui achemine le trafic vers le processus par défaut et désactivez l'écouteur par défaut.

Pour configurer l'équilibrer de charge de cet exemple

1. Configurez le processus par défaut. Sélectionnez le processus par défaut, puis, pour Actions, choisissez Edit (Modifier). Pour Process port (Port de processus), tapez 443.



2. Ajoutez un écouteur de port 443. Ajoutez un écouteur. Pour Port d'écoute, tapez 443. Pour Process port (Port de processus), assurez-vous que 443 est sélectionné.

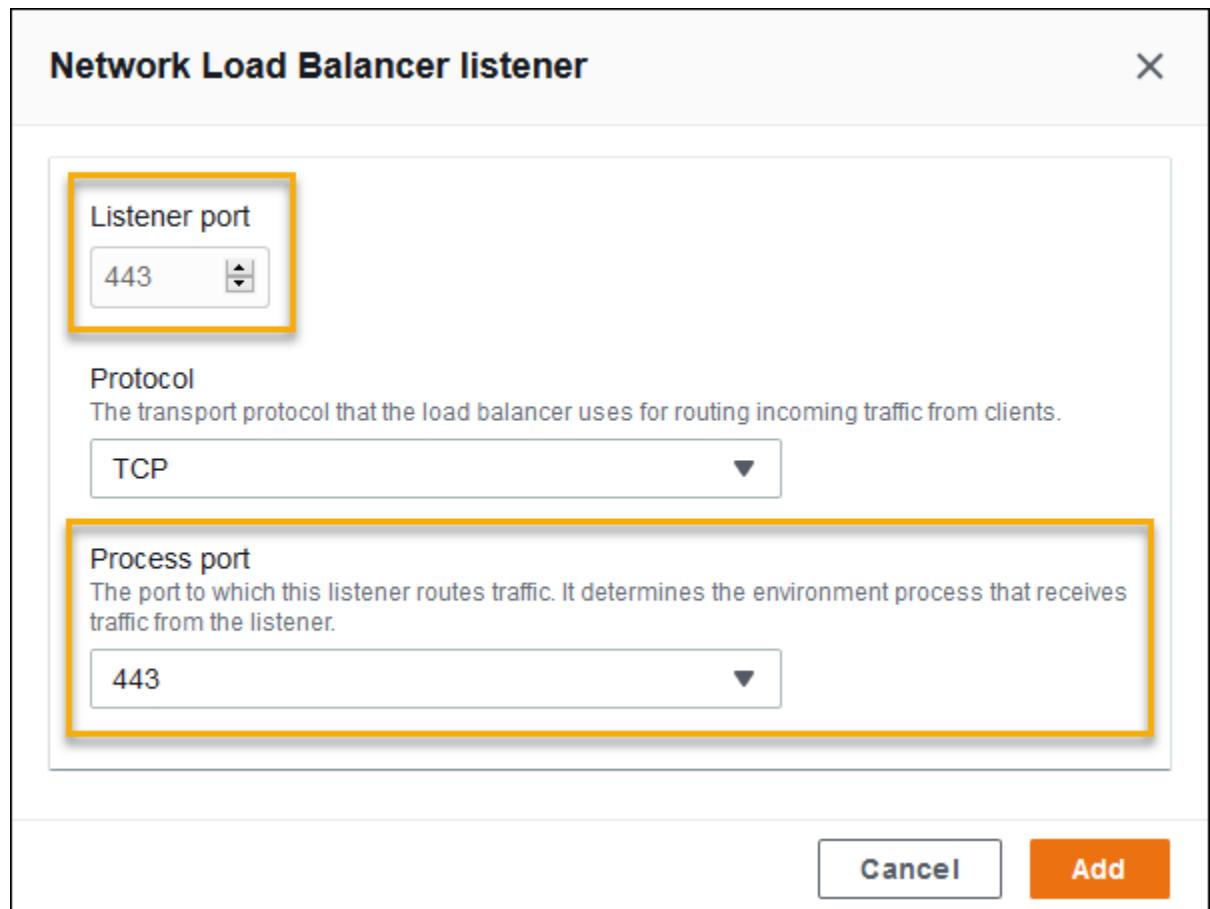
### Network Load Balancer listener

Listener port  
443

Protocol  
TCP

Process port  
443

Cancel Add



Vous voyez désormais votre écouteur supplémentaires dans la liste.

<input type="checkbox"/>	Listener port	Process port	Protocol	En
<input type="checkbox"/>	80	443	TCP	<input checked="" type="checkbox"/>
<input type="checkbox"/>	443	443	TCP	<input checked="" type="checkbox"/>

3. Désactivez l'écouteur du port 80 par défaut. Pour l'écouteur par défaut, désactivez l'option Activé.

<input type="checkbox"/>	Listener port	Process port	Protocol	En
<input type="checkbox"/>	80	443	TCP	<input checked="" type="checkbox"/>
<input type="checkbox"/>	443	443	TCP	<input type="checkbox"/>

## Configuration d'un équilibrer de charge Network Load Balancer à l'aide de l'interface de ligne de commande EB

L'interface de ligne de commande EB vous invite à choisir un type d'équilibrer de charge lorsque vous exécutez la commande [eb create \(p. 1078\)](#).

```
$ eb create
Enter Environment Name
(default is my-app): test-env
Enter DNS CNAME prefix
(default is my-app): test-env-DLW24ED23SF

Select a load balancer type
1) classic
2) application
3) network
(default is 1): 3
```

Vous pouvez également spécifier un type d'équilibrer de charge à l'aide de l'option `--elb-type`.

```
$ eb create test-env --elb-type network
```

## Espaces de noms de l'équilibrer de charge Network Load Balancer

Vous trouverez les paramètres liés aux équilibrers Network Load Balancer dans les espaces de noms suivants :

- [aws:elasticbeanstalk:environment \(p. 700\)](#) – Choisissez le type d'équilibrer de charge pour votre environnement. La valeur d'un équilibrer de charge Network Load Balancer est `network`.
- [aws:elbv2:listener \(p. 720\)](#) – Configurez les écouteurs sur l'équilibrer de charge Network Load Balancer. Ces paramètres sont mappés aux paramètres de `aws:elb:listener` pour les équilibrers de charge Classic Load Balancers.
- [aws:elasticbeanstalk:environment:process \(p. 701\)](#) – Configurez les vérifications de l'état et spécifiez le port et le protocole des processus qui s'exécutent sur les instances de votre environnement. Les paramètres de port et de protocole sont mappés sur ceux de l'instance dans `aws:elb:listener` pour un écouteur sur un équilibrer de charge Classic Load Balancer. Les paramètres de vérification de l'état sont mappés sur les paramètres des espaces de noms `aws:elb:healthcheck` et `aws:elasticbeanstalk:application`.

Example `.ebextensions/network-load-balancer.config`

Pour vous familiariser avec un équilibrer de charge Network Load Balancer utilisez un [fichier de configuration \(p. 737\)](#) pour définir le type d'équilibrer de charge sur `network`.

```
option_settings:
  aws:elasticbeanstalk:environment:
    LoadBalancerType: network
```

### Note

Le type d'équilibrer de charge peut uniquement être défini lors de la création de l'environnement.

### Example .ebextensions/nlb-default-process.config

Le fichier de configuration suivant modifie les paramètres de vérification de l'état sur le processus par défaut.

```
option_settings:  
  aws:elasticbeanstalk:environment:process:default:  
    DeregistrationDelay: '20'  
    HealthCheckInterval: '10'  
    HealthyThresholdCount: '5'  
    UnhealthyThresholdCount: '5'  
    Port: '80'  
    Protocol: TCP
```

### Example .ebextensions/nlb-secure-listener.config

Le fichier de configuration suivant ajoute un écouteur pour sécuriser le trafic sur le port 443 et un processus cible correspondant qui écoute le port 443.

```
option_settings:  
  aws:elbv2:listener:443:  
    DefaultProcess: https  
    ListenerEnabled: 'true'  
  aws:elasticbeanstalk:environment:process:https:  
    Port: '443'
```

L'option `DefaultProcess` est nommée ainsi en raison des équilibriseurs de charge Application Load Balancer, qui peuvent avoir des écouteurs autres que ceux par défaut sur le même port pour le trafic vers des chemins spécifiques (veuillez consulter [Application Load Balancer \(p. 574\)](#) pour plus de détails). Pour un équilibrateur de charge Network Load Balancer, l'option spécifie le seul processus cible de cet écouteur.

Dans cet exemple, nous avons nommé le processus `https`, car il écoute le trafic sécurisé (HTTPS). L'écouteur envoie le trafic vers le processus sur le port désigné à l'aide du protocole TCP, car un équilibrateur de charge Network Load Balancer fonctionne uniquement avec TCP. C'est tout à fait normal, car le trafic réseau pour HTTP et HTTPS est mis en œuvre par-dessus TCP.

## Configuration des journaux d'accès

Vous pouvez utiliser des [fichiers de configuration \(p. 737\)](#) afin de configurer l'équilibrEUR de charge de votre environnement pour qu'il charge les journaux d'accès dans un compartiment Amazon S3. Pour obtenir des instructions, reportez-vous aux exemples de fichiers de configuration suivants sur GitHub :

- [`loadbalancer-accesslogs-existingbucket.config`](#) – Configurez l'équilibrEUR de charge pour qu'il charge les journaux d'accès dans un compartiment Amazon S3 existant.
- [`loadbalancer-accesslogs-newbucket.config`](#) – Configurez l'équilibrEUR de charge pour qu'il charge les journaux d'accès dans un nouveau compartiment.

## Ajout d'une base de données à votre environnement Elastic Beanstalk

Elastic Beanstalk offre une intégration à [Amazon Relational Database Service \(Amazon RDS\)](#). Vous pouvez utiliser Elastic Beanstalk pour ajouter une base de données MySQL, PostgreSQL, Oracle ou SQL Server à un environnement existant ou à un nouvel environnement, lorsque vous le créez. Lorsque

vous ajoutez une instance de base de données, Elastic Beanstalk fournit les informations de connexion à votre application. Pour ce faire, définissez les propriétés d'environnement pour le nom d'hôte de la base de données, le port, le nom d'utilisateur, le mot de passe et le nom de la base de données.

Si vous n'avez encore jamais utilisé d'instance de base de données avec votre application, nous vous recommandons d'abord d'utiliser le processus décrit dans cette rubrique pour ajouter une base de données à un environnement de test à l'aide du service Elastic Beanstalk. Cette opération vous permet de vérifier que votre application peut lire les propriétés de l'environnement, créer une chaîne de connexion et se connecter à une instance de base de données sans le travail de configuration supplémentaire requis pour une base de données externe à Elastic Beanstalk.

Après avoir vérifié que votre application fonctionne correctement avec la base de données, vous pouvez envisager de passer à un environnement de production. À ce stade, vous avez la possibilité de découpler la base de données de votre environnement Elastic Beanstalk pour passer à une configuration offrant une plus grande flexibilité. La base de données découpée peut demeurer opérationnelle en tant qu'instance de base de données Amazon RDS externe. L'état de l'environnement n'est pas affecté par le découplage de la base de données. Si vous devez résilier l'environnement, vous pouvez le faire et choisir l'option permettant de maintenir la base de données disponible et opérationnelle en dehors d'Elastic Beanstalk.

L'utilisation d'une base de données externe présente plusieurs avantages. Vous pouvez vous connecter à la base de données externe à partir de plusieurs environnements, utiliser des types de base de données non pris en charge par les bases de données intégrées et effectuer des déploiements bleu/vert. Au lieu d'utiliser une base de données découpée créée par Elastic Beanstalk, vous pouvez également créer une instance de base de données en dehors de votre environnement Elastic Beanstalk. Quelle que soit l'option privilégiée, l'instance de base de données externe à votre environnement Elastic Beanstalk générée nécessite une configuration supplémentaire pour le groupe de sécurité et la chaîne de connexion. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#).

#### Sections

- [Cycle de vie de base de données \(p. 618\)](#)
- [Ajout d'une instance de base de données Amazon RDS à votre environnement à l'aide de la console \(p. 619\)](#)
- [Connexion à la base de données \(p. 620\)](#)
- [Configuration d'une instance de base de données RDS intégrée à l'aide de la console \(p. 621\)](#)
- [Configuration d'une instance de base de données RDS intégrée à l'aide des fichiers de configuration \(p. 621\)](#)
- [Découplage d'une instance de base de données RDS à l'aide de la console \(p. 622\)](#)
- [Découplage d'une instance de base de données RDS à l'aide de fichiers de configuration \(p. 625\)](#)

## Cycle de vie de base de données

Vous pouvez déterminer que faire de la base de données une fois que vous l'avez découpée de votre environnement Elastic Beanstalk. Les options parmi lesquelles vous pouvez choisir sont collectivement appelées stratégies de suppression. Les stratégies de suppression suivantes s'appliquent à une base de données après que vous l'ayez [découplée d'un environnement Elastic Beanstalk \(p. 622\)](#) ou que vous ayez résilié l'environnement Elastic Beanstalk.

- Snapshot (Instantané) : avant qu'Elastic Beanstalk résilie la base de données, il enregistre un instantané de celle-ci. Vous pouvez restaurer une base de données à partir d'un instantané lorsque vous ajoutez une instance de base de données à un environnement Elastic Beanstalk ou lorsque vous créez une base de données autonome. Pour plus d'informations sur la création d'une nouvelle instance de base de données autonome à partir d'un instantané, consultez [Restauration à partir d'un instantané de base de données](#) dans le Guide de l'utilisateur Amazon RDS. Le stockage des instantanés de base de données peut entraîner des frais. Pour de plus amples informations, veuillez consulter la section Stockage de sauvegarde dans [Tarification d'Amazon RDS](#).

- Delete (Supprimer) : Elastic Beanstalk résilie la base de données. Une fois la base de données résiliée, l'instance de base de données n'est plus disponible pour aucune opération.
- Retain (Conserver) : l'instance de base de données n'est pas résiliée. Elle reste disponible et opérationnelle, bien qu'elle ait été découpée d'Elastic Beanstalk. Vous pouvez ensuite configurer un ou plusieurs environnements pour connexion à la base de données en tant qu'instance de base de données Amazon RDS externe. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#).

## Ajout d'une instance de base de données Amazon RDS à votre environnement à l'aide de la console

Vous pouvez ajouter une instance de base de données à votre environnement à l'aide de la console Elastic Beanstalk.

Pour ajouter une instance DB à votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Choisissez un moteur de base de données, puis saisissez un nom d'utilisateur et un mot de passe.
6. Choisissez Apply.

Vous pouvez configurer les options suivantes :

- Instantané – Choisissez un instantané existant de la base de données. Elastic Beanstalk restaure l'instantané et l'ajoute à votre environnement. La valeur par défaut est Aucune. Lorsque la valeur par défaut est None (Aucun), vous pouvez configurer la nouvelle base de données en utilisant les autres paramètres sur cette page.
- Moteur – Choisissez un moteur de base de données.
- Version du moteur – Choisissez une version spécifique du moteur de base de données.
- Classe d'instance – Choisissez la classe d'instance de base de données. Pour de plus amples informations sur les classes d'instance de base de données, consultez <http://aws.amazon.com/rds/>.
- Stockage – Choisissez la quantité de stockage à allouer à votre base de données. Vous pouvez augmenter le stockage alloué ultérieurement, mais vous ne pouvez pas le réduire. Pour de plus amples informations sur l'allocation de stockage, veuillez consulter [Fonctionnalités](#).
- Username (Nom d'utilisateur) : saisissez un nom d'utilisateur de votre choix en utilisant une combinaison de chiffres et de lettres uniquement.
- Mot de passe – Entrez un mot de passe de votre choix contenant 8–16 caractères ASCII imprimables (sauf /, \ et @).
- Disponibilité – Choisissez Élevée (Multi-AZ) pour exécuter une sauvegarde à chaud dans une deuxième zone de disponibilité pour une haute disponibilité.

- Database deletion policy (Stratégie de suppression de base de données) : la stratégie de suppression détermine ce qu'il advient de la base de données une fois qu'elle est [découplée \(p. 622\)](#) de votre environnement. Elle peut être définie sur les valeurs suivantes : Create Snapshot, Retain ou Delete. Ces valeurs sont décrites dans [Cycle de vie de base de données \(p. 618\)](#) de cette rubrique.

#### Note

Elastic Beanstalk crée un utilisateur principal pour la base de données en utilisant le nom d'utilisateur et le mot de passe que vous fournissez. Pour de plus amples informations sur l'utilisateur principal et ses priviléges, veuillez consulter [Priviléges du compte utilisateur principal](#).

L'ajout d'une instance de base de données prend environ 10 minutes. Lorsque la mise à jour est terminée, la nouvelle base de données est couplée à votre environnement. Le nom d'hôte de l'instance de base de données et les autres informations de connexion sont disponibles dans votre application via les propriétés d'environnement suivantes.

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

## Connexion à la base de données

Utilisez les informations de connectivité pour vous connecter à votre base de données à partir de votre application, par le biais des variables d'environnement. Pour plus d'informations sur l'utilisation d'Amazon RDS avec vos applications, consultez les rubriques suivantes.

- Java SE – [Connexion à une base de données \(plateformes Java SE\) \(p. 138\)](#)
- Java avec Tomcat – [Connexion à une base de données \(plateformes Tomcat\) \(p. 138\)](#)
- Node.js – [Connexion à une base de données \(p. 289\)](#)
- .NET – [Connexion à une base de données \(p. 223\)](#)
- PHP – [Connexion à une base de données avec un PDO ou MySQLi \(p. 355\)](#)
- Python – [Connexion à une base de données \(p. 383\)](#)

- Ruby – [Connexion à une base de données \(p. 402\)](#)

## Configuration d'une instance de base de données RDS intégrée à l'aide de la console

Vous pouvez visualiser et modifier les paramètres de configuration de votre instance de base de données dans la section Database (Base de données) de la page Configuration de l'environnement, dans la [console Elastic Beanstalk \(p. 429\)](#).

Pour configurer l'instance de base de données de votre environnement dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).

Vous pouvez modifier les paramètres Instance class (Classe d'instance), Storage (Stockage), Password (Mot de passe), Availability (Disponibilité) et Database deletion policy (Stratégie de suppression de base de données) après la création de la base de données. Si vous modifiez la classe d'instance, Elastic Beanstalk reprovisionne l'instance de base de données.

Si vous n'avez plus besoin d'Elastic Beanstalk pour associer la base de données à l'environnement, vous pouvez décider de la découpler en sélectionnant Decouple database (Découpler la base de données). Il importe de comprendre les options et les considérations liées à cette opération. Pour de plus amples informations, veuillez consulter [the section called “Découplage d'une instance de base de données RDS à l'aide de la console” \(p. 622\)](#).

### Warning

Ne modifiez pas les paramètres sur l'instance de base de données couplée en dehors de la fonctionnalité fournie par Elastic Beanstalk (par exemple, dans la console Amazon RDS).

Si vous le faites, votre configuration de base de données Amazon RDS risque de ne pas être synchronisée avec la définition de votre environnement. Lorsque vous mettez à jour ou redémarrez votre environnement, les paramètres spécifiés dans l'environnement remplacent ceux que vous avez définis en dehors d'Elastic Beanstalk.

Si vous devez modifier les paramètres que Elastic Beanstalk ne prend pas en charge directement, utilisez les [fichiers de configuration \(p. 621\)](#) Elastic Beanstalk.

## Configuration d'une instance de base de données RDS intégrée à l'aide des fichiers de configuration

Vous pouvez configurer une instance de base de données de votre environnement à l'aide de [fichiers de configuration \(p. 737\)](#). Utilisez les options de l'espace de noms `aws:rds:dbinstance` (p. 725). L'exemple suivant remplace la taille de stockage de base de données allouée par 100 Go.

#### Example .ebextensions/db-instance-options.config

```
option_settings:  
  aws:rds:dbinstance:  
    DBAllocatedStorage: 100
```

Si vous souhaitez configurer les propriétés d'instance de base de données qui ne sont pas prises en charge par Elastic Beanstalk, vous pouvez utiliser un fichier de configuration et spécifier vos paramètres à l'aide de la clé `resources`. L'exemple suivant définit les valeurs sur les propriétés Amazon RDS `StorageType` et `Iops`.

#### Example .ebextensions/db-instance-properties.config

```
Resources:  
  AWSEBRDSDatabase:  
    Type: AWS::RDS::DBInstance  
    Properties:  
      StorageType:io1  
      Iops: 1000
```

## Découplage d'une instance de base de données RDS à l'aide de la console

Vous pouvez découpler votre base de données d'un environnement Elastic Beanstalk sans affecter l'état de l'environnement. Avant de découpler la base de données, tenez compte des exigences suivantes :

Que doit-il advenir de la base de données une fois qu'elle est découpée ?

Vous pouvez choisir de créer un instantané de la base de données, puis de la résilier, de conserver la base de données opérationnelle en tant que base de données autonome externe à Elastic Beanstalk, ou de supprimer définitivement la base de données. Le paramètre Database deletion policy (Stratégie de suppression de base de données) détermine ce résultat. Pour bénéficier d'une description détaillée des stratégies de suppression, consultez [Cycle de vie de base de données \(p. 618\)](#) dans cette rubrique.

Avez-vous besoin d'apporter des modifications aux paramètres de configuration de la base de données avant de la découpler ?

Si vous devez apporter des modifications de configuration à la base de données, vous devez les appliquer avant le découplage de la base de données. Cela inclut les modifications apportées au paramètre Database deletion policy (Stratégie de suppression de base de données). Toute modification en attente envoyée simultanément avec le paramètre Decouple database (Découpler la base de données) est ignorée, car seul le paramètre de découplage est appliqué.

Pour découpler une instance de base de données d'un environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Database (Base de données), choisissez Edit (Modifier).
5. Passez en revue toutes les valeurs de configuration dans la section Database settings (Paramètres de la base de données), en particulier la valeur Database deletion policy (Stratégie de suppression de base de données), qui détermine ce qu'il advient de la base de données une fois qu'elle est découpée.

The screenshot shows the 'Database settings' configuration page. It includes fields for Engine (mysql), Engine version (MySQL 5.7), Instance class (db.t2.micro), Storage (5 GB), Username (test), Password (\*\*\*\*\*), Availability (Low (one AZ)), and a Database deletion policy section. The 'Retain' option is selected, indicating the database will remain available and operational external to Elastic Beanstalk.

**Database settings**  
Choose an engine and instance type for your environment's database.

**Engine**  
mysql

**Engine version**  
--

**Instance class**  
db.t2.micro

**Storage**  
Choose a number between 5 GB and 1024 GB.  
5

**Username**  
test

**Password**  
\*\*\*\*\*

**Availability**  
Low (one AZ)

**Database deletion policy**  
This policy applies when you decouple a database or terminate the environment coupled to it.

**Create snapshot**  
Elastic Beanstalk saves a snapshot of the database and then deletes it. You can restore a database from a snapshot when you add a DB to an Elastic Beanstalk environment when you create a standalone database. You might incur charges for storing database snapshots.

**Retain**  
The decoupled database will remain available and operational external to Elastic Beanstalk.

**Delete**  
Elastic Beanstalk terminates the database. The database will no longer be available.

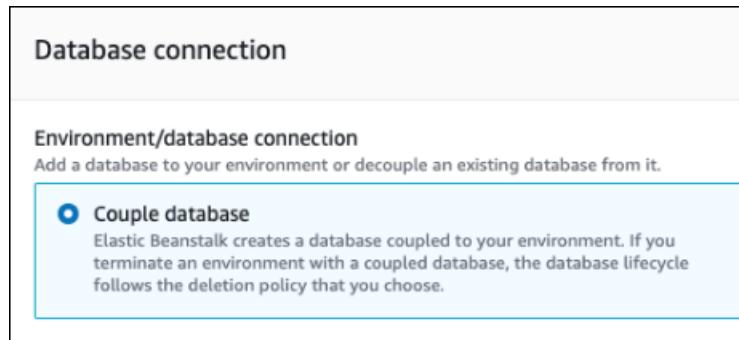
Si tous les autres paramètres de configuration sont corrects, passez à l'étape 6 pour découpler la base de données.

### Warning

Il est important d'appliquer le paramètre Database deletion policy (Stratégie de suppression de base de données) séparément du paramètre Decouple database (Découpler la base de données). Si vous sélectionnez Apply (Appliquer) dans le but d'enregistrer à la fois Decouple database (Découpler la base de données) et une nouvelle Database deletion policy (Stratégie de suppression de base de données), la nouvelle stratégie de suppression que vous avez choisie est ignorée. Elastic Beanstalk découple alors la base de données en suivant la stratégie de suppression définie précédemment. Si la stratégie de suppression antérieure est Delete ou Create Snapshot, vous risquez de perdre la base de données au lieu de suivre la stratégie en attente prévue.

Si l'un des paramètres de configuration nécessite des mises à jour, procédez comme suit :

1. Apportez les modifications requises dans le panneau Database settings (Paramètres de la base de données).
2. Choisissez Apply. Quelques minutes sont nécessaires à l'enregistrement des modifications de configuration de votre base de données.
3. Revenez à l'étape 3 et sélectionnez Configuration dans le panneau de navigation.
6. Accédez à la section Database connection (Connexion de la base de données) du panneau.



7. Sélectionnez Decouple database (Découpler la base de données).
8. Sélectionnez Apply (Appliquer) pour lancer l'opération de découplage de la base de données.

Le paramètre de stratégie de suppression détermine le résultat de la base de données et la durée nécessaire pour découpler la base de données.

- Si la stratégie de suppression est définie sur Delete, la base de données est supprimée. L'opération peut prendre environ 10 à 20 minutes selon la taille de la base de données.
- Si la stratégie de suppression est définie sur Snapshot, un instantané de la base de données est créé. Ensuite, la base de données est supprimée. La durée requise pour ce processus varie en fonction de la taille de la base de données.
- Si la stratégie de suppression est définie sur Retain, la base de données reste opérationnelle en dehors de l'environnement Elastic Beanstalk. Généralement, moins de cinq minutes sont nécessaires pour découpler une base de données.

Si vous décidez de conserver la base de données de manière externe à votre environnement Elastic Beanstalk, vous devrez prendre des mesures supplémentaires afin de la configurer. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#). Si vous envisagez d'utiliser la base de données que vous découplez pour un environnement de production, vérifiez que le type de stockage utilisé par la base de données est adapté à votre charge de travail. Pour de plus

amples informations, consultez [Stockage d'instance de base de données](#) et [Modification d'une instance de base de données](#) dans le Guide de l'utilisateur Amazon RDS.

## Découplage d'une instance de base de données RDS à l'aide de fichiers de configuration

Vous pouvez découpler votre instance de base de données d'un environnement Elastic Beanstalk sans affecter l'état de l'environnement. L'instance de base de données suit la stratégie de suppression de base de données appliquée lorsque la base de données a été découpée.

Les deux options requises pour découpler la base de données se trouvent dans l'espace de noms [the section called "aws:rds:dbinstance" \(p. 725\)](#). Ce sont les suivants :

- L'option `DBDeletionPolicy` définit la stratégie de suppression. Elle peut être définie sur les valeurs suivantes : `Snapshot`, `Delete` ou `Retain`. Ces valeurs sont décrites dans [Cycle de vie de base de données \(p. 618\)](#) de cette rubrique.
- L'option `HasCoupledDatabase` détermine si votre environnement possède une base de données couplée.
  - Si elle bascule sur `true`, Elastic Beanstalk crée une nouvelle instance de base de données couplée à votre environnement.
  - Si elle bascule sur `false`, Elastic Beanstalk commence à découpler l'instance de base de données de votre environnement.

Si vous souhaitez modifier la configuration de votre base de données avant de la découpler, appliquez d'abord, au cours d'une opération distincte, toutes les modifications de configuration. Cela comprend de modifier la configuration `DBDeletionPolicy`. Une fois vos modifications appliquées, exécutez une commande distincte pour définir l'option de découplage. Si, en plus du paramètre de découplage, vous soumettez d'autres paramètres de configuration, les autres paramètres d'option de configuration sont ignorés lorsque le paramètre de découplage est appliqué.

### Warning

Il est important que vous exécutiez les commandes pour appliquer les paramètres `DBDeletionPolicy` et `HasCoupledDatabase` en deux opérations distinctes. Si la stratégie de suppression active est déjà définie sur `Delete` ou `Snapshot`, vous risquez de perdre la base de données. La base de données suit la stratégie de suppression actuellement active, plutôt que la stratégie de suppression en attente souhaitée.

### Pour découpler une instance de base de données d'un environnement

Suivez ces étapes pour découpler la base de données de votre environnement Elastic Beanstalk. Vous pouvez utiliser l'interface de ligne de commande (CLI) EB ou l'AWS CLI pour terminer les étapes. Pour de plus amples informations, consultez [Personnalisation avancée de l'environnement avec des fichiers de configuration \(p. 737\)](#).

1. Si vous souhaitez modifier la stratégie de suppression, configurez un fichier de configuration au format suivant. Dans cet exemple, la stratégie de suppression est définie de manière à être conservée.

### Example

```
option_settings:  
  aws:rds:dbinstance:  
    DBDeletionPolicy: Retain
```

2. Exédez la commande à l'aide de votre outil préféré pour terminer la mise à jour de configuration.
3. Configurez un fichier de configuration afin de définir `HasCoupledDatabase` sur `false`.

### Example

```
option_settings:  
  aws:rds:dbinstance:  
    HasCoupledDatabase: false
```

- Exécutez la commande à l'aide de votre outil préféré pour terminer la mise à jour de configuration.

Le paramètre de stratégie de suppression détermine le résultat de la base de données et la durée nécessaire pour découpler la base de données.

- Si la stratégie de suppression est définie sur `Delete`, la base de données est supprimée. L'opération peut prendre environ 10 à 20 minutes selon la taille de la base de données.
- Si la stratégie de suppression est définie sur `Snapshot`, un instantané de la base de données est créé. Ensuite, la base de données est supprimée. La durée requise pour ce processus varie en fonction de la taille de la base de données.
- Si la stratégie de suppression est définie sur `Retain`, la base de données reste opérationnelle en dehors de l'environnement Elastic Beanstalk. Généralement, moins de cinq minutes sont nécessaires pour découpler une base de données.

Si vous décidez de conserver la base de données de manière externe à votre environnement Elastic Beanstalk, vous devrez prendre des mesures supplémentaires afin de la configurer. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#). Si vous envisagez d'utiliser la base de données que vous découpez pour un environnement de production, vérifiez que le type de stockage utilisé par la base de données est adapté à votre charge de travail. Pour de plus amples informations, consultez [Stockage d'instance de base de données](#) et [Modification d'une instance de base de données](#) dans le Guide de l'utilisateur Amazon RDS.

## Sécurité de votre environnement AWS Elastic Beanstalk

Elastic Beanstalk propose plusieurs options permettant de contrôler la sécurité de votre environnement et des instances Amazon EC2 qu'il contient. Cette rubrique explique comment configurer ces options.

### Sections

- [Configuration de la sécurité de votre environnement \(p. 626\)](#)
- [Espaces de noms de configuration de la sécurité de l'environnement \(p. 628\)](#)

## Configuration de la sécurité de votre environnement

Vous pouvez modifier la configuration de sécurité de votre environnement Elastic Beanstalk dans la console Elastic Beanstalk.

Pour configurer la sécurité de l'environnement dans la console Elastic Beanstalk

- Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
- Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Security (Sécurité), choisissez Edit (Modifier).

Les paramètres suivants sont disponibles.

#### Paramètres

- Rôle de service (p. 628)
- EC2 key pair (p. 628)
- Profil d'instance IAM (p. 628)

The screenshot shows the 'Modify security' configuration page for the environment 'GettingStartedApp-env'. The top navigation bar includes 'Elastic Beanstalk > Environments > GettingStartedApp-env > Configuration'. The main section is titled 'Modify security'. It contains three configuration groups: 'Service role', 'Virtual machine permissions', and 'AWS Lambda permissions'. Each group has a dropdown menu for selecting roles or profiles, a 'Choose' button, and a 'Cancel' or 'Continue' button at the bottom right.

Elastic Beanstalk > Environments > GettingStartedApp-env > Configuration

## Modify security

**Service role**

Service role

aws-elasticbeanstalk-service-role

**Virtual machine permissions**

EC2 key pair

-- Choose a key pair --

IAM instance profile

aws-elasticbeanstalk-ec2-role

Cancel Continue

## Rôle de service

Sélectionnez un [rôle de service \(p. 926\)](#) à associer à votre environnement Elastic Beanstalk. Elastic Beanstalk assume la fonction de service au moment où il accède à d'autres services AWS en votre nom. Pour plus d'informations, consultez [Gestion des rôles de service Elastic Beanstalk \(p. 926\)](#).

## EC2 key pair

Vous pouvez vous connecter en toute sécurité aux instances Amazon Elastic Compute Cloud (Amazon EC2) provisionnées pour votre application Elastic Beanstalk avec une paire de clés Amazon EC2. Pour obtenir des instructions sur la création d'une paire de clés, veuillez consulter [Créer une paire de clés à l'aide de Amazon EC2](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

### Note

Lorsque vous créez une paire de clés, Amazon EC2 stocke une copie de votre clé publique. Si vous n'en avez plus besoin pour vous connecter aux instances de l'environnement, vous pouvez la supprimer d'Amazon EC2. Pour de plus amples informations, veuillez consulter la section [Suppression de votre paire de clés](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Choisissez une paire de clés EC2 depuis le menu déroulant pour attribuez-la aux instances de votre environnement. Lorsque vous assignez une paire de clés, la clé publique est stockée sur l'instance pour authentifier la clé privée, que vous stockez localement. La clé privée n'est jamais stockée dans AWS.

Pour de plus amples informations sur la connexion à des instances Amazon EC2, veuillez consulter [Connectez-vous à votre instance](#) et [Connectez-vous à vos instances Linux/UNIX depuis Windows avec PuTTY](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

## Profil d'instance IAM

Un [profil d'instance \(p. 22\)](#) est un rôle IAM qui est appliqué aux instances lancées dans votre environnement Elastic Beanstalk. Les instances Amazon EC2 assument le rôle de profil d'instance pour signer des requêtes vers AWS et accéder à des API, par exemple, pour télécharger des journaux sur Amazon S3.

La première fois que vous créez un environnement dans la console Elastic Beanstalk, Elastic Beanstalk vous invite à créer un profil d'instance avec un ensemble d'autorisations par défaut. Vous pouvez ajouter des autorisations à ce profil pour fournir à vos instances un accès à d'autres services AWS. Pour plus d'informations, consultez [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#).

## Espaces de noms de configuration de la sécurité de l'environnement

Elastic Beanstalk propose des [options de configuration \(p. 658\)](#) dans les espaces de noms suivants, vous permettant de personnaliser la sécurité de votre environnement :

- [aws:elasticbeanstalk:environment \(p. 700\)](#) – Configurez le rôle de service de l'environnement à l'aide de l'option `ServiceRole`.
- [aws:autoscaling:launchconfiguration \(p. 681\)](#) – Configurez les autorisations pour les instances Amazon EC2 de l'environnement à l'aide des options `EC2KeyName` et `IamInstanceProfile`.

L'interface de ligne de commande EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Pour de plus amples informations, veuillez consulter [Valeurs recommandées \(p. 660\)](#).

# Balisage des ressources dans vos environnements Elastic Beanstalk

Vous pouvez appliquer des balises à vos environnements AWS Elastic Beanstalk. Les balises sont des paires clé-valeur associées aux ressources AWS. Pour de plus amples informations sur le balisage des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur de balise, et les types de ressources pris en charge, veuillez consulter [Balisage des ressources d'application Elastic Beanstalk \(p. 423\)](#).

Elastic Beanstalk applique des balises d'environnement à la ressource d'environnement elle-même, ainsi qu'à d'autres ressources AWS créées par Elastic Beanstalk pour l'environnement. Vous pouvez utiliser des balises pour gérer les autorisations au niveau de la ressource spécifique d'un environnement. Pour de plus amples informations, veuillez consulter [Balisage de vos ressources Amazon EC2](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Par défaut, Elastic Beanstalk applique quelques balises à votre environnement :

- `elasticbeanstalk:environment-name` : Nom de l'environnement.
- `elasticbeanstalk:environment-id` : ID de l'environnement.
- `Name` : aussi le nom de l'environnement. `Name` est utilisé dans le tableau de bord Amazon EC2 pour identifier et trier les ressources.

Vous ne pouvez pas modifier ces balises par défaut.

Vous pouvez spécifier des balises lorsque vous créez l'environnement Elastic Beanstalk. Dans un environnement existant, vous pouvez ajouter ou supprimer des balises, ainsi que mettre à jour les valeurs des balises existantes. Un environnement peut avoir jusqu'à 50 balises, y compris les balises par défaut.

## Ajout de balises lors de la création de l'environnement

Lorsque vous utilisez la console Elastic Beanstalk pour créer un environnement, vous pouvez spécifier des clés et valeurs de balise sur la page de configuration [Modify tags \(Modifier les balises\)](#) de l'assistant [Create New Environment \(Créer un nouvel environnement\) \(p. 442\)](#).

Elastic Beanstalk > Applications > getting-started-app

### Modify tags

Apply up to 50 tags to the resources in your environment in addition to the default tags.

Key	Value
mytag1	value1

Add tag

49 remaining

Si vous utilisez l'interface de ligne de commande EB pour créer des environnements, utilisez l'option --tags avec [eb create \(p. 1078\)](#) pour ajouter des balises.

```
~/workspace/my-app$ eb create --tags mytag1=value1,mytag2=value2
```

Avec l'AWS CLI ou d'autres clients basés sur une API, utilisez le paramètre --tags dans la commande [create-environment](#).

```
$ aws elasticbeanstalk create-environment \
  --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \
  --application-name my-app --environment-name my-env --cname-prefix my-app --version-
label v1 --template-name my-saved-config
```

Les [configurations enregistrées \(p. 663\)](#) incluent des balises définies par l'utilisateur. Lorsque vous appliquez une configuration enregistrée contenant des balises lors de la création de l'environnement, ces balises s'appliquent au nouvel environnement tant que vous ne spécifiez pas de nouvelles balises. Si vous ajoutez des balises à un environnement via l'une des méthodes précédentes, les balises définies dans la configuration enregistrée sont ignorées.

## Gestion des balises d'un environnement existant

Vous pouvez ajouter, mettre à jour et supprimer des balises dans un environnement Elastic Beanstalk existant. Elastic Beanstalk applique les modifications aux ressources de votre environnement.

Vous ne pouvez cependant pas modifier les balises par défaut qu'Elastic Beanstalk applique à votre environnement.

Pour gérer les balises d'un environnement dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, sélectionnez Tags.

La page de gestion des balises affiche la liste des balises qui existent actuellement dans l'environnement.

The screenshot shows the 'Tags' section of the AWS Elastic Beanstalk console. At the top, the navigation path is: Elastic Beanstalk > Environments > GettingStartedApp-env > Tags. The main title is 'Tags for GettingStartedApp-env'. A note says: 'Apply up to 47 tags in addition to the default tags to the resources in your environment. You can use tags to group environments. A tag is a key-value pair. The key must be unique within the environment and is case-sensitive.' Below this, there is a table with two columns: 'Key' and 'Value'. The first row shows 'elasticbeanstalk:environment-id' with value 'e-cubmdjm6ga'. The second row shows 'elasticbeanstalk:environment-name' with value 'GettingStartedApp-env'. There are also rows for 'Name' (value 'GettingStartedApp-env'), 'mytag1' (value 'value1'), and 'mytag2' (value 'value2'). At the bottom left is a button 'Add tag' and a note '45 remaining'.

4. Ajoutez, modifiez ou supprimez les balises :

- Pour ajouter une balise, tapez-la dans les zones vides en bas de la liste. Pour ajouter une autre balise, choisissez Add tag (Ajouter une balise) et Elastic Beanstalk ajoute une autre paire de cases vides.
- Pour mettre à jour la valeur ou la clé d'une balise, modifiez la zone correspondante dans la ligne de la balise.
- Pour supprimer une balise, choisissez Remove (Retirer) en regard de la zone de valeur de la balise.

5. Choisissez Apply (Appliquer).

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour les environnements, utilisez [eb tags \(p. 1118\)](#) pour ajouter, mettre à jour, supprimer ou répertorier des balises.

Par exemple, la commande suivante répertorie les balises de votre environnement par défaut.

```
~/workspace/my-app$ eb tags --list
```

La commande suivante met à jour la balise mytag1 et supprime la balise mytag2.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2
```

Pour obtenir une liste complète des options et d'autres exemples, consultez [eb tags \(p. 1118\)](#).

Avec l'AWS CLI ou d'autres clients basés sur l'API, utilisez la commande [list-tags-for-resource](#) pour afficher les balises d'un environnement.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:environment/my-app/my-env"
```

Utilisez la commande [update-tags-for-resource](#) pour ajouter, mettre à jour ou supprimer des balises dans un environnement.

```
$ aws elasticbeanstalk update-tags-for-resource \  
  --tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \  
  --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:environment/my-app/my-env"
```

Spécifiez les balises à ajouter et les balises à mettre à jour dans le paramètre `--tags-to-add` de `update-tags-for-resource`. Une balise inexistante est ajoutée et la valeur d'une balise existante est mise à jour.

#### Note

Pour utiliser ces deux commandes de l'AWS CLI avec un environnement Elastic Beanstalk, vous avez besoin de l'ARN de l'environnement. Vous pouvez extraire l'ARN à l'aide de la commande suivante.

```
$ aws elasticbeanstalk describe-environments
```

## Propriétés de l'environnement et autres paramètres de logiciel

Utilisez la page [Modify software configuration](#) (Modifier la configuration du logiciel) pour configurer le logiciel sur les instances Amazon Elastic Compute Cloud (Amazon EC2) exécutées sur votre application. Vous pouvez configurer les propriétés de l'environnement, le débogage AWS X-Ray, le stockage et le streaming des journaux d'instance et les paramètres spécifiques à la plateforme.

## Modify software

The following settings control platform behavior and let you pass key-value pairs in as OS environment variables. [Learn more](#)

### Platform options

Target .NET runtime

4.0

Enable 32-bit applications

False

### AWS X-Ray

X-Ray daemon

#### Rubriques

- [Configurer les paramètres spécifiques à la plateforme \(p. 633\)](#)
- [Configuration des propriétés de l'environnement \(p. 634\)](#)
- [Espaces de noms des paramètres de logiciel \(p. 636\)](#)
- [Accès aux propriétés de l'environnement \(p. 637\)](#)
- [Configuration du débogage AWS X-Ray \(p. 638\)](#)
- [Affichage de vos journaux d'environnement Elastic Beanstalk \(p. 641\)](#)

## Configurer les paramètres spécifiques à la plateforme

Outre l'ensemble d'options standard disponibles pour tous les environnements, la plupart des plateformes Elastic Beanstalk vous permettent de spécifier des paramètres propres à une langue ou à l'infrastructure. Ceux-ci apparaissent dans la section Platform options (Options de la plateforme) dans la page Modify software (Modifier le logiciel) et peuvent prendre les formes suivantes.

- Propriétés de l'environnement prédéfinies – La plateforme Ruby utilise des propriétés d'environnement pour les paramètres d'infrastructure comme `RACK_ENV` et `BUNDLE_WITHOUT`.
- Propriétés de l'environnement d'espace réservé – La plateforme Tomcat définit une propriété d'environnement nommée `JDBC_CONNECTION_STRING` qui n'a aucune valeur définie. Ce type de paramètre était plus fréquent sur les anciennes versions de la plateforme.
- Options de configuration – La plupart des plateformes définissent des [options de configuration \(p. 658\)](#) dans des espaces de noms propres à la plateforme ou partagés tels que `aws:elasticbeanstalk:xray` ou `aws:elasticbeanstalk:container:python`.

Pour configurer les paramètres spécifiques à la plateforme dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Sous Platform options (Options de la plateforme), apportez les modifications nécessaires aux paramètres des options.
6. Choisissez Apply.

Pour plus d'informations sur les options propres aux plateformes, et sur l'obtention des valeurs de propriété d'environnement dans votre code, reportez-vous à la rubrique relative à la plateforme pour votre langue ou votre infrastructure :

- Docker – [the section called “Configuration de l'environnement” \(p. 85\)](#)
- Go – [Utilisation de la plateforme Go Elastic Beanstalk \(p. 101\)](#)
- Java SE – [Utilisation de la plateforme Java SE Elastic Beanstalk \(p. 129\)](#)
- Tomcat : – [Utilisation de la plateforme Elastic Beanstalk Tomcat \(p. 118\)](#)
- .NET Core sous Linux – [Utilisation de la plateforme .NET Core sous Linux \(p. 161\)](#)
- .NET – [Utilisation de la plateforme .NET Elastic Beanstalk \(p. 196\)](#)
- Node.js – [Utilisation de la plateforme Elastic Beanstalk Node.js \(p. 255\)](#)
- PHP – [Utilisation de la plateforme PHP Elastic Beanstalk \(p. 293\)](#)
- Python – [Utilisation de la plateforme Python Elastic Beanstalk \(p. 359\)](#)
- Ruby – [Utilisation de la plateforme Elastic Beanstalk Ruby \(p. 387\)](#)

## Configuration des propriétés de l'environnement

Vous pouvez utiliser des propriétés d'environnement pour transmettre des secrets, des points de terminaison, des paramètres de débogage et d'autres informations à votre application. Les propriétés d'environnement vous aident à exécuter votre application dans plusieurs environnements et pour différents objectifs tels que le développement, le test, la phase intermédiaire et la production.

En outre, lorsque vous [ajoutez une base de données à votre environnement \(p. 617\)](#), Elastic Beanstalk définit les propriétés d'environnement, comme `RDS_HOSTNAME`, que vous pouvez lire dans le code de votre application pour construire un objet ou une chaîne de connexion.

#### Variables d'environnement

Dans la plupart des cas, les propriétés d'environnement sont transmises à votre application sous forme de variables d'environnement, mais le comportement dépend de la plateforme. Par exemple, [la plateforme Java SE \(p. 129\)](#) définit des variables d'environnement que vous récupérez avec `System.getenv`, tandis que [la plateforme Tomcat \(p. 118\)](#) définit des propriétés système Java que vous récupérez avec `System.getProperty`. En général, les propriétés ne sont pas visibles si vous vous connectez à une instance et exécutez `env`.

Pour configurer les propriétés d'environnement dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Sous Propriétés de l'environnement, saisissez les paires clé/valeur.

The screenshot shows the 'Environment properties' configuration page. At the top, it says 'The following properties are passed in the application as environment properties.' Below this is a table with two columns: 'Name' and 'Value'. There are three rows in the table:

Name	Value
JDBC_CONNECTION_STRING	<input type="text"/>
XRAY_ENABLED	{ "Fn::GetOptionSetting" : {"N:
	<input type="text"/>

6. Choisissez Apply.

#### Limites des propriétés de l'environnement

- Les clés peuvent contenir des caractères alphanumériques et les symboles suivants : \_ . : / + \ - @

Les symboles répertoriés sont valides pour les clés de propriété d'environnement, mais peuvent ne pas être valides pour les noms de variable d'environnement sur la plateforme de votre environnement. Pour assurer une compatibilité avec toutes les plateformes, limitez les propriétés d'environnement au schéma suivant : [A-Z\_][A-Z0-9\_]\*

- Les valeurs peuvent contenir des caractères alphanumériques, des espaces et les symboles suivants : \_ . : / = + \ - @ ' "

Note

Les guillemets simples et doubles dans les valeurs doivent être placés dans une séquence d'échappement.

- Les clés peuvent contenir jusqu'à 128 caractères. Les valeurs peuvent contenir jusqu'à 256 caractères.
- Les clés et les valeurs sont sensibles à la casse.

- La taille combinée de toutes les propriétés d'environnement ne peut pas dépasser 4 096 octets lorsqu'elles sont stockées en tant que chaînes au format `clé=valeur`.

## Espaces de noms des paramètres de logiciel

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour définir des options de configuration et exécuter d'autres tâches de configuration d'instance pendant les déploiements. Les options de configuration peuvent être définies par le service Elastic Beanstalk ou la plateforme que vous utilisez et sont organisées en espaces de noms.

Vous pouvez utiliser des [fichiers de configuration \(p. 737\)](#) Elastic Beanstalk pour définir les propriétés de l'environnement et les options de configuration dans votre code source. Utilisez l'espace de noms `aws:elasticbeanstalk:application:environment (p. 697)` pour définir les propriétés de l'environnement.

Example `.ebextensions/options.config`

```
option_settings:  
  aws:elasticbeanstalk:application:environment:  
    API_ENDPOINT: www.example.com/api
```

Si vous utilisez des fichiers de configuration ou des modèles AWS CloudFormation pour créer des [ressources personnalisées \(p. 759\)](#), vous pouvez utiliser une fonction AWS CloudFormation pour obtenir des informations sur la ressource et l'attribuer de manière dynamique à une propriété d'environnement pendant le déploiement. L'exemple suivant du référentiel [elastic-beanstalk-samples](#) GitHub utilise la [fonction Ref \(p. 764\)](#) pour obtenir l'ARN d'une rubrique Amazon SNS qu'il crée et l'affecte à une propriété d'environnement nommée `NOTIFICATION_TOPIC`.

### Notes

- Si vous utilisez une fonction AWS CloudFormation pour définir une propriété d'environnement, la console Elastic Beanstalk affiche la valeur de la propriété avant l'évaluation de la fonction. Vous pouvez utiliser le [script de plateforme get-config \(p. 43\)](#) pour confirmer les valeurs des propriétés d'environnement disponibles pour votre application.
- La plateforme [Docker multiconteneurs \(p. 64\)](#) n'utilise pas AWS CloudFormation pour créer des ressources de conteneur. Par conséquent, cette plateforme ne prend pas en charge la définition des propriétés d'environnement à l'aide de fonctions AWS CloudFormation.

Example `.Ebextensions/sns-topic.config`

```
Resources:  
  NotificationTopic:  
    Type: AWS::SNS::Topic  
  
option_settings:  
  aws:elasticbeanstalk:application:environment:  
    NOTIFICATION_TOPIC: `{"Ref" : "NotificationTopic"}'
```

Vous pouvez également utiliser cette fonctionnalité pour propager les informations à partir des [pseudoparamètres AWS CloudFormation](#). Cet exemple obtient la région actuelle et l'affecte à une propriété nommée `AWS_REGION`.

Example `.Ebextensions/env-regionname.config`

```
option_settings:  
  aws:elasticbeanstalk:application:environment:
```

```
AWS_REGION: `{"Ref" : "AWS::Region"}'
```

La plupart des plateformes Elastic Beanstalk définissent des espaces de noms supplémentaires avec des options pour configurer le logiciel qui s'exécute sur l'instance, tel que le proxy inversé qui relaie des requêtes pour votre application. Pour plus d'informations sur les espaces de noms disponibles pour votre plateforme, consultez les ressources suivantes :

- Go – [Espaces de noms de la configuration Go \(p. 103\)](#)
- Java SE – [Espaces de noms de la configuration Java SE \(p. 131\)](#)
- Tomcat : – [Espaces de noms de la configuration Tomcat \(p. 121\)](#)
- .NET Core sous Linux – [Espace de noms de la configuration .NET Core sous Linux \(p. 163\)](#)
- .NET – [Espace de noms aws:elasticbeanstalk:container:dotnet:apppool \(p. 197\)](#)
- Node.js – [Espaces de noms de la configuration Node.js \(p. 257\)](#)
- PHP – [Espace de noms aws:elasticbeanstalk:container:php:phpini \(p. 295\)](#)
- Python – [Espaces de noms de la configuration Python \(p. 361\)](#)
- Ruby – [Espaces de noms de configuration Ruby \(p. 389\)](#)

Elastic Beanstalk fournit de nombreuses options de configuration pour personnaliser votre environnement. En plus des fichiers de configuration, vous pouvez également définir des options de configuration à l'aide de la console, de configurations enregistrées, de la CLI EB ou d'AWS CLI. Pour plus d'informations, consultez [Options de configuration \(p. 658\)](#).

## Accès aux propriétés de l'environnement

Dans la plupart des cas, vous accédez aux propriétés de l'environnement dans le code de votre application sous forme d'une variable d'environnement. Toutefois, les propriétés d'environnement sont généralement transmises uniquement à l'application et ne peuvent pas être affichées via une connexion à une instance dans votre environnement et l'exécution de env.

- [Go \(p. 102\) – os.Getenv](#)

```
endpoint := os.Getenv("API_ENDPOINT")
```

- [Java SE \(p. 131\) – System.getenv](#)

```
String endpoint = System.getenv("API_ENDPOINT");
```

- [Tomcat \(p. 120\) – System.getProperty](#)

```
String endpoint = System.getProperty("API_ENDPOINT");
```

- [.NET Core sous Linux \(p. 162\) – Environment.GetEnvironmentVariable](#)

```
string endpoint = Environment.GetEnvironmentVariable("API_ENDPOINT");
```

- [.NET \(p. 197\) – appConfig](#)

```
NameValuePairCollection appConfig = ConfigurationManager.AppSettings;
string endpoint = appConfig["API_ENDPOINT"];
```

- [Node.js \(p. 256\) – process.env](#)

```
var endpoint = process.env.API_ENDPOINT
```

- [PHP \(p. 295\)](#) – `$_SERVER`

```
$endpoint = $_SERVER['API_ENDPOINT'];
```

- [Python \(p. 361\)](#) – `os.environ`

```
import os
endpoint = os.environ['API_ENDPOINT']
```

- [Ruby \(p. 388\)](#) – `ENV`

```
endpoint = ENV['API_ENDPOINT']
```

En dehors du code de l'application, par exemple dans un script exécuté pendant le déploiement, vous pouvez accéder aux propriétés de l'environnement avec le script de plateforme [get-config \(p. 43\)](#). Consultez le référentiel GitHub [elastic-beanstalk-samples](#) pour obtenir des exemples de configuration utilisant `get-config`.

## Configuration du débogage AWS X-Ray

Vous pouvez utiliser la console AWS Elastic Beanstalk ou un fichier de configuration pour exécuter le démon AWS X-Ray sur les instances de votre environnement. X-Ray est un service AWS qui collecte des données sur les demandes traitées par votre application et les utilise pour construire une cartographie des services qui vous permet d'identifier les problèmes liés à votre application et les opportunités d'optimisation.

### Note

Certaines régions ne proposent pas X-Ray. Si vous créez un environnement dans l'une de ces régions, vous ne pouvez pas exécuter le démon X-Ray sur les instances dans votre environnement.

Pour de plus amples informations sur les services AWS proposés dans chaque région, veuillez consulter le [Tableau des régions](#).



X-Ray fournit un kit de développement logiciel (SDK) que vous pouvez utiliser pour instrumentaliser votre code d'application et une application démon qui transmet les informations de débogage du SDK vers l'API X-Ray.

#### Plateformes prises en charge

Vous pouvez utiliser le kit SDK X-Ray avec les plateformes Elastic Beanstalk suivantes :

- Go - Version 2.9.1 et ultérieure
- Java 8 - version 2.3.0 et ultérieures
- Java 8 avec Tomcat 8 - version 2.4.0 et ultérieures
- Node.js - version 3.2.0 et ultérieures
- Windows Server - toutes les versions de plateforme publiées le 18 décembre 2016 ou après cette date
- Python version 2.5.0 et ultérieures

Sur les plateformes prises en charge, vous pouvez utiliser une option de configuration pour exécuter le démon X-Ray sur les instances de votre environnement. Vous pouvez activer le démon dans la [console Elastic Beanstalk \(p. 640\)](#) ou à l'aide d'un [fichier de configuration \(p. 640\)](#).

Pour pouvoir charger des données dans X-Ray, le démon X-Ray nécessite des autorisations IAM dans la stratégie gérée AWSXrayWriteOnlyAccess. Ces autorisations sont incluses dans [le profil d'instance Elastic Beanstalk \(p. 22\)](#). Si vous n'utilisez pas le profil d'instance par défaut, veuillez consulter [Autorisation du démon à envoyer des données à X-Ray](#) dans le Guide du développeur AWS X-Ray.

Le débogage avec X-Ray nécessite l'utilisation du kit SDK X-Ray. Pour obtenir des instructions et des exemples d'applications, veuillez consulter [Mise en route avec AWS X-Ray](#) dans le Guide du développeur AWS X-Ray.

Si vous utilisez une version de plateforme qui n'inclut pas le démon, vous pouvez malgré tout l'exécuter avec un script dans un fichier de configuration. Pour de plus amples informations, veuillez consulter [Téléchargement et exécution du Démon X-Ray manuellement \(avancé\)](#) dans le Guide du développeur AWS X-Ray.

#### Sections

- [Configuration du débogage \(p. 640\)](#)
- [Espace de noms aws:elasticbeanstalk:xray \(p. 640\)](#)

## Configuration du débogage

Vous pouvez activer le démon X-Ray sur un environnement en cours d'exécution dans la console Elastic Beanstalk.

Pour activer le débogage dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Dans la section AWS X-Ray, sélectionnez X-Ray daemon (démon X-Ray).
6. Choisissez Apply.

Vous pouvez également activer cette option lors de la création de l'environnement. Pour de plus amples informations, veuillez consulter [Assistant de création d'un environnement \(p. 442\)](#).

## Espace de noms aws:elasticbeanstalk:xray

Vous pouvez utiliser l'option `xRayEnabled` dans l'espace de noms `aws:elasticbeanstalk:xray` pour activer le débogage.

Pour activer automatiquement le débogage lorsque vous déployez votre application, définissez l'option dans un [fichier de configuration \(p. 737\)](#) dans votre code source, comme suit.

Example `.ebextensions/debugging.config`

```
option_settings:
```

```
aws:elasticbeanstalk:xray:  
XRayEnabled: true
```

## Affichage de vos journaux d'environnement Elastic Beanstalk

AWS Elastic Beanstalk offre deux façons d'afficher régulièrement des journaux à partir des instances Amazon EC2 qui exécutent votre application :

- Configurez votre environnement Elastic Beanstalk pour charger les journaux d'instance ayant subi une rotation dans le compartiment Amazon S3 de l'environnement.
- Configurez l'environnement pour diffuser les journaux d'instance vers Amazon CloudWatch Logs.

Lorsque vous configurez le streaming de journaux d'instance vers CloudWatch Logs, Elastic Beanstalk crée des groupes de journaux CloudWatch Logs pour les journaux de proxy et de déploiement sur les instances Amazon EC2, et transfère ces fichiers journaux vers CloudWatch Logs en temps réel. Pour plus d'informations sur les journaux d'instance, consultez [Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk \(p. 884\)](#).

Outre les journaux d'instance, si vous activez les [rapports améliorés sur l'état \(p. 837\)](#) pour votre environnement, vous pouvez configurer ce dernier de sorte à diffuser les informations d'état vers CloudWatch Logs. Lorsque l'état d'intégrité de l'environnement change, Elastic Beanstalk ajoute un enregistrement à un groupe de journaux d'intégrité, avec le nouvel état et une description de la cause du changement. Pour plus d'informations sur le streaming de l'intégrité d'environnement, consultez [Diffusion d'informations sur l'état de l'environnement Elastic Beanstalk vers Amazon CloudWatch Logs \(p. 903\)](#).

## Configuration de l'affichage des journaux d'instance

Pour afficher les journaux d'instance, vous pouvez activer la rotation des journaux d'instance et le streaming des journaux dans la console Elastic Beanstalk.

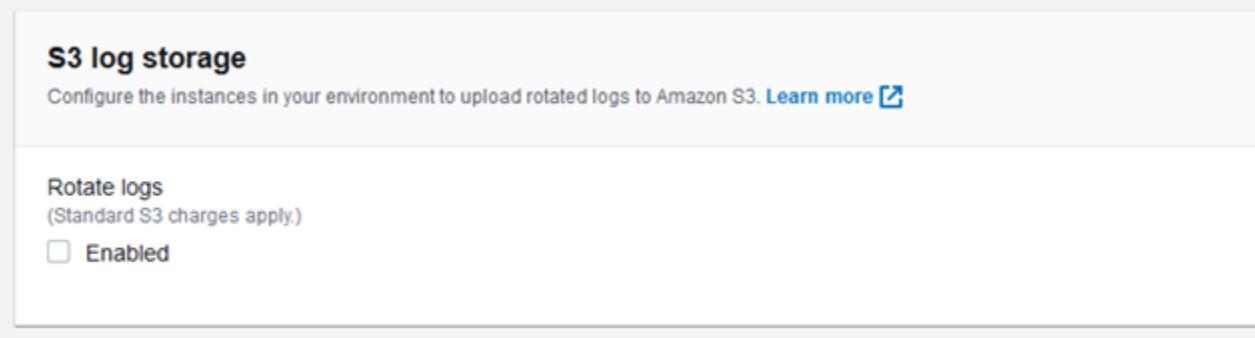
Pour configurer la rotation du journal d'instance et le streaming de journaux dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Dans la section S3 log storage (Stockage des journaux S3) sélectionnez Rotate logs (Rotation des journaux) pour activer le téléchargement des journaux ayant subi une rotation vers Amazon S3.



**S3 log storage**  
Configure the instances in your environment to upload rotated logs to Amazon S3. [Learn more](#)

Rotate logs  
(Standard S3 charges apply.)

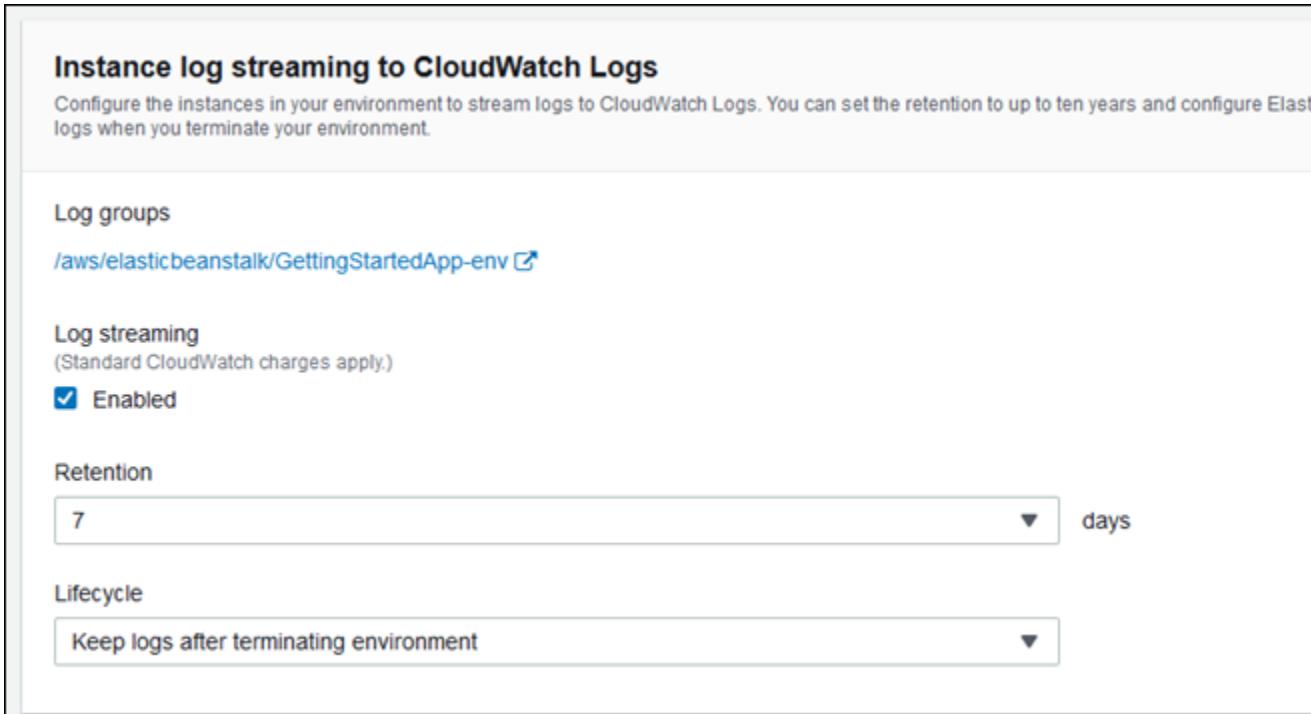
Enabled

6. Sous Instance log streaming to CloudWatch Logs (Streaming des journaux d'instance vers CloudWatch Logs), configurez les paramètres suivants :

- Log streaming (Streaming de journaux) – Sélectionnez cette option pour activer le streaming de journaux.
- Conservation – Spécifiez le nombre de jours de conservation des journaux dans CloudWatch Logs.
- Cycle de vie – Définissez ce paramètre sur Delete logs upon termination (Supprimer les journaux à la résiliation) pour supprimer immédiatement les journaux de CloudWatch Logs si l'environnement est arrêté, plutôt que d'attendre qu'ils arrivent à expiration.

7. Choisissez Apply.

Après avoir activé le streaming de journaux, vous pouvez revenir à la page ou à la catégorie de configuration Logiciels pour trouver le lien Groupes de journaux. Cliquez sur ce lien pour afficher vos journaux d'instance sur la console CloudWatch.



**Instance log streaming to CloudWatch Logs**  
Configure the instances in your environment to stream logs to CloudWatch Logs. You can set the retention to up to ten years and configure Elastic logs when you terminate your environment.

Log groups  
</aws/elasticbeanstalk/GettingStartedApp-env>

Log streaming  
(Standard CloudWatch charges apply.)

Enabled

Retention  
7 days

Lifecycle  
Keep logs after terminating environment

## Configuration de l'affichage des journaux d'intégrité d'environnement

Pour afficher les journaux d'intégrité d'environnement, vous pouvez activer le streaming des journaux d'intégrité d'environnement sur la console Elastic Beanstalk.

Pour configurer le streaming du journal d'intégrité de l'environnement dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Sous Streaming d'événements de santé vers CloudWatch Logs, configurez les paramètres suivants :
  - Streaming des journaux – Choisissez cette option pour activer la diffusion de journaux.
  - Conservation – Spécifiez le nombre de jours de conservation des journaux dans CloudWatch Logs.
  - Cycle de vie – Définissez ce paramètre sur Delete logs upon termination (Supprimer les journaux à la résiliation) pour supprimer immédiatement les journaux de CloudWatch Logs si l'environnement est arrêté, plutôt que d'attendre qu'ils arrivent à expiration.
6. Choisissez Apply.

Après avoir activé le streaming de journaux, vous pouvez revenir à la page ou à la catégorie de configuration Surveillance pour trouver le lien Groupe de journaux. Cliquez sur ce lien pour afficher les journaux d'état de votre environnement dans la console CloudWatch.

### Health event streaming to CloudWatch Logs

Configure Elastic Beanstalk to stream environment health events to CloudWatch Logs. You can set the retention period to a maximum of ten years and configure Elastic Beanstalk to delete the logs when you terminate your environment.

**Log group** /aws/elasticbeanstalk/GettingStartedApp-env/environment-health.log 

**Log streaming**  Enabled (Standard CloudWatch charges apply.)

**Retention** 7 days

**Lifecycle** Keep logs after terminating environment

## Espaces de noms de l'affichage des journaux

Les paramètres liés à l'affichage des journaux se trouvent dans les espaces de noms suivants :

- [aws:elasticbeanstalk:hostmanager \(p. 707\)](#) – Configurer le téléchargement de journaux tournés vers Amazon S3.
- [aws:elasticbeanstalk:cloudwatch:logs \(p. 698\)](#) – Configurer le streaming des journaux d'instance dans CloudWatch.
- [aws:elasticbeanstalk:cloudwatch:logs:health \(p. 698\)](#) – Configurer le streaming des journaux d'intégrité d'environnement dans CloudWatch.

# Notifications d'environnement Elastic Beanstalk avec Amazon SNS

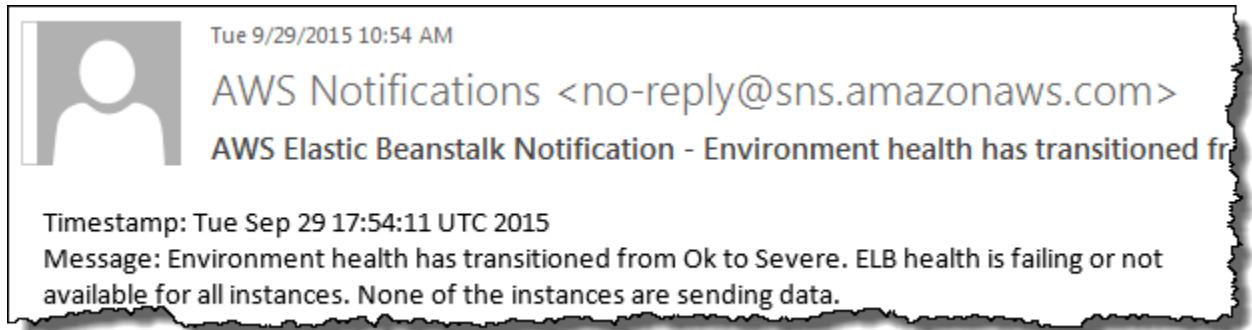
Vous pouvez configurer votre environnement AWS Elastic Beanstalk pour utiliser Amazon Simple Notification Service (Amazon SNS) et vous informer des événements importants qui affectent votre application. Pour recevoir des e-mails d'AWS lorsqu'une erreur survient ou que l'état de votre environnement change, spécifiez une adresse e-mail lorsque vous créez un environnement ou ultérieurement.

### Note

Elastic Beanstalk utilise Amazon SNS pour les notifications. Pour de plus amples informations sur la tarification Amazon SNS, veuillez consulter <https://aws.amazon.com/sns/pricing/>.

Lorsque vous configurez des notifications pour votre environnement, Elastic Beanstalk crée une rubrique Amazon SNS pour votre environnement en votre nom. Pour envoyer des messages à une rubrique Amazon SNS, Elastic Beanstalk doit disposer de l'autorisation requise. Pour de plus amples informations, veuillez consulter [Configuration des autorisations d'envoi de notifications \(p. 648\)](#).

Lorsqu'un [événement \(p. 879\)](#) notable se produit, Elastic Beanstalk envoie un message à la rubrique. Ensuite, Amazon SNS transmet les messages qu'il reçoit aux abonnés à la rubrique. Les événements importants incluent les erreurs de création d'environnement et toutes les modifications de l'[état d'un environnement ou d'une instance \(p. 837\)](#). Les événements relatifs à des opérations Amazon EC2 Auto Scaling (comme l'ajout et la suppression d'instances dans l'environnement) et les autres événements informatifs ne déclenchent pas l'envoi de notifications.



Vous pouvez entrer une adresse e-mail dans la console Elastic Beanstalk lorsque vous créez un environnement ou quelque temps après. Cela créera une rubrique Amazon SNS et vous y abonnera. Elastic Beanstalk gère le cycle de vie de la rubrique et la supprime lorsque votre environnement est arrêté ou lorsque vous supprimez votre adresse e-mail de la [console de gestion de l'environnement \(p. 429\)](#).

L'espace de noms `aws:elasticbeanstalk:sns:topics` fournit des options permettant de configurer une rubrique Amazon SNS à l'aide de fichiers de configuration, d'une interface de ligne de commande (CLI) ou d'un kit SDK. En employant l'une de ces méthodes, vous pouvez configurer le type d'abonné et le point de terminaison. Pour le type d'abonné, vous pouvez choisir une file d'attente Amazon SQS ou une URL HTTP.

Vous pouvez seulement activer ou désactiver les notifications Amazon SNS. Selon la taille et la composition de votre environnement, la fréquence des notifications envoyées à la rubrique peut être élevée. Pour configurer les notifications à envoyer dans des cas précis, vous disposez d'autres options. Vous pouvez [définir des règles axées sur les événements \(p. 906\)](#) avec Amazon EventBridge qui vous avertit lorsque Elastic Beanstalk émet des événements répondant à des critères spécifiques. En variante, vous pouvez [configurer votre environnement de sorte qu'il publie des métriques personnalisées \(p. 862\)](#) et [définir des alarmes Amazon CloudWatch \(p. 874\)](#) pour être informé lorsque ces métriques atteignent le seuil de défectuosité.

## Configuration des notifications à l'aide de la console Elastic Beanstalk

Vous pouvez entrer une adresse e-mail dans la console Elastic Beanstalk pour créer une rubrique Amazon SNS pour votre environnement.

Pour configurer les notifications dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Notifications, choisissez Edit (Modifier).
5. Entrez une adresse e-mail.

## Modify notifications

Email notifications

Enter an email address to receive email notifications for important events from your environment.

Email

Cancel Confirm

6. Choisissez Apply.

Si vous saisissez une adresse e-mail pour les notifications, Elastic Beanstalk crée une rubrique Amazon SNS pour votre environnement et ajoute un abonnement. Amazon SNS envoie un e-mail à l'adresse indiquée afin de confirmer l'abonnement. Vous devez cliquer sur le lien figurant dans l'e-mail de confirmation afin d'activer l'abonnement et de recevoir les notifications.

## Configuration des notifications à l'aide des options de configuration

Utilisez les options de l'espace de noms `aws:elasticbeanstalk:sns:topics` (p. 709) afin de configurer les notifications Amazon SNS pour votre environnement. Vous pouvez définir ces options à l'aide de [fichiers de configuration](#) (p. 737), d'une CLI ou d'un kit SDK.

- Point de terminaison de notification : l'adresse e-mail, la file d'attente Amazon SQS ou l'URL à laquelle les notifications doivent être envoyées. Si vous définissez cet option, une file d'attente SQS et un abonnement pour le point de terminaison spécifié sont créés. Si le point de terminaison n'est pas une adresse e-mail, vous devez également définir l'option `Notification Protocol`. SNS valide la valeur de `Notification Endpoint` en fonction de la valeur de `Notification Protocol`. Si vous configurez plusieurs fois cette option, vous créez des abonnements supplémentaires à la rubrique. Si vous supprimez cette option, la rubrique est supprimée.
- Protocole de notification : le protocole qui est utilisé pour envoyer les notifications au `Notification Endpoint`. La valeur par défaut de cette option est `email1`. Définissez cette option sur `email-json` pour envoyer des e-mails au format JSON, sur `http` ou `https` pour publier les notifications au format JSON à un point de terminaison HTTP, ou sur `sqs` pour envoyer les notifications à une file d'attente SQS.

### Note

AWS Lambda Les notifications ne sont pas prises en charge.

- ARN de rubrique de notification : une fois que vous avez défini un point de terminaison de notification pour votre environnement, consultez ce paramètre afin d'obtenir l'ARN de la rubrique SNS. Vous pouvez également définir cette option afin d'utiliser une rubrique SNS existante pour les notifications. La rubrique que vous associez à votre environnement via cette option n'est pas supprimée lorsque l'option est modifiée ou que l'environnement est arrêté.

Pour configurer les notifications Amazon SNS, vous devez disposer des autorisations requises. Si votre utilisateur IAM utilise la [stratégie utilisateur gérée \(p. 946\)](#) AdministratorAccess-AWSElasticBeanstalk d'Elastic Beanstalk, vous disposez déjà des autorisations requises pour configurer la rubrique Amazon SNS par défaut créée par Elastic Beanstalk pour votre environnement. Toutefois, si vous configurez une rubrique Amazon SNS qu'Elastic Beanstalk ne gère pas, vous devez ajoutez la stratégie suivante à votre rôle utilisateur.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "sns:SetTopicAttributes",  
                "sns:GetTopicAttributes",  
                "sns:Subscribe",  
                "sns:Unsubscribe",  
                "sns:Publish"  
            ],  
            "Resource": [  
                "arn:aws:sns:us-east-2:123456789012:sns_topic_name"  
            ]  
        }  
    ]  
}
```

- Nom de rubrique de notification : définissez cette option pour personnaliser le nom de la rubrique Amazon SNS utilisée pour les notifications relatives à l'environnement. Si une rubrique portant le même nom existe déjà, Elastic Beanstalk associe cette rubrique à l'environnement.

**Warning**

Si vous associez une rubrique SNS existante à un environnement avec `Notification Topic Name`, Elastic Beanstalk supprimera la rubrique si vous arrêtez l'environnement ou modifiez ce paramètre ultérieurement.

Si vous modifiez cette option, l'`Notification Topic ARN` est également modifié. Si une rubrique est déjà associée à l'environnement, Elastic Beanstalk supprime l'ancienne rubrique et crée une rubrique et un abonnement.

En utilisant un nom de rubrique personnalisé, vous devez également fournir l'ARN d'une rubrique personnalisée créée en externe. Comme la stratégie utilisateur géré ne détecte pas automatiquement une rubrique avec un nom personnalisé, vous devez fournir des autorisations Amazon SNS personnalisées à vos utilisateurs IAM. Utilisez une stratégie semblable à celle utilisée pour un ARN de rubrique personnalisée, mais incluez les ajouts suivants :

- Incluez deux autres actions dans la liste `Actions`, en particulier : `sns:CreateTopic` et `sns>DeleteTopic`
- Si vous modifiez le `Notification Topic Name` d'une rubrique personnalisée à une autre, vous devez inclure les ARN des deux rubriques dans la liste `Resource`. En variante, incluez une expression régulière qui couvre les deux rubriques. Ainsi, Elastic Beanstalk dispose des autorisations pour supprimer l'ancienne rubrique et en créer une nouvelle.

La CLI EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez [Valeurs recommandées \(p. 660\)](#) pour plus de détails.

## Configuration des autorisations d'envoi de notifications

Cette section présente les considérations de sécurité liées aux notifications envoyées avec Amazon SNS. Il existe deux cas distincts :

- Utilisez la rubrique Amazon SNS par défaut qu'Elastic Beanstalk crée pour votre environnement.
- Fournissez une rubrique Amazon SNS externe en utilisant les options de configuration.

La stratégie d'accès par défaut pour une rubrique Amazon SNS permet au seul propriétaire de la rubrique de la publier ou de s'y abonner. Toutefois, en utilisant la configuration de stratégie appropriée, Elastic Beanstalk peut être autorisé à publier sur une rubrique Amazon SNS dans l'un des deux cas décrits dans cette section. Vous trouverez davantage d'informations dans les sous-sections suivantes.

### Autorisations pour une rubrique par défaut

Lorsque vous configurez des notifications pour votre environnement, Elastic Beanstalk crée une rubrique Amazon SNS pour votre environnement. Pour envoyer des messages à une rubrique Amazon SNS, Elastic Beanstalk doit disposer de l'autorisation requise. Si votre environnement utilise le [rôle de service \(p. 926\)](#) généré pour lui par la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB, ou le [rôle lié au service de surveillance \(p. 935\)](#) de votre compte, vous n'avez pas besoin de faire quoi que ce soit d'autre. Ces rôles gérés contiennent l'autorisation nécessaire qui permet à Elastic Beanstalk d'envoyer des messages à la rubrique Amazon SNS.

Toutefois, si vous avez fourni un rôle de service personnalisé lorsque vous avez créé votre environnement, vérifiez que ce rôle de service personnalisé inclut bien la stratégie suivante.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "sns:Publish"  
            ],  
            "Resource": [  
                "arn:aws:sns:us-east-2:123456789012:ElasticBeanstalkNotifications*"  
            ]  
        }  
    ]  
}
```

### Autorisations pour une rubrique externe

[Configuration des notifications à l'aide des options de configuration \(p. 646\)](#) explique la façon dont vous pouvez remplacer la rubrique Amazon SNS qu'Elastic Beanstalk fournit avec une autre rubrique Amazon SNS. Si vous avez remplacé la rubrique, Elastic Beanstalk doit vérifier que vous êtes autorisé à publier dans cette rubrique SNS afin que vous puissiez associer la rubrique SNS à l'environnement. Vous devez disposer des éléments suivants `sns:Publish`. Le rôle de service utilise la même autorisation. Pour vérifier que c'est le cas, Elastic Beanstalk envoie une notification test à SNS dans le cadre de votre action de création ou de mise à jour de l'environnement. En cas d'échec de ce test, votre tentative de création ou de mise à jour de l'environnement échoue également. Elastic Beanstalk affiche un message détaillant la raison de cet échec.

Si vous fournissez un rôle de service personnalisé pour votre environnement, vérifiez que votre rôle de service personnalisé inclut bien la stratégie suivante pour permettre à Elastic Beanstalk d'envoyer des messages à la rubrique Amazon SNS. Dans le code suivant, remplacez `sns_topic_name` par le nom de la rubrique Amazon SNS que vous avez fourni dans les options de configuration.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "sns:Publish"  
            ],  
            "Resource": [  
                "arn:aws:sns:us-east-2:123456789012:sns_topic_name"  
            ]  
        }  
    ]  
}
```

Pour plus d'informations sur le contrôle d'accès Amazon SNS, veuillez consulter [Exemples de cas pour le contrôle d'accès Amazon SNS](#) dans le Guide du développeur Amazon Simple Notification Service.

## Configuration d'Amazon Virtual Private Cloud (Amazon VPC) avec Elastic Beanstalk

[Amazon Virtual Private Cloud](#) (Amazon VPC) est le service réseau qui achemine le trafic en toute sécurité vers les instances EC2 qui exécutent votre application dans Elastic Beanstalk. Si vous ne configurez pas de VPC lorsque vous lancez votre environnement, Elastic Beanstalk utilise le VPC par défaut.

Vous pouvez lancer votre environnement dans un VPC personnalisé afin de personnaliser les paramètres de mise en réseau et de sécurité. Elastic Beanstalk vous permet de choisir quels sous-réseaux utiliser pour vos ressources et comment configurer les adresses IP des instances et de l'équilibrer de charge dans votre environnement. Un environnement est lié à un VPC lorsque vous le créez, mais vous pouvez modifier les paramètres de sous-réseau et d'adresse IP dans un environnement d'exécution.

### Note

Si vous avez créé votre compte AWS avant le 4 décembre 2013, il se peut que certains de vos environnements utilisent la configuration réseau Amazon EC2-Classic dans certaines régions AWS au lieu d'Amazon VPC. Pour de plus amples informations sur la migration de vos environnements d'une configuration réseau EC2-Classic vers une configuration réseau VPC, veuillez consulter [Migration d'environnements Elastic Beanstalk d'EC2-Classic vers un VPC \(p. 653\)](#).

## Configuration des paramètres VPC dans la console Elastic Beanstalk

Si vous avez sélectionné un VPC personnalisé lors de la création de votre environnement, vous pouvez modifier ses paramètres VPC dans la console Elastic Beanstalk.

Pour configurer les paramètres VPC de votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Network (Réseau), choisissez Edit (Modifier).

Les paramètres suivants sont disponibles.

Options

- [VPC \(p. 650\)](#)
- [Visibilité de l'équilibreur de charge \(p. 650\)](#)
- [Sous-réseaux de l'équilibreur de charge \(p. 651\)](#)
- [Adresse IP publique d'instance \(p. 651\)](#)
- [Sous-réseaux d'instance \(p. 651\)](#)
- [Sous-réseaux de la base de données \(p. 652\)](#)

## VPC

Sélectionnez un VPC pour votre environnement. Vous ne pouvez modifier ce paramètre que lors de la création de l'environnement.

The screenshot shows the 'Modify network' section of the AWS Elastic Beanstalk configuration interface. At the top, a breadcrumb trail indicates the path: Elastic Beanstalk > Environments > GettingStartedApp-env > Configuration. Below this, the title 'Modify network' is displayed. A section titled 'Virtual private cloud (VPC)' is shown. Under 'VPC', there is a note: 'Launch your environment in a custom VPC instead of the default VPC. You can create a VPC and subnets in the VPC management console.' A dropdown menu shows the selected VPC: 'vpc-0f9c96ae77f3c49c1 (172.31.0.0/16) | private-public'. A 'Create custom VPC' button is also visible. On the right side of the interface, there is a small circular icon with a 'C' inside.

## Visibilité de l'équilibreur de charge

Dans un environnement à charge équilibrée, choisissez le schéma de l'équilibreur de charge. Par défaut, l'équilibreur de charge est public, avec une adresse IP et un nom de domaine publics. Si votre application ne gère que du trafic provenant de votre VPC ou d'un VPN connecté, désélectionnez cette option et choisissez des sous-réseaux privés pour votre équilibreur de charge afin de rendre l'équilibreur de charge interne et de désactiver l'accès depuis Internet.

## Sous-réseaux de l'équilibrer de charge

Dans un environnement à charge équilibrée, choisissez les sous-réseaux que votre équilibrer de charge utilise pour gérer le trafic. Dans le cas d'une application publique, choisissez des sous-réseaux publics. Utilisez des sous-réseaux dans plusieurs zones de disponibilité pour une haute disponibilité. Dans le cas d'une application interne, choisissez des sous-réseaux privés et désactivez la visibilité de l'équilibrer de charge.

The screenshot shows the 'Load balancer settings' section of the AWS Elastic Beanstalk configuration interface. It includes a note about assigning a load balancer to subnets across Availability Zones, a visibility dropdown set to 'Public', and a table for selecting load balancer subnets. The table lists four subnets: two in us-east-2a (public-a and private-a) and two in us-east-2b (private-b and public-b). Subnets public-a and private-b have checkboxes checked.

	Availability Zone	Subnet	CIDR	Name
<input checked="" type="checkbox"/>	us-east-2a	subnet-04a707767b8ca8023	172.31.0.0/24	public-a
<input type="checkbox"/>	us-east-2a	subnet-0c559eeeb1a89adb4	172.31.3.0/24	private-a
<input checked="" type="checkbox"/>	us-east-2b	subnet-034a813125cd2077a	172.31.2.0/24	private-b
<input type="checkbox"/>	us-east-2b	subnet-09a24e24e7f7359fa	172.31.1.0/24	public-b

## Adresse IP publique d'instance

Si vous choisissez des sous-réseaux publics pour les instances de votre application, activez des adresses IP publiques pour les rendre routables à partir d'Internet.

## Sous-réseaux d'instance

Choisissez des sous-réseaux pour les instances de votre application. Choisissez au moins un sous-réseau pour chaque zone de disponibilité utilisée par votre équilibrer de charge. Si vous choisissez des sous-réseaux privés pour vos instances, votre VPC doit avoir une passerelle NAT dans un sous-réseau public que les instances peuvent utiliser pour accéder à Internet.

### Instance settings

Choose a subnet in each AZ for the instances that run your application. To avoid exposing your instances to the Internet, run your instances and load balancer in public subnets. To run your load balancer and instances in the same public subnets, assign public IP addresses to the instances.

Public IP address  
Assign a public IP address to the Amazon EC2 instances in your environment.

### Instance subnets

Availability Zone	Subnet	CIDR	Name
<input type="checkbox"/> us-east-2a	subnet-04a707767b8ca8023	172.31.0.0/24	public-a
<input checked="" type="checkbox"/> us-east-2a	subnet-0c559eeeb1a89adb4	172.31.3.0/24	private-a
<input type="checkbox"/> us-east-2b	subnet-034a813125cd2077a	172.31.2.0/24	private-b
<input checked="" type="checkbox"/> us-east-2b	subnet-09a24e24e7f7359fa	172.31.1.0/24	public-b

Cancel Configure

## Sous-réseaux de la base de données

Lorsque vous exécutez une base de données Amazon RDS attachée à votre environnement Elastic Beanstalk, choisissez des sous-réseaux pour vos instances de base de données. Pour une haute disponibilité, rendez la base de données Multi-AZ et choisissez un sous-réseau pour chaque zone de disponibilité. Pour vous assurer que votre application puisse se connecter à votre base de données, exécutez les deux dans les mêmes sous-réseaux.

## Espace de noms aws:ec2:vpc

Vous pouvez utiliser les options de configuration de l'espace de noms [aws:ec2:vpc \(p. 696\)](#) pour configurer les paramètres réseau de votre environnement.

Le [fichier de configuration \(p. 737\)](#) suivant utilise des options dans cet espace de noms pour définir le VPC et les sous-réseaux de l'environnement pour une configuration publique-privée. Pour définir l'ID de VPC dans un fichier de configuration, le fichier doit être inclus dans le bundle de sources de l'application lors de la création de l'environnement. Consultez [Définition des options de configuration lors de la création de l'environnement \(p. 665\)](#) pour d'autres méthodes de configuration de ces paramètres lors de la création de l'environnement.

Example .ebextensions/vpc.config – Public-private

```
option_settings:
```

```
aws:ec2:vpc:  
    VPCId: vpc-087a68c03b9c50c84  
    AssociatePublicIpAddress: 'false'  
    ELBScheme: public  
    ELBSubnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2  
    Subnets: subnet-026c6117b178a9c45,subnet-0839e902f656e8bd1
```

Cet exemple illustre une configuration public-public, dans laquelle l'équilibrer de charge et les instances EC2 s'exécutent dans les mêmes sous-réseaux publics.

Example .ebextensions/vpc.config – Public-public

```
option_settings:  
    aws:ec2:vpc:  
        VPCId: vpc-087a68c03b9c50c84  
        AssociatePublicIpAddress: 'true'  
        ELBScheme: public  
        ELBSubnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2  
        Subnets: subnet-0fe6b36bcb0ffc462,subnet-032fe3068297ac5b2
```

## Migration d'environnements Elastic Beanstalk d'EC2-Classic vers un VPC

Cette rubrique décrit les différentes options pour procéder à la migration de vos environnements Elastic Beanstalk d'une plateforme réseau EC2-Classic vers un réseau [Amazon Virtual Private Cloud](#) (Amazon VPC).

Si vous avez créé votre compte AWS avant le 4 décembre 2013, il se peut que certains de vos environnements utilisent la configuration réseau EC2-Classic dans certaines régions Régions AWS . Tous les comptes AWS créés le 4 décembre 2013 ou après cette date sont déjà VPC uniquement dans chaque région AWS. Si Amazon EC2-Classic a été activé à la suite d'une demande de support, l'exception s'applique alors.

### Note

Vous pouvez afficher les paramètres de configuration réseau de votre environnement dans la catégorie Network configuration (Configuration réseau) de la page [Configuration overview \(p. 534\)](#) (Aperçu de la configuration) de la [console Elastic Beanstalk](#).

## Pourquoi migrer

Amazon EC2-Classic atteindra la fin de sa prise en charge standard le 15 août 2022. Pour éviter toute interruption de vos charges de travail, nous vous recommandons de procéder à la migration d'Amazon EC2-Classic vers un VPC avant le 15 août 2022. Nous vous demandons également de ne pas lancer de ressources AWS sur Amazon EC2-Classic à l'avenir et de plutôt utiliser Amazon VPC.

Lorsque vous procédez à la migration de vos environnements Elastic Beanstalk d'Amazon EC2-Classic vers Amazon VPC, vous devez créer un nouveau compte AWS. Vous devez également recréer vos environnements AWS EC2-Classic dans votre nouveau compte AWS. Aucun travail de configuration supplémentaire pour vos environnements n'est requis afin d'utiliser le VPC par défaut. Si le VPC par défaut ne répond pas à vos exigences, créez manuellement un VPC personnalisé et associez-le à vos environnements.

Sinon, si votre compte AWS existant dispose de ressources dont la migration vers un nouveau compte AWS est impossible, vous pouvez ajouter un VPC à votre compte actuel. Configurez ensuite vos environnements pour qu'ils utilisent le VPC.

Pour de plus amples informations, consultez le billet de blog [EC2-Classic Networking is Retiring - Here's How to Prepare](#).

## Migration d'un environnement depuis EC2-Classic vers un nouveau compte AWS (recommandé)

Si vous n'avez pas encore de compte AWS créé le ou après le 4 décembre 2013, créez un nouveau compte. Vous allez procéder à la migration de vos environnements vers ce nouveau compte.

1. Votre nouveau compte AWS fournit un VPC par défaut à ses environnements. Si vous n'avez pas besoin de créer un VPC personnalisé, passez à l'étape 2.

Vous pouvez créer un VPC personnalisé de l'une des manières suivantes :

- Créez rapidement un VPC à l'aide de l'assistant de console Amazon VPC et de l'une des options de configuration disponibles. Pour de plus amples informations, veuillez consulter [Configurations de l'assistant de la console Amazon VPC](#).
- Créez un VPC personnalisé sur la console Amazon VPC si vous disposez d'exigences plus spécifiques pour votre VPC. Nous vous recommandons cette opération, par exemple, si votre cas d'utilisation nécessite un nombre spécifique de sous-réseaux. Pour de plus amples informations, veuillez consulter [VPC et sous-réseaux](#).
- Créez un VPC à l'aide du référentiel [elastic-beanstalk-samples](#) sur le site web GitHub si vous préférez utiliser des modèles AWS CloudFormation avec vos environnements Elastic Beanstalk. Ce référentiel comprend des modèles AWS CloudFormation. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#).

### Note

Vous pouvez également créer un VPC personnalisé en même temps que vous recréez l'environnement dans votre nouveau compte AWS à l'aide de l'[assistant de création d'un nouvel environnement \(p. 442\)](#). Si vous utilisez l'assistant et choisissez de créer un VPC personnalisé, l'assistant vous redirige vers la console Amazon VPC.

2. Dans votre nouveau compte AWS, créez un nouvel environnement. Nous vous recommandons de créer un environnement qui inclut la même configuration que votre environnement existant dans le compte AWS à partir duquel vous procédez à la migration. Pour ce faire, utilisez l'une des méthodes suivantes :

### Note

Si votre nouvel environnement doit utiliser le même CNAME après la migration, vous devez d'abord résilier l'environnement d'origine sur la plateforme EC2-Classic. En conséquence, CNAME est libéré et prêt à être utilisé. Cependant, cela peut entraîner des temps d'arrêt pour cet environnement et le risque qu'un autre client sélectionne votre CNAME entre la résiliation de votre environnement EC2-Classic et la création du nouvel environnement. Pour de plus amples informations, veuillez consulter [Arrêt d'un environnement Elastic Beanstalk \(p. 464\)](#). Pour les environnements qui ont leur propre nom de domaine propriétaire, le CNAME n'a pas ce problème. Vous pouvez simplement mettre à jour votre système de noms de domaine (DNS) pour transférer les demandes à votre nouveau CNAME.

- Utilisez l'[assistant de création d'environnement \(p. 442\)](#) sur la [console Elastic Beanstalk](#). L'Assistant fournit une option pour créer un VPC personnalisé. Si vous ne choisissez pas de créer un VPC personnalisé, un VPC par défaut est affecté.
- Utilisez l'interface de ligne de commande Elastic Beanstalk (EB CLI) pour recréer votre environnement dans votre nouveau compte AWS. L'un des [exemples \(p. 1087\)](#) de la description de la commande eb create illustre la création d'un environnement dans un VPC personnalisé. Si vous ne fournissez pas d'ID de VPC, l'environnement utilise le VPC par défaut.

En utilisant cette approche, vous pouvez utiliser un fichier de configuration enregistré sur les deux comptes AWS. Par conséquent, vous ne devez pas saisir manuellement toutes les informations de configuration. Cependant, vous devez enregistrer les paramètres de configuration de l'environnement EC2-Classic dont vous procédez à la migration avec la commande [eb config save \(p. 1071\)](#). Copiez le fichier de configuration enregistré dans un nouveau répertoire pour le nouvel environnement de compte.

#### Note

Vous devez modifier certaines données du fichier de configuration enregistré avant de pouvoir les utiliser dans le nouveau compte. Vous devez également mettre à jour les informations relatives à votre compte précédent avec les bonnes données pour votre nouveau compte. Par exemple, vous devez remplacer l'Amazon Resource Name (ARN) du rôle AWS Identity and Access Management (IAM) par l'ARN du rôle IAM pour le nouveau compte.

Si vous utilisez la commande [eb create \(p. 1078\)](#) avec l'option `cfg`, le nouvel environnement est créé à l'aide du fichier de configuration enregistré spécifié. Pour de plus amples informations, veuillez consulter [Utilisation des configurations enregistrées par Elastic Beanstalk \(p. 779\)](#).

## Migration d'un environnement depuis EC2-Classic au sein de votre même compte AWS

Votre compte AWS existant peut comporter des ressources que vous ne pouvez pas migrer vers un nouveau compte AWS. Dans ce cas, vous devez recréer vos environnements et configurer manuellement un VPC pour chaque environnement que vous créez.

### Migrer vos environnements vers un VPC personnalisé

#### Prerequisites

Avant de commencer, vous devez disposer d'un VPC. Vous pouvez créer un VPC non par défaut (personnalisé) de l'une des manières suivantes :

- Créez rapidement un VPC à l'aide de l'assistant de console Amazon VPC et de l'une des options de configuration disponibles. Pour de plus amples informations, veuillez consulter [Configurations de l'assistant de la console Amazon VPC](#).
- Créez un VPC personnalisé sur la console Amazon VPC si vous disposez d'exigences plus spécifiques pour votre VPC. Nous vous recommandons cette opération, par exemple, si votre cas d'utilisation nécessite un nombre spécifique de sous-réseaux. Pour de plus amples informations, veuillez consulter [VPC et sous-réseaux](#).
- Créez un VPC à l'aide du référentiel [elastic-beanstalk-samples](#) sur le site web GitHub si vous préférez utiliser des modèles AWS CloudFormation avec vos environnements Elastic Beanstalk. Ce référentiel comprend des modèles AWS CloudFormation. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#).

Au cours des étapes suivantes, vous utilisez l'ID VPC généré et les ID de sous-réseau lorsque vous configurez le VPC dans le nouvel environnement.

1. Créez un nouvel environnement qui inclut la même configuration que votre environnement existant. Pour ce faire, utilisez l'une des méthodes suivantes :

#### Note

La fonctionnalité Configurations enregistrées peut vous aider à recréer vos environnements dans le nouveau compte. Cette fonction peut enregistrer la configuration d'un environnement

de manière à ce que vous puissiez l'appliquer lorsque vous créez ou mettez à jour d'autres environnements. Pour de plus amples informations, veuillez consulter [Utilisation des configurations enregistrées par Elastic Beanstalk \(p. 779\)](#).

- À l'aide de la [console Elastic Beanstalk](#), appliquez une configuration enregistrée à partir de votre environnement EC2-Classic lorsque vous configurez le nouvel environnement. Cette configuration utilise le VPC. Pour de plus amples informations, veuillez consulter [Utilisation des configurations enregistrées par Elastic Beanstalk \(p. 779\)](#).
  - À l'aide de l'interface de ligne de commande Elastic Beanstalk (EB CLI), exécutez la commande [eb create \(p. 1078\)](#) pour recréer votre environnement. Fournissez les paramètres de votre environnement d'origine et l'identifiant du VPC. L'un des [exemples \(p. 1087\)](#) de la description de la commande eb create illustre la création d'un environnement dans un VPC personnalisé.
  - Utilisez l'AWS Command Line Interface (AWS CLI) et recréez votre environnement à l'aide de la commande elasticbeanstalk create-environment. Fournissez les paramètres de votre environnement d'origine avec l'identifiant VPC. Pour obtenir des instructions, consultez [Création d'environnements Elastic Beanstalk avec l'interface de ligne de commande \(CLI\) AWS \(p. 465\)](#).
2. Échangez le CNAME de l'environnement existant avec celui du nouvel environnement. De cette façon, le nouvel environnement que vous avez créé peut être référencé avec l'adresse familière. Vous pouvez utiliser l'interface de ligne de commande (CLI) EB ou le AWS CLI.
- À l'aide de l'interface de ligne de commande (CLI) EB, permutez les CNAME d'environnement en exécutant la commande eb swap. Pour de plus amples informations, veuillez consulter [Utilisation de l'interface de ligne de commande Elastic Beanstalk \(EB\) \(p. 1027\)](#).
  - À l'aide de l'AWS CLI, permutez les CNAME d'environnement avec la commande [elasticbeanstalk swap-environment-cnames](#). Pour plus d'informations, consultez la [Référence des commandes d'AWS CLI](#).

## Nom de domaine de votre environnement Elastic Beanstalk

Par défaut, votre environnement est mis à la disposition des utilisateurs à un sous-domaine de [elasticbeanstalk.com](#). Lorsque vous [créez un environnement \(p. 439\)](#), vous pouvez choisir un nom d'hôte pour votre application. Le sous-domaine et le domaine sont autorenseignés sur [region.elasticbeanstalk.com](#).

Pour acheminer les utilisateurs vers votre environnement, Elastic Beanstalk effectue un enregistrement CNAME qui pointe vers l'équilibrEUR de charge de votre environnement. Vous pouvez voir l'URL de l'application de votre environnement avec la valeur actuelle du CNAME dans la page de [présentation de l'environnement \(p. 430\)](#) de la console Elastic Beanstalk.

Elastic Beanstalk > Environments > GettingStartedApp-env

**GettingStartedApp-env**

GettingStartedApp-env.bx7dx222kw.us-east-2.elasticbeanstalk.com

**Health**

Ok

**Running version**

Sample Application-3

Upload and deploy

Choisissez l'URL sur la page de présentation ou choisissez Go to environment (Aller à l'environnement) dans le volet de navigation pour accéder à la page web de votre application.

Vous pouvez changer le CNAME sur votre environnement en l'échangeant avec le CNAME d'un autre environnement. Pour obtenir des instructions, consultez [Déploiements bleu/vert avec Elastic Beanstalk \(p. 486\)](#).

Si vous possédez un nom de domaine, vous pouvez utiliser Amazon Route 53 pour le résoudre dans votre environnement. Vous pouvez acheter un nom de domaine avec Amazon Route 53, ou en utiliser un que vous achetez auprès d'un autre fournisseur.

Pour acheter un nom de domaine avec Route 53, veuillez consulter [Enregistrement d'un nouveau domaine](#) dans le Guide du développeur Amazon Route 53.

Pour en savoir plus sur l'utilisation d'un domaine personnalisé, veuillez consulter [Routage du trafic vers un environnement AWS Elastic Beanstalk](#) dans le Guide du développeur Amazon Route 53.

**Important**

Si vous résiliez un environnement, vous devez également supprimer tous les mappages CNAME que vous avez créés, car d'autres clients peuvent réutiliser un nom d'hôte disponible.

# Configuration d'environnements Elastic Beanstalk (niveau avancé)

Lorsque vous créez un environnement AWS Elastic Beanstalk, Elastic Beanstalk alloue et configure toutes les ressources AWS nécessaires pour exécuter et prendre en charge votre application. En plus de configurer le comportement de mise à jour et les métadonnées de votre environnement, vous pouvez personnaliser ces ressources en fournissant des valeurs pour les [options de configuration \(p. 658\)](#). Par exemple, vous pouvez ajouter une file d'attente Amazon SQS et une alarme relative à la longueur de la file d'attente, ou ajouter un cluster Amazon ElastiCache.

La plupart des options de configuration sont associées à des valeurs par défaut qui sont automatiquement appliquées par Elastic Beanstalk. Vous pouvez remplacer ces valeurs par défaut via des fichiers de configuration, des configurations enregistrées, des options de ligne de commande, ou en appelant directement l'API Elastic Beanstalk. L'interface de ligne de commande EB et la console Elastic Beanstalk appliquent également les valeurs recommandées pour certaines options.

Vous pouvez facilement personnaliser votre environnement en même temps que vous déployez votre version d'application, en incluant un fichier de configuration dans votre bundle source. Lorsque vous personnalisez le logiciel sur votre instance, il est plus avantageux d'utiliser un fichier de configuration que de créer une AMI personnalisée, car vous n'avez pas besoin de gérer un ensemble d'AMI.

Lorsque vous déployez vos applications, vous pouvez personnaliser et configurer le logiciel dont dépend votre application. Ces fichiers peuvent être des dépendances requises par l'application (par exemple, des packages supplémentaires provenant du référentiel yum) ou des fichiers de configuration (par exemple, pour que httpd.conf remplace des paramètres spécifiques qui sont définis par défaut par AWS Elastic Beanstalk).

## Rubriques

- [Options de configuration \(p. 658\)](#)
- [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#)
- [Utilisation des configurations enregistrées par Elastic Beanstalk \(p. 779\)](#)
- [Manifeste d'environnement \(env.yaml\) \(p. 785\)](#)
- [Utilisation d'une Amazon Machine Image \(AMI\) personnalisée \(p. 787\)](#)
- [Service de fichiers statiques \(p. 790\)](#)
- [Configuration de HTTPS pour votre environnement Elastic Beanstalk \(p. 792\)](#)

## Options de configuration

Elastic Beanstalk définit un grand nombre d'options de configuration que vous pouvez utiliser pour configurer le comportement de votre environnement et les ressources qu'il contient. Les options de configuration sont organisées en espaces de noms comme `aws:autoscaling:asg`, qui définit les options de groupe Auto Scaling d'un environnement.

La console Elastic Beanstalk et l'interface de ligne de commande (CLI) EB définissent les options de configuration lorsque vous créez un environnement, y compris les options que vous définissez

explicitement, et les [valeurs recommandées \(p. 660\)](#) définies par le client. Vous pouvez également définir les options de configuration dans les configurations enregistrées et les fichiers de configuration. Si la même option figure dans plusieurs emplacements, la valeur utilisée est déterminée par l'[ordre de priorité \(p. 659\)](#).

Paramètres d'option de configuration peuvent être composés au format texte et enregistrés avant la phase de création de l'environnement, appliqués au cours de la création de l'environnement à l'aide de n'importe quel client pris en charge et ajoutés, modifiés ou retirés après la création de l'environnement. Pour une description détaillée de toutes les méthodes disponibles pour l'utilisation des options de configuration à chacun de ces trois étapes, consultez les rubriques suivantes :

- [Définition d'options de configuration avant la création de l'environnement \(p. 661\)](#)
- [Définition des options de configuration lors de la création de l'environnement \(p. 665\)](#)
- [Définition des options de configuration après la création de l'environnement \(p. 671\)](#)

Pour obtenir une liste complète des espaces de noms et des options, y compris les valeurs par défaut et celles prises en charge pour chacun, veuillez consulter [Options générales pour tous les environnements \(p. 679\)](#) et [Options spécifiques à une plateforme \(p. 727\)](#).

## Precedence

Au cours de la création de l'environnement, des options de configuration sont appliquées à partir de plusieurs sources avec la priorité suivante, de la plus élevée à la plus faible :

- Paramètres appliqués directement à l'environnement – Les paramètres spécifiés pendant une opération de création ou de mise à jour d'environnement sur l'API Elastic Beanstalk par n'importe quel client, y compris la console Elastic Beanstalk, l'interface de ligne de commande (CLI) EB, l'AWS CLI et les kits SDK. La console Elastic Beanstalk et l'interface de ligne de commande (CLI) EB appliquent également des [valeurs recommandées \(p. 660\)](#) pour certaines options qui correspondent à ce niveau, sauf en cas de remplacement.
- Configurations sauvegardées – Les paramètres pour toutes les options qui ne sont pas appliquées directement à l'environnement sont chargés à partir d'une configuration enregistrée, si spécifié.
- Fichiers de configuration (.ebextensions) – Les paramètres pour toutes les options qui ne sont pas appliquées directement à l'environnement et ne sont pas non plus précisées dans une configuration enregistrée, sont chargés à partir de fichiers de configuration dans le dossier .ebextensions à la racine du bundle de fichiers source de l'application.

Les fichiers de configuration sont exécutés dans l'ordre alphabétique. Par exemple, `.ebextensions/01run.config` est exécuté avant `.ebextensions/02do.config`.

- Valeurs par défaut – Si une option de configuration a une valeur par défaut, elle ne s'applique que lorsque l'option n'est pas définie sur un des niveaux ci-dessus.

Si la même option de configuration est définie dans plusieurs emplacements, le paramètre ayant la plus haute priorité est appliqué. Lorsqu'un paramètre est appliqué à partir d'une configuration enregistrée ou de paramètres appliqués directement à l'environnement, le paramètre est stocké dans le cadre de la configuration de l'environnement. Ces paramètres peuvent être supprimés avec [l'AWS CLI \(p. 678\)](#) ou avec [l'interface de ligne de commande \(CLI\) EB \(p. 675\)](#).

Les paramètres dans les fichiers de configuration ne sont pas appliqués directement à l'environnement et ne peuvent pas être supprimés sans modifier les fichiers de configuration ni déployer une nouvelle version de l'application. Si un paramètre appliqué avec l'une ou l'autre des méthodes est supprimé, ce même paramètre sera chargé à partir des fichiers de configuration dans le groupe source.

Imaginons par exemple que vous définissiez le nombre minimum d'instances dans votre environnement sur 5 lors de la création de l'environnement, à l'aide de la console Elastic Beanstalk, d'une option de ligne de

commande ou d'une configuration sauvegardée. Le groupe source de votre application inclut également un fichier de configuration qui définit le nombre minimum d'instances sur 2.

Lorsque vous créez l'environnement, Elastic Beanstalk définit l'option `MinSize` dans l'espace de noms `aws:autoscaling:asg` sur 5. Si vous supprimez ensuite l'option de la configuration de l'environnement, la valeur dans le fichier de configuration est chargée, et le nombre minimum d'instances est défini sur 2. Si vous supprimez ensuite le fichier de configuration du bundle de fichiers source et que vous redéployez, Elastic Beanstalk utilise 1 comme paramètre par défaut.

## Valeurs recommandées

L'interface de ligne de commande Elastic Beanstalk (EB CLI) et la console Elastic Beanstalk fournissent des valeurs recommandées pour certaines options de configuration. Ces valeurs peuvent être différentes des valeurs par défaut et sont définies au niveau de l'API lors de la création de votre environnement. Les valeurs recommandées permettent à Elastic Beanstalk d'améliorer la configuration de l'environnement par défaut sans procéder à des modifications précédentes incompatibles au niveau de l'API.

Par exemple, l'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk définissent l'option de configuration pour le type d'instance EC2 (`InstanceType` dans l'espace de noms `aws:autoscaling:launchconfiguration`). Chaque client fournit une manière différente de remplacer le paramètre par défaut. Dans la console, vous pouvez choisir un autre type d'instance à partir d'un menu déroulant sur la page Détails de configuration de l'assistant Créez un nouvel environnement. Avec l'interface de ligne de commande (CLI) EB, vous pouvez utiliser le paramètre `--instance_type` pour [eb create \(p. 1078\)](#).

Etant donné que les valeurs recommandées sont définies au niveau de l'API, elles remplaceront les valeurs pour les mêmes options que vous définissez dans les fichiers de configuration ou les configurations enregistrées. Les options suivantes sont définies :

### Console Elastic Beanstalk

- Espace de nom : `aws:autoscaling:launchconfiguration`  
Noms d'options : `IamInstanceProfile`, `EC2KeyName`, `InstanceType`
- Espace de nom : `aws:autoscaling:updatepolicy:rollingupdate`  
Noms d'options : `RollingUpdateType` et `RollingUpdateEnabled`
- Espace de nom : `aws:elasticbeanstalk:application`  
Nom d'option: `Application Healthcheck URL`
- Espace de nom : `aws:elasticbeanstalk:command`  
Nom d'option : `DeploymentPolicy`, `BatchSize` et `BatchSizeType`
- Espace de nom : `aws:elasticbeanstalk:environment`  
Nom d'option: `ServiceRole`
- Espace de nom : `aws:elasticbeanstalk:healthreporting:system`  
Nom d'option : `SystemType` et `HealthCheckSuccessThreshold`
- Espace de nom : `aws:elasticbeanstalk:sns:topics`  
Nom d'option: `Notification Endpoint`
- Espace de nom : `aws:elasticbeanstalk:sqsd`  
Nom d'option: `HttpConnections`

- Espace de nom : `aws:elb:loadbalancer`  
Nom d'option: `CrossZone`
- Espace de nom : `aws:elb:policies`  
Noms d'options : `ConnectionDrainingTimeout` et `ConnectionDrainingEnabled`

#### INTERFACE DE LIGNE DE COMMANDE (CLI) EB

- Espace de nom : `aws:autoscaling:launchconfiguration`  
Noms d'options : `IamInstanceProfile`, `InstanceType`
- Espace de nom : `aws:autoscaling:updatepolicy:rollingupdate`  
Noms d'options : `RollingUpdateType` et `RollingUpdateEnabled`
- Espace de nom : `aws:elasticbeanstalk:command`  
Nom d'option : `BatchSize` et `BatchSizeType`
- Espace de nom : `aws:elasticbeanstalk:environment`  
Nom d'option: `ServiceRole`
- Espace de nom : `aws:elasticbeanstalk:healthreporting:system`  
Nom d'option: `SystemType`
- Espace de nom : `aws:elb:loadbalancer`  
Nom d'option: `CrossZone`
- Espace de nom : `aws:elb:policies`  
Noms d'options : `ConnectionDrainingEnabled`

## Définition d'options de configuration avant la création de l'environnement

AWS Elastic Beanstalk prend en charge un grand nombre d'[options de configuration \(p. 658\)](#) qui vous permettent de modifier les paramètres qui sont appliqués aux ressources présentes dans votre environnement. Plusieurs de ces options ont des valeurs par défaut qui peuvent être ignorées pour personnaliser votre environnement. D'autres options peuvent être configurées pour activer des fonctionnalités supplémentaires.

Elastic Beanstalk prend en charge deux méthodes d'enregistrement des paramètres des options de configuration. Les fichiers de configuration au format YAML ou JSON peuvent être inclus dans le code source de votre application dans un répertoire nommé `.ebextensions` et déployés dans le cadre du bundle de fichiers source de votre application. Vous créez et gérez des fichiers de configuration localement.

Les configurations enregistrées sont des modèles que vous créez à partir d'un environnement en cours d'exécution ou du fichier d'options JSON et que vous stockez dans Elastic Beanstalk. Les configurations existantes enregistrées peuvent être étendues pour créer une nouvelle configuration.

### Note

Les paramètres définis dans les fichiers de configuration et les configurations enregistrées ont une priorité moindre que les paramètres configurés pendant ou après la création de l'environnement,

y compris les valeurs recommandées appliquées par la console Elastic Beanstalk et l'[interface de ligne de commande \(CLI\) EB. \(p. 1027\)](#). Consultez [Préférence \(p. 659\)](#) pour plus de détails.

Les options peuvent aussi être spécifiées dans un document JSON et transmises directement à Elastic Beanstalk lorsque vous créez ou mettez à jour un environnement avec l'interface de ligne de commande (CLI) EB ou l'AWS CLI. Les options fournies directement à Elastic Beanstalk de cette manière remplacent toutes les autres méthodes.

Pour obtenir une liste complète des options disponibles, veuillez consulter [Options de configuration \(p. 658\)](#).

#### Méthodes

- [Fichiers de configuration \(.ebextensions\) \(p. 662\)](#)
- [Configurations enregistrées \(p. 663\)](#)
- [Document JSON \(p. 665\)](#)
- [Configuration de l'interface de ligne de commande \(CLI\) EB \(p. 665\)](#)

## Fichiers de configuration (.ebextensions)

Utilisez `.ebextensions` pour configurer les options qui sont requises pour que votre application fonctionne et fournissez des valeurs par défaut pour d'autres options qui peuvent être remplacées à un niveau supérieur de [priorité \(p. 659\)](#). Les options spécifiées dans `.ebextensions` ont le niveau de priorité le plus bas et sont remplacées par des paramètres de n'importe quel autre niveau.

Pour utiliser des fichiers de configuration, créez un dossier nommé `.ebextensions` au niveau supérieur du code source de votre projet. Ajoutez un fichier avec l'extension `.config` et spécifiez les options de la façon suivante :

```
option_settings:  
  - namespace: namespace  
    option_name: option name  
    value: option value  
  - namespace: namespace  
    option_name: option name  
    value: option value
```

Par exemple, le fichier de configuration suivant définit l'URL de vérification de l'état de l'application sur `/health`:

`healthcheckurl.config`

```
option_settings:  
  - namespace: aws:elasticbeanstalk:application  
    option_name: Application Healthcheck URL  
    value: /health
```

Dans JSON :

```
{  
  "option_settings" :  
  [  
    {  
      "namespace" : "aws:elasticbeanstalk:application",  
      "option_name" : "Application Healthcheck URL",  
      "value" : "/health"  
    }  
  ]  
}
```

```
    ]  
}
```

Cela configure l'équilibrer de charge Elastic Load Balancing dans votre environnement Elastic Beanstalk pour effectuer une requête HTTP vers le chemin d'accès /health à chaque instance EC2 afin de déterminer si elle est saine ou non.

#### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Incluez le répertoire .ebextensions dans le [bundle de fichiers source de l'application \(p. 415\)](#) et déployez-le dans un environnement Elastic Beanstalk nouveau ou existant.

Les fichiers de configuration prennent en charge plusieurs sections en plus des option\_settings pour personnaliser le logiciel et les fichiers qui s'exécutent sur les serveurs dans votre environnement. Pour plus d'informations, consultez [.Ebextensions \(p. 737\)](#).

## Configurations enregistrées

Créez une configuration enregistrée pour enregistrer des paramètres que vous avez appliqués à un environnement existant pendant ou après la création de l'environnement à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande (CLI) EB ou de l'AWS CLI. Les configurations enregistrées appartiennent à une application et peuvent être appliquées aux environnements nouveaux ou existants pour cette application.

#### Clients

- [Console Elastic Beanstalk \(p. 663\)](#)
- [INTERFACE DE LIGNE DE COMMANDE \(CLI\) EB \(p. 664\)](#)
- [AWS CLI \(p. 664\)](#)

## Console Elastic Beanstalk

Pour créer une configuration enregistrée (console Elastic Beanstalk)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment action (Actions d'environnement), puis Save configuration (Enregistrer la configuration).
4. Utilisez la boîte de dialogue à l'écran pour terminer l'action.

Les configurations enregistrées sont stockées dans le compartiment S3 Elastic Beanstalk, dans un dossier nommé en fonction de votre application. Par exemple, les configurations pour une application nommée my-app dans la région us-west-2 pour le compte n° 123456789012 sont disponibles à l'emplacement s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app.

## INTERFACE DE LIGNE DE COMMANDE (CLI) EB

L'interface de ligne de commande (CLI) EB (p. 1027) fournit également des sous-commandes permettant d'interagir avec des configurations enregistrées sous `eb config` (p. 1071) :

Pour créer une configuration enregistrée (CLI EB)

1. Enregistrez la configuration actuelle de l'environnement attaché :

```
~/project$ eb config save --cfg my-app-v1
```

L'interface de ligne de commande (CLI) EB enregistre la configuration dans `~/project/.elasticbeanstalk/saved_configs/my-app-v1.cfg.yml`

2. Modifiez la configuration sauvegardée localement si nécessaire.
3. Chargez la configuration enregistrée dans S3 :

```
~/project$ eb config put my-app-v1
```

## AWS CLI

Créez une configuration enregistrée à partir d'un environnement en cours d'exécution avec `aws elasticbeanstalk create-configuration-template`

Pour créer une configuration enregistrée (AWS CLI)

1. Identifiez l'ID d'environnement de votre environnement Elastic Beanstalk avec `describe-environments`:

```
$ aws elasticbeanstalk describe-environments --environment-name my-env
{
    "Environments": [
        {
            "ApplicationName": "my-env",
            "EnvironmentName": "my-env",
            "VersionLabel": "89df",
            "Status": "Ready",
            "Description": "Environment created from the EB CLI using \"eb create\"",
            "EnvironmentId": "e-vcghmm2zkwk",
            "EndpointURL": "awseb-e-v-AWSEBLoa-1JUM8159RA11M-43V6ZI1194.us-west-2.elb.amazonaws.com",
            "SolutionStackName": "64bit Amazon Linux 2015.03 v2.0.2 running Multi-container Docker 1.7.1 (Generic)",
            "CNAME": "my-env-nfptuqaper.elasticbeanstalk.com",
            "Health": "Green",
            "AbortableOperationInProgress": false,
            "Tier": {
                "Version": " ",
                "Type": "Standard",
                "Name": "WebServer"
            },
            "HealthStatus": "Ok",
            "DateUpdated": "2015-10-01T00:24:04.045Z",
            "DateCreated": "2015-09-30T23:27:55.768Z"
        }
    ]
}
```

2. Enregistrez la configuration actuelle de l'environnement avec `create-configuration-template`:

```
$ aws elasticbeanstalk create-configuration-template --environment-id e-vcghmm2zwm --  
application-name my-app --template-name v1
```

Elastic Beanstalk enregistre la configuration dans votre compartiment Elastic Beanstalk dans Amazon S3.

## Document JSON

Si vous utilisez l'AWS CLI pour créer et mettre à jour des environnements, vous pouvez également fournir des options de configuration au format JSON. Une bibliothèque de fichiers de configuration dans JSON est utile si vous utilisez l'AWS CLI pour créer et gérer des environnements.

Par exemple, les ensembles de documents JSON suivants définissent l'URL d'intégrité de l'application sur /health:

~/ebconfigs/healthcheckurl.json

```
[  
  {  
    "Namespace": "aws:elasticbeanstalk:application",  
    "OptionName": "Application Healthcheck URL",  
    "Value": "/health"  
  }  
]
```

## Configuration de l'interface de ligne de commande (CLI) EB

En plus de prendre en charge les configurations enregistrées et la configuration de l'environnement direct avec les commandes eb config, l'interface de ligne de commande (CLI) EB a un fichier de configuration avec une option nommée default\_ec2\_keyname que vous pouvez utiliser pour spécifier une paire de clés Amazon EC2 pour l'accès SSH aux instances dans votre environnement. L'interface de ligne de commande (CLI) EB utilise cette option pour définir l'option de configuration EC2KeyName dans l'espace de noms aws:autoscaling:launchconfiguration.

~/workspace/my-app/.elasticbeanstalk/config.yml

```
branch-defaults:  
  master:  
    environment: my-env  
  develop:  
    environment: my-env-dev  
deploy:  
  artifact: ROOT.war  
global:  
  application_name: my-app  
  default_ec2_keyname: my-keypair  
  default_platform: Tomcat 8 Java 8  
  default_region: us-west-2  
  profile: null  
  sc: git
```

## Définition des options de configuration lors de la création de l'environnement

Lorsque vous créez un environnement AWS Elastic Beanstalk à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande (CLI) EB, de l'AWS CLI, d'un kit SDK ou de l'API Elastic Beanstalk, vous

pouvez fournir des valeurs pour les options de configuration afin de personnaliser votre environnement et les ressources AWS qui y sont lancées.

Pour toute autre action qu'une modification ponctuelle de la configuration, vous pouvez [stocker les fichiers de configuration \(p. 661\)](#) localement, dans votre bundle de fichiers source ou dans Amazon S3.

Cette rubrique inclut les procédures relatives à toutes les méthodes de définition des options de configuration lors de la création de l'environnement.

#### Clients

- [Dans la console Elastic Beanstalk. \(p. 666\)](#)
- [Utilisation de l'interface de ligne de commande \(CLI\) EB \(p. 668\)](#)
- [À partir d AWS CLI \(p. 669\)](#)

## Dans la console Elastic Beanstalk.

Lorsque vous créez un environnement Elastic Beanstalk dans la console Elastic Beanstalk, vous pouvez fournir des options de configuration à l'aide de fichiers de configuration, de configurations enregistrées et de formulaires dans l'assistant Create New Environment (Créer un nouvel environnement).

#### Méthodes

- [Utilisation des fichiers de configuration \(.ebextensions\) \(p. 666\)](#)
- [Utilisation d'une configuration enregistrée \(p. 666\)](#)
- [Utilisation de l'assistant de création d'un environnement \(p. 667\)](#)

### Utilisation des fichiers de configuration (.ebextensions)

Incluez les fichiers `.config` dans le [bundle source de votre application \(p. 415\)](#) dans un dossier nommé `.ebextensions`.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter [.Ebextensions \(p. 737\)](#).

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
|   |   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Téléchargez le bundle de fichiers source vers Elastic Beanstalk normalement, lors de la [création de l'environnement \(p. 439\)](#).

La console Elastic Beanstalk applique des [valeurs recommandées \(p. 660\)](#) pour certaines options de configuration et inclut des champs de formulaire pour les autres. Les options configurées par la console Elastic Beanstalk sont appliquées directement à l'environnement et remplacent les paramètres dans les fichiers de configuration.

### Utilisation d'une configuration enregistrée

Lorsque vous créez un environnement à l'aide de la console Elastic Beanstalk, l'une des premières étapes consiste à choisir une configuration. Il peut s'agir d'une [configuration prédéfinie \(p. 31\)](#), qui correspond généralement à la dernière version d'une plateforme telle que PHP ou Tomcat, ou d'une configuration enregistrée.

Pour appliquer une configuration enregistrée lors de la création de l'environnement (console Elastic Beanstalk)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Saved configurations (Configurations enregistrées).
4. Sélectionnez la configuration enregistrée que vous souhaitez appliquer, puis choisissez Launch environment (Lancer l'environnement).
5. Suivez les étapes de l'assistant pour créer votre environnement.

Les configurations enregistrées sont spécifiques aux applications. Pour de plus amples informations sur la création de configurations enregistrées, veuillez consulter [Configurations enregistrées \(p. 663\)](#).

## Utilisation de l'assistant de création d'un environnement

La plupart des options de configuration standards sont présentées dans les pages Configurer plus d'options de l'assistant [Création d'un environnement \(p. 442\)](#). Si vous créez une base de données Amazon RDS ou configurez un VPC pour votre environnement, des options de configuration supplémentaires vous sont proposées pour ces ressources.

Pour définir les options de configuration lors de la création de l'environnement (console Elastic Beanstalk)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le volet de navigation, choisissez Applications.
3. Choisissez ou [créez \(p. 406\)](#) une application.
4. Choisissez Actions, puis Create environment (Créer un environnement).
5. Suivez les étapes de l'assistant, puis choisissez Configurer plus d'options.
6. Choisissez l'un des configuration presets (préréglages de la configuration), puis choisissez Edit (Modifier) dans une ou plusieurs catégories de configuration pour modifier un groupe d'options de configuration associées.
7. Une fois que vous avez sélectionné toutes les options souhaitées, choisissez Créer un environnement.

Toutes les options que vous définissez dans l'assistant de création d'un environnement sont directement définies dans l'environnement. Elles remplacent les paramètres d'option figurant dans les configurations enregistrées ou dans les fichiers de configuration (.ebextensions) que vous appliquez. Vous pouvez supprimer des paramètres après la création de l'environnement, via [l'interface de ligne de commande \(CLI\) EB \(p. 674\)](#) ou [l'AWS CLI \(p. 676\)](#) afin d'autoriser l'affichage des paramètres dans les configurations enregistrées ou les fichiers de configuration.

Pour plus d'informations sur l'assistant de création d'environnement, reportez-vous à la section [Assistant de création d'un environnement \(p. 442\)](#).

## Utilisation de l'interface de ligne de commande (CLI) EB

### Méthodes

- [Utilisation des fichiers de configuration \(.ebextensions\) \(p. 668\)](#)
- [Utilisations de configurations enregistrées \(p. 668\)](#)
- [Utilisation d'options de ligne de commande \(p. 669\)](#)

### Utilisation des fichiers de configuration (.ebextensions)

Incluez les fichiers `.config` dans votre dossier de projet sous `.ebextensions` afin de les déployer avec votre code d'application.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter [.Ebextensions \(p. 737\)](#).

```
~/workspace/my-app/
|-- .ebextensions
|   |-- environmentvariables.config
|   |   `-- healthcheckurl.config
|   |-- elasticbeanstalk
|   |   `-- config.yml
|   |-- index.php
`-- styles.css
```

Créez votre environnement et déployez votre code source dans ce dernier à l'aide de la commande `eb create`.

```
~/workspace/my-app$ eb create my-env
```

### Utilisations de configurations enregistrées

Pour appliquer une configuration enregistrée lorsque vous créez un environnement via [eb create \(p. 1078\)](#), utilisez l'option `--cfg`.

```
~/workspace/my-app$ eb create --cfg savedconfig
```

Vous pouvez stocker la configuration enregistrée dans votre dossier de projet ou dans votre emplacement de stockage Elastic Beanstalk sur Amazon S3. Dans l'exemple ci-avant, l'interface de ligne de commande (CLI) EB commence par rechercher un fichier de configuration enregistré nommé `savedconfig.cfg.yml` dans le dossier `.elasticbeanstalk/saved_configs/`. N'incluez pas les extensions des noms de fichier (`.cfg.yml`) lorsque vous appliquez une configuration enregistrée avec `--cfg`.

```
~/workspace/my-app/
|-- .ebextensions
|   `-- healthcheckurl.config
|-- .elasticbeanstalk
|   |-- saved_configs
|   |   `-- savedconfig.cfg.yml
|   |-- config.yml
|-- index.php
`-- styles.css
```

Si l'interface de ligne de commande (CLI) EB ne trouve pas la configuration localement, elle cherche dans l'emplacement de stockage Elastic Beanstalk dans Amazon S3. Pour de plus amples informations

sur la création, la modification et le téléchargement des configurations enregistrées, veuillez consulter [Configurations enregistrées \(p. 663\)](#).

## Utilisation d'options de ligne de commande

La commande eb create de l'interface de ligne de commande (CLI) EB comporte plusieurs [options \(p. 1079\)](#), qui vous permettent de définir des options de configuration lors de la création de l'environnement. Vous pouvez utiliser ces options pour ajouter une base de données RDS à votre environnement, configurer un VPC ou remplacer des [valeurs recommandées \(p. 660\)](#).

Par exemple, l'interface de ligne de commande (CLI) EB utilise le type d'instance t2.micro par défaut. Pour choisir un autre type d'instance, utilisez l'option --instance\_type.

```
$ eb create my-env --instance_type t2.medium
```

Pour créer une instance de base de données Amazon RDS et l'associer à votre environnement, utilisez les options --database :

```
$ eb create --database.engine postgres --database.username dbuser
```

Si vous n'indiquez pas le nom de l'environnement, le mot de passe de la base de données ou tout autre paramètre requis pour créer votre environnement, l'interface de ligne de commande (CLI) EB vous invite à saisir ces données.

Consultez [eb create \(p. 1078\)](#) pour obtenir la liste complète des options disponibles et des exemples d'utilisation.

## À partir d AWS CLI

Si vous utilisez la commande create-environment pour créer un environnement Elastic Beanstalk via la AWS CLI, la AWS CLI n'applique aucune [valeur recommandée \(p. 660\)](#). Toutes les options de configuration sont définies dans les fichiers de configuration du bundle source que vous spécifiez.

### Méthodes

- [Utilisation des fichiers de configuration \(.ebextensions\) \(p. 669\)](#)
- [Utilisation d'une configuration enregistrée \(p. 670\)](#)
- [Utilisation d'options de ligne de commande \(p. 670\)](#)

## Utilisation des fichiers de configuration (.ebextensions)

Pour appliquer des fichiers de configuration à un environnement que vous créez via la AWS CLI, incluez-les dans la solution groupée de fichiers source de l'application que vous téléchargez dans Amazon S3.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter [.Ebextensions \(p. 737\)](#).

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Pour charger le bundle source d'une application et créer un environnement via l AWS CLI

1. Si vous n'avez pas encore de compartiment Elastic Beanstalk dans Amazon S3, créez-en un avec `create-storage-location`.

```
$ aws elasticbeanstalk create-storage-location
{
    "S3Bucket": "elasticbeanstalk-us-west-2-123456789012"
}
```

2. Téléchargez votre bundle de fichiers source d'application sur Amazon S3.

```
$ aws s3 cp sourcebundle.zip s3://elasticbeanstalk-us-west-2-123456789012/my-app/
sourcebundle.zip
```

3. Créez la version de l'application.

```
$ aws elasticbeanstalk create-application-version --application-name my-app --
version-label v1 --description MyAppv1 --source-bundle S3Bucket="elasticbeanstalk-us-
west-2-123456789012",S3Key="my-app/sourcebundle.zip" --auto-create-application
```

4. Création de l'environnement

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-name
my-env --version-label v1 --solution-stack-name "64bit Amazon Linux 2015.03 v2.0.0
running Tomcat 8 Java 8"
```

## Utilisation d'une configuration enregistrée

Pour appliquer une configuration enregistrée à un environnement lors de la création, utilisez le paramètre **--template-name**.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-name my-
env --template-name savedconfig --version-label v1
```

Lorsque vous spécifiez une configuration enregistrée, ne spécifiez pas un nom de pile de solutions en plus. Les configurations enregistrées spécifient déjà une pile de solutions, et Elastic Beanstalk affichera une erreur si vous essayez d'utiliser les deux options.

## Utilisation d'options de ligne de commande

Utilisez le paramètre **--option-settings** pour spécifier les options de configuration au format JSON.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-name my-
env --version-label v1 --template-name savedconfig --option-settings [
    {
        "Namespace": "aws:elasticbeanstalk:application",
        "OptionName": "Application Healthcheck URL",
        "Value": "/health"
    }
]
```

Pour charger le JSON à partir d'un fichier, utilisez le préfixe **file://**.

```
$ aws elasticbeanstalk create-environment --application-name my-app --environment-
name my-env --version-label v1 --template-name savedconfig --option-settings file://
healthcheckurl.json
```

Elastic Beanstalk applique directement les paramètres d'option que vous spécifiez via l'option **--option-settings** dans votre environnement. Si les mêmes options sont spécifiées dans une configuration enregistrée ou un fichier de configuration, **--option-settings** remplace ces valeurs.

## Définition des options de configuration après la création de l'environnement

Vous pouvez modifier les paramètres d'option dans un environnement en cours d'exécution en appliquant des configurations enregistrées, en chargeant un nouveau bundle de fichiers source à l'aide des fichiers de configuration (`.ebextensions`) ou un document JSON. L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk ont également des fonctionnalités spécifiques client pour définir et mettre à jour des options de configuration.

Lorsque vous définissez ou modifiez une option de configuration, vous pouvez déclencher une mise à jour environnement complet, selon la gravité du changement. Par exemple, des modifications apportées aux options dans [aws:autoscaling:launchconfiguration \(p. 681\)](#), comme `InstanceType`, nécessitent que les instances Amazon EC2 dans votre environnement soient réapprovisionnées. Cela déclenche une [mise à jour propagée \(p. 489\)](#). D'autres changements de configuration peuvent être appliqués sans interruption ou réapprovisionnement.

Vous pouvez supprimer les paramètres d'option à partir d'un environnement avec des commandes de l'interface de ligne de commande (CLI) EB ou de la AWS CLI. La suppression d'une option qui a été définie directement sur un environnement à un niveau d'API autorise des paramètres dans des fichiers de configuration, qui sont autrement masqués par des paramètres appliqués directement dans un environnement, à s'exposer et à prendre effet.

Vous pouvez remplacer les paramètres des configurations enregistrées et des fichiers de configuration en définissant la même option directement au niveau de l'environnement à l'aide de l'une des autres méthodes de configuration. Toutefois, le seul moyen de supprimer complètement ces paramètres consiste à appliquer une nouvelle configuration enregistrée ou un nouveau fichier de configuration. Lorsqu'une option n'est pas définie dans une configuration enregistrée, dans le fichier de configuration, ou directement sur un environnement, la valeur par défaut s'applique, s'il y en a une. Consultez [Precedence \(p. 659\)](#) pour plus de détails.

### Clients

- [La console Elastic Beanstalk. \(p. 671\)](#)
- [Interface de ligne de commande \(CLI\) EB \(p. 674\)](#)
- [. AWS CLI \(p. 676\)](#)

## La console Elastic Beanstalk.

Vous pouvez mettre à jour des paramètres d'option de configuration dans la console Elastic Beanstalk en déployant un bundle de fichiers source d'application qui contient des fichiers de configuration, en appliquant une configuration enregistrée ou en modifiant l'environnement directement avec la page Configuration dans la console de gestion de l'environnement.

### Méthodes

- [Utilisation des fichiers de configuration \(.ebextensions\) \(p. 671\)](#)
- [Utilisation d'une configuration enregistrée \(p. 672\)](#)
- [Utilisation de la console Elastic Beanstalk \(p. 673\)](#)

### Utilisation des fichiers de configuration (.ebextensions)

Mettez à jour les fichiers de configuration dans votre répertoire source, créez un bundle de fichiers source et déployez la nouvelle version dans votre environnement Elastic Beanstalk pour appliquer les modifications.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter [.Ebextensions \(p. 737\)](#).

#### Pour déployer un groupe source

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

##### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Sur la page de présentation de l'environnement, choisissez Upload and deploy (Charger et déployer).
4. Utilisez la boîte de dialogue à l'écran pour charger le bundle source.
5. Choisissez Deploy (Déployer).
6. Lorsque le déploiement est terminé, vous pouvez sélectionner l'URL de site pour ouvrir votre site web dans un nouvel onglet.

Les modifications apportées aux fichiers de configuration ne remplaceront pas les paramètres d'options dans les configurations enregistrées ou les paramètres appliqués directement à l'environnement au niveau de l'API. Pour plus d'informations, consultez [Precedence \(p. 659\)](#).

#### Utilisation d'une configuration enregistrée

Appliquez une configuration enregistrée à un environnement en cours d'exécution pour appliquer des paramètres d'option qu'il définit.

Pour appliquer une configuration enregistrée à un environnement en cours d'exécution (console Elastic Beanstalk)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

##### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Saved configurations (Configurations enregistrées).
4. Sélectionnez la configuration enregistrée à appliquer, puis choisissez Load (Charger).
5. Sélectionnez un environnement, puis Charge.

Les paramètres définis dans une configuration enregistrée remplacent les paramètres dans les fichiers de configuration et sont remplacés par les paramètres configurés à l'aide de la console de gestion d'environnement.

Pour de plus amples informations sur la création de configurations enregistrées, veuillez consulter [Configurations enregistrées \(p. 663\)](#).

## Utilisation de la console Elastic Beanstalk

La console Elastic Beanstalk présente beaucoup d'options de configuration sur la page Configuration (Configuration) de chaque environnement.

Pour modifier les options de configuration dans un environnement en cours d'exécution (console Elastic Beanstalk)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Recherchez la page de configuration que vous souhaitez modifier :
  - Si vous voyez l'option qui vous intéresse, ou que vous savez à quelle catégorie de configuration elle appartient, choisissez Edit (Modifier) dans sa catégorie de configuration.
  - Pour rechercher une option, activez la Table View (Vue Table), puis saisissez vos termes de recherche dans la zone de recherche. À mesure que vous saisissez du texte, la liste se réduit et affiche uniquement les options qui correspondent à vos termes de recherche.

Lorsque vous voyez l'option que vous recherchez, choisissez Edit (Modifier) dans la catégorie de configuration qui la contient.

The screenshot shows the 'Configuration overview' page with a search bar containing 'log'. Below the search bar, there's a table with two columns: 'Category' and 'Options'. The 'Category' column lists 'Software' and 'Monitoring'. The 'Options' column contains configuration details for each. For 'Software', it says 'Lifecycle: Keep logs after terminating environment', 'Rotate logs: disabled', and 'Log streaming: disabled'. For 'Monitoring', it says 'Log group: logStream:group=/aws/elasticbeanstalk/Node-env/environment-health target=\_blank>/aws/elasticbeanstalk/Node-env/environment-health.log'.

Category	Options
Software	Lifecycle: Keep logs after terminating environment Rotate logs: disabled Log streaming: disabled
Monitoring	Log group: logStream:group=/aws/elasticbeanstalk/Node-env/environment-health target=_blank>/aws/elasticbeanstalk/Node-env/environment-health.log

5. Modifiez les paramètres, puis choisissez Enregistrer.
6. Si vous souhaitez modifier d'autres catégories de configuration, répétez les deux étapes précédentes.
7. Choisissez Apply.

Les modifications apportées aux options de configuration dans la console de gestion de l'environnement sont directement appliquées à l'environnement. Ces modifications remplacent les paramètres pour les mêmes options dans les fichiers de configuration ou dans les configurations enregistrées. Pour plus d'informations, consultez [Precedence \(p. 659\)](#) (Priorité).

Pour de plus amples informations sur la modification des options de configuration au niveau d'un environnement en cours d'exécution à l'aide de la console Elastic Beanstalk, veuillez consulter les rubriques sous [Configuration d'environnements Elastic Beanstalk \(p. 532\)](#).

## Interface de ligne de commande (CLI) EB

Vous pouvez mettre à jour les paramètres d'option de configuration avec l'interface de ligne de commande (CLI) EB en déployant le code source qui contient des fichiers de configuration, en appliquant des paramètres à partir d'une configuration enregistrée ou en modifiant la configuration de l'environnement directement avec la commande eb config.

### Méthodes

- [Utilisation des fichiers de configuration \(.ebextensions\) \(p. 674\)](#)
- [Utilisation d'une configuration enregistrée \(p. 674\)](#)
- [Utiliser eb config \(p. 674\)](#)
- [Utiliser eb setenv \(p. 676\)](#)

### Utilisation des fichiers de configuration (.ebextensions)

Incluez les fichiers .config dans votre dossier de projet sous .ebextensions afin de les déployer avec votre code d'application.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter [.Ebextensions \(p. 737\)](#).

```
~/workspace/my-app/
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- .elasticbeanstalk
|   '-- config.yml
|-- index.php
`-- styles.css
```

Déployez votre code source avec eb deploy.

```
~/workspace/my-app$ eb deploy
```

### Utilisation d'une configuration enregistrée

Vous pouvez utiliser la commande eb config pour appliquer une configuration enregistrée à un environnement en cours d'exécution. Utilisez l'option --cfg avec le nom de la configuration enregistrée pour appliquer ses paramètres à votre environnement.

```
$ eb config --cfg v1
```

Dans cet exemple, v1 est le nom d'un [fichier de configuration enregistré précédemment créé \(p. 663\)](#).

Les paramètres appliqués à un environnement avec cette commande remplacent les paramètres qui ont été appliqués au cours de la création de l'environnement et les paramètres définis dans les fichiers de configuration dans le groupe source de votre application.

### Utiliser eb config

La commande eb config de l'interface de ligne de commande (CLI) EB vous permet de définir et de supprimer des paramètres d'option directement dans un environnement à l'aide d'un éditeur de texte.

Lorsque vous exécutez eb config, l'interface de ligne de commande (CLI) EB affiche les paramètres appliqués à votre environnement depuis toutes les sources, y compris les fichiers de configuration, les configurations enregistrées, les valeurs recommandées, le jeu d'options directement sur l'environnement et les valeurs par défaut d'API.

#### Note

eb config n'affiche pas les propriétés de l'environnement. Pour définir les propriétés de l'environnement que vous pouvez lire à partir de votre application, utilisez [eb setenv \(p. 676\)](#).

L'exemple suivant montre les paramètres appliqués dans l'espace de noms `aws:autoscaling:launchconfiguration`. Ces paramètres sont les suivants :

- Deux valeurs recommandées, pour `IamInstanceProfile` et `InstanceType`, appliquées par l'interface de ligne de commande (CLI) EB lors de la création de l'environnement.
- L'option `EC2KeyName`, définie directement sur l'environnement lors de la création selon la configuration du référentiel.
- Les valeurs d'API par défaut pour les autres options.

```
ApplicationName: tomcat
DateUpdated: 2015-09-30 22:51:07+00:00
EnvironmentName: tomcat
SolutionStackName: 64bit Amazon Linux 2015.03 v2.0.1 running Tomcat 8 Java 8
settings:
...
aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: null
    EC2KeyName: my-key
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    ImageId: ami-1f316660
    InstanceType: t2.micro
...
```

Pour définir ou modifier les options de configuration avec eb config

1. Exécutez eb config pour afficher la configuration de votre environnement.

```
~/workspace/my-app/$ eb config
```

2. Modifiez toutes valeurs de paramètre à l'aide de l'éditeur de texte par défaut.

```
aws:autoscaling:launchconfiguration:
    BlockDeviceMappings: null
    EC2KeyName: my-key
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    ImageId: ami-1f316660
    InstanceType: t2.medium
```

3. Enregistrez le fichier de configuration temporaire et quittez.
4. L'interface de ligne de commande (CLI) EB met à jour la configuration de votre environnement.

Définir des options de configuration avec eb config remplace les paramètres de toutes les autres sources.

Vous pouvez également supprimer des options à partir de votre environnement avec eb config.

Pour supprimer des options de configuration (CLI EB)

1. Exécutez eb config pour afficher la configuration de votre environnement.

```
~/workspace/my-app/$ eb config
```

2. Remplacez toute valeur indiquée par la chaîne `null`. Vous pouvez aussi supprimer la ligne complète contenant l'option que vous souhaitez supprimer.

```
aws:autoscaling:launchconfiguration:  
  BlockDeviceMappings: null  
  EC2KeyName: my-key  
  IamInstanceProfile: aws-elasticbeanstalk-ec2-role  
  ImageId: ami-1f316660  
  InstanceType: null
```

3. Enregistrez le fichier de configuration temporaire et quittez.
4. L'interface de ligne de commande (CLI) EB met à jour la configuration de votre environnement.

Supprimer des options de votre environnement avec `eb config` permet aux paramètres pour les mêmes options de s'exposer depuis les fichiers de configuration dans le groupe source de votre application. Pour plus d'informations, consultez [Precedence \(p. 659\)](#).

## Utiliser eb setenv

Pour définir des propriétés de l'environnement avec l'interface de ligne de commande (CLI) EB, utilisez `eb setenv`.

```
~/workspace/my-app/$ eb setenv ENVVAR=TEST  
INFO: Environment update is starting.  
INFO: Updating environment my-env's configuration settings.  
INFO: Environment health has transitioned from Ok to Info. Command is executing on all instances.  
INFO: Successfully deployed new configuration to environment.
```

Cette commande définit les propriétés de l'environnement dans l'[aws:elasticbeanstalk:application:environment](#) espace de noms (p. 697). Les propriétés d'environnement définies avec `eb setenv` sont disponibles pour votre application après un court processus de mise à jour.

Affichez les propriétés de l'environnement définies sur votre environnement avec `eb printenv`.

```
~/workspace/my-app/$ eb printenv  
Environment Variables:  
ENVVAR = TEST
```

## . AWS CLI

Vous pouvez mettre à jour les paramètres d'option de configuration avec la AWS CLI en déployant une solution groupée source qui contient des fichiers de configuration, en appliquant une configuration enregistrée stockée à distance ou en modifiant l'environnement directement avec la commande `aws elasticbeanstalk update-environment`.

### Méthodes

- [Utilisation des fichiers de configuration \(.ebextensions\) \(p. 677\)](#)
- [Utilisation d'une configuration enregistrée \(p. 677\)](#)
- [Utilisation d'options de ligne de commande \(p. 678\)](#)

## Utilisation des fichiers de configuration (.ebextensions)

Pour appliquer des fichiers de configuration à un environnement en cours d'exécution avec la AWS CLI, incluez-le dans la solution groupée de fichiers source de l'application que vous téléchargez sur Amazon S3.

Pour de plus amples informations sur les fichiers de configuration, veuillez consulter [.Ebextensions \(p. 737\)](#).

```
~/workspace/my-app-v1.zip
|-- .ebextensions
|   |-- environmentvariables.config
|   `-- healthcheckurl.config
|-- index.php
`-- styles.css
```

Pour télécharger un bundle source d'application et l'appliquer à un environnement en cours d'exécution (AWS CLI)

1. Si vous n'avez pas encore de compartiment Elastic Beanstalk dans Amazon S3, créez-en un avec `create-storage-location`:

```
$ aws elasticbeanstalk create-storage-location
{
    "S3Bucket": "elasticbeanstalk-us-west-2-123456789012"
}
```

2. Téléchargez votre bundle de fichiers source d'application sur Amazon S3.

```
$ aws s3 cp sourcebundlev2.zip s3://elasticbeanstalk-us-west-2-123456789012/my-app/sourcebundlev2.zip
```

3. Créez la version de l'application.

```
$ aws elasticbeanstalk create-application-version --application-name my-app --version-label v2 --description MyAppv2 --source-bundle S3Bucket="elasticbeanstalk-us-west-2-123456789012",S3Key="my-app/sourcebundlev2.zip"
```

4. Mettez à jour l'environnement.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --version-label v2
```

## Utilisation d'une configuration enregistrée

Vous pouvez appliquer une configuration enregistrée dans un environnement en cours d'exécution avec l'option `--template-name` sur la commande `aws elasticbeanstalk update-environment`.

La configuration enregistrée doit être dans votre compartiment Elastic Beanstalk dans un chemin d'accès nommé en fonction de votre application sous `resources/templates`. Par exemple, le modèle v1 de l'application `my-app` dans la région USA Ouest (Oregon) (`us-west-2`) pour le compte n° 123456789012 est disponible à l'emplacement `s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app/v1`.

Pour appliquer une configuration enregistrée à un environnement en cours d'exécution (AWS CLI)

- Spécifiez la configuration enregistrée dans un appel `update-environment` avec l'option `--template-name`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --template-name v1
```

Elastic Beanstalk place les configurations enregistrées dans cet emplacement lorsque vous les créez avec `aws elasticbeanstalk create-configuration-template`. Vous pouvez également modifier les configurations enregistrées localement et les mettre à cet emplacement vous-même.

## Utilisation d'options de ligne de commande

Pour modifier des options de configuration avec un document JSON (AWS CLI)

1. Définissez vos paramètres d'options au format JSON dans un fichier local.
2. Exécutez `update-environment` avec l'option `--option-settings`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --option-settings file://~/ebconfigs/as-zero.json
```

Dans cet exemple, `as-zero.json` définit des options qui configurent l'environnement avec un minimum et un maximum de zéro instance. Cela interrompt les instances dans l'environnement sans mise hors service de ce dernier.

`~/ebconfigs/as-zero.json`

```
[  
  {  
    "Namespace": "aws:autoscaling:asg",  
    "OptionName": "MinSize",  
    "Value": "0"  
  },  
  {  
    "Namespace": "aws:autoscaling:asg",  
    "OptionName": "MaxSize",  
    "Value": "0"  
  },  
  {  
    "Namespace": "aws:autoscaling:updatepolicy:rollingupdate",  
    "OptionName": "RollingUpdateEnabled",  
    "Value": "false"  
  }  
]
```

### Note

Définir des options de configuration avec `update-environment` remplace les paramètres de toutes les autres sources.

Vous pouvez également supprimer des options à partir de votre environnement avec `update-environment`.

Pour supprimer des options de configuration (AWS CLI)

- Exécutez la commande `update-environment` avec l'option `--settings-to-remove`.

```
$ aws elasticbeanstalk update-environment --environment-name my-env --options-to-remove Namespace=aws:autoscaling:launchconfiguration,OptionName=InstanceType
```

Supprimer des options de votre environnement avec `update-environment` permet aux paramètres pour les mêmes options de s'exposer depuis les fichiers de configuration dans le groupe source de votre application. Si une option n'est pas configurée à l'aide d'une de ces méthodes, la valeur par défaut de l'API s'applique, le cas échéant. Pour plus d'informations, consultez [Precedence \(p. 659\)](#).

## Options générales pour tous les environnements

### Espaces de noms

- [aws:autoscaling:asg \(p. 680\)](#)
- [aws:autoscaling:launchconfiguration \(p. 681\)](#)
- [aws:autoscaling:scheduledaction \(p. 687\)](#)
- [aws:autoscaling:trigger \(p. 688\)](#)
- [aws:autoscaling:updatepolicy:rollingupdate \(p. 690\)](#)
- [aws:ec2:instances \(p. 693\)](#)
- [aws:ec2:vpc \(p. 696\)](#)
- [aws:elasticbeanstalk:application \(p. 697\)](#)
- [aws:elasticbeanstalk:application:environment \(p. 697\)](#)
- [aws:elasticbeanstalk:cloudwatch:logs \(p. 698\)](#)
- [aws:elasticbeanstalk:cloudwatch:logs:health \(p. 698\)](#)
- [aws:elasticbeanstalk:command \(p. 699\)](#)
- [aws:elasticbeanstalk:environment \(p. 700\)](#)
- [aws:elasticbeanstalk:environment:process:default \(p. 701\)](#)
- [aws:elasticbeanstalk:environment:process:process\\_name \(p. 703\)](#)
- [aws:elasticbeanstalk:environment:proxy:staticfiles \(p. 706\)](#)
- [aws:elasticbeanstalk:healthreporting:system \(p. 706\)](#)
- [aws:elasticbeanstalk:hostmanager \(p. 707\)](#)
- [aws:elasticbeanstalk:managedactions \(p. 707\)](#)
- [aws:elasticbeanstalk:managedactions:platformupdate \(p. 708\)](#)
- [aws:elasticbeanstalk:monitoring \(p. 709\)](#)
- [aws:elasticbeanstalk:sns:topics \(p. 709\)](#)
- [aws:elasticbeanstalk:sqsd \(p. 710\)](#)
- [aws:elasticbeanstalk:trafficsplitting \(p. 712\)](#)
- [aws:elasticbeanstalk:xray \(p. 712\)](#)
- [aws:elb:healthcheck \(p. 712\)](#)
- [aws:elb:loadbalancer \(p. 713\)](#)
- [aws:elb:listener \(p. 714\)](#)
- [aws:elb:listener:listener\\_port \(p. 715\)](#)
- [aws:elb:policies \(p. 716\)](#)
- [aws:elb:policies:policy\\_name \(p. 717\)](#)
- [aws:elbv2:listener:default \(p. 718\)](#)
- [aws:elbv2:listener:listener\\_port \(p. 720\)](#)
- [aws:elbv2:listenerrule:rule\\_name \(p. 721\)](#)
- [aws:elbv2:loadbalancer \(p. 723\)](#)
- [aws:rds:dbinstance \(p. 725\)](#)

## aws:autoscaling:asg

Configurez le groupe Auto Scaling de votre environnement. Pour plus d'informations, consultez [the section called “Groupe Auto Scaling” \(p. 547\)](#).

Espace de noms : **aws:autoscaling:asg**

Name (Nom)	Description	Par défaut	Valeurs valides
Availability Zones	Les zones de disponibilité sont des emplacements distincts dans une région AWS, conçues pour être isolées des défaillances d'autres zones de disponibilité. Elles fournissent une connectivité réseau à faible latence à d'autres zones de disponibilité de la même région. Choisissez le nombre de zones de disponibilité pour vos instances.	Any	Any Any 1 Any 2 Any 3
Cooldown	Le temps de stabilisation permet d'empêcher Amazon EC2 Auto Scaling de lancer d'autres activités de mise à l'échelle avant que les effets des activités précédentes soient visibles. Un temps de stabilisation est la durée, en secondes, qui doit s'écouler entre la fin d'une activité de dimensionnement et le début d'une autre.	360	0 sur 10000
Custom Availability Zones	Définissez les zones de disponibilité pour vos instances.	Aucun	us-east-1a us-east-1b us-east-1c us-east-1d us-east-1e eu-central-1
EnableCapacityRebalancing	Rebalance peut activer la fonction de rééquilibrage de capacité pour les instances Spot dans votre groupe Auto Scaling. Pour plus d'informations, consultez <a href="#">Rééquilibrage de la capacité</a> dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.  Cette option n'est pertinente que lorsque EnableSpot a la valeur true dans l'espace de noms <a href="#">aws:ec2:instances (p. 693)</a> et quand votre groupe Auto Scaling comprend au moins une instance Spot.	false	true false
MinSize	Nombre minimal d'instances souhaitées dans votre groupe Auto Scaling.	1	1 sur 10000

Name (Nom)	Description	Par défaut	Valeurs valides
MaxSize	Nombre maximal d'instances souhaitées dans votre groupe Auto Scaling.	4	1 sur 10000

## aws:autoscaling:launchconfiguration

Configurez les instances Amazon Elastic Compute Cloud (Amazon EC2) de votre environnement.

Les instances utilisées pour votre environnement sont créées à l'aide d'un modèle de lancement Amazon EC2 ou d'une ressource de configuration de lancement de groupe Auto Scaling. Les options suivantes fonctionnent avec ces deux types de ressources.

Pour de plus amples informations, veuillez consulter [the section called “Instances Amazon EC2” \(p. 538\)](#).

Espace de noms : **aws:autoscaling:launchconfiguration**

Name (Nom)	Description	Par défaut	Valeurs valides
DisableIMDSv1	Définissez sur <code>true</code> pour désactiver Instance Metadata Service Version 1 (IMDSv1). Par défaut, les instances de votre environnement activent IMDSv1 et IMDSv2. Pour de plus amples informations, veuillez consulter <a href="#">Configuration du service des métadonnées d'instance</a> .	<code>false</code>	<code>true</code> <code>false</code>
EC2KeyName	<p>Vous pouvez utiliser une paire de clés pour vous connecter en toute sécurité à votre instance EC2.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console remplace cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>	Aucune	
IamInstanceProfile	<p>Un profil d'instance permet aux utilisateurs d'AWS Identity and Access Management (IAM) et des services AWS d'accéder aux informations d'identification de sécurité temporaires pour effectuer des appels d'API AWS. Spécifiez le nom du profil d'instance ou son ARN.</p> <p><b>Exemples :</b></p> <ul style="list-style-type: none"> <li>• <code>aws-elasticbeanstalk-ec2-role</code></li> <li>• <code>arn:aws:iam::123456789012:instance-profile/aws-elasticbeanstalk-ec2-role</code></li> </ul>	Aucune	Nom du profil d'instance ou ARN.

Name (Nom)	Description	Par défaut	Valeurs valides
	<p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console et l'interface de ligne de commande (CLI) EB remplacent cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>		
ImageId	<p>Vous pouvez remplacer la valeur Amazon Machine Image (AMI) par défaut en spécifiant votre propre ID d'AMI personnalisé.</p> <p>Exemple : <code>ami-1f316660</code></p>	Aucune	

Name (Nom)	Description	Par défaut	Valeurs valides
InstanceType	<p>Type d'instance utilisé pour exécuter votre application dans un environnement Elastic Beanstalk.</p> <p><b>Important</b></p> <p>L'option <code>InstanceType</code> est obsolète. Elle est remplacée par la nouvelle option <code>InstanceTypes</code> plus puissante dans l'espace de noms <a href="#">aws:ec2:instances (p. 693)</a>. Vous pouvez utiliser cette nouvelle option pour spécifier une liste d'un ou plusieurs types d'instance pour votre environnement. La première valeur de cette liste est équivalente à la valeur de l'option <code>InstanceType</code> incluse dans l'espace de noms <code>aws:autoscaling:launchconfiguration</code> décrit ici. Nous vous recommandons de spécifier les types d'instance en utilisant la nouvelle option. Si elle est spécifiée, la nouvelle option prévaut sur la précédente. Pour de plus amples informations, veuillez consulter <a href="#">the section called “Espace de noms aws:ec2:instances” (p. 555)</a>.</p> <p>Les types d'instance disponibles dépendent des zones de disponibilité et de la région utilisées. Si vous choisissez un sous-réseau, la zone de disponibilité qui contient ce sous-réseau détermine les types d'instance disponibles.</p> <ul style="list-style-type: none"> <li>• Elastic Beanstalk ne prend pas en charge les types d'instance Amazon EC2 Mac et Amazon EC2 Graviton (basées sur ARM) pour le moment.</li> <li>• Pour de plus amples informations sur les familles d'instance et types d'instance Amazon EC2, consultez <a href="#">Types d'instance</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux ou <a href="#">Types d'instance</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.</li> </ul>	<p>Varie selon le compte et la région.</p>	<p>Un type d'instance EC2.</p> <p>Varie selon le compte, la région et la zone de disponibilité. Vous pouvez obtenir une liste des types d'instance Amazon EC2 filtrés par ces valeurs. Pour de plus amples informations, consultez <a href="#">Types d'instance disponibles</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux ou <a href="#">Types d'instance disponibles</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.</p>

Name (Nom)	Description	Par défaut	Valeurs valides
	<ul style="list-style-type: none"> <li>Pour de plus amples informations sur les types d'instance disponibles dans les régions, consultez <a href="#">Types d'instance disponibles</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux ou <a href="#">Types d'instance disponibles</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.</li> </ul> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console et l'interface de ligne de commande (CLI) EB remplacent cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>		
MonitoringInterval	Intervalle (en minutes) auquel vous souhaitez que les métriques Amazon CloudWatch soient renvoyées.	5 minute	1 minute 5 minute
SecurityGroups	<p>Répertorie les groupes de sécurité Amazon EC2 à attribuer aux instances EC2 dans le groupe Auto Scaling afin de définir des règles de pare-feu pour les instances.</p> <p>Vous pouvez fournir une chaîne unique de valeurs séparées par des virgules contenant le nom des groupes de sécurité Amazon EC2 existants ou des références à des ressources AWS::EC2::SecurityGroup créées dans le modèle. Les noms des groupes de sécurité sont sensibles à la casse.</p> <p>Si vous utilisez <a href="#">Amazon Virtual Private Cloud</a> (Amazon VPC) avec Elastic Beanstalk afin que vos instances soient lancées dans un Virtual Private Cloud (VPC), spécifiez des ID de groupe de sécurité au lieu des noms de groupe de sécurité.</p>	elasticbeanstalk-default	

Name (Nom)	Description	Par défaut	Valeurs valides
	<p>Utilisé pour verrouiller l'accès SSH à un <b>SSHSourceRestriction</b>. Par exemple, vous pouvez verrouiller l'accès SSH aux instances EC2 afin que seul un hôte bastion puisse accéder aux instances du sous-réseau privé.</p> <p>Cette chaîne prend la forme suivante :</p> <p><i>protocol, fromPort, toPort, source_restriction</i></p> <p><b>protocole</b></p> <p>Le protocole pour la règle de trafic entrant.</p> <p><b>fromPort</b></p> <p>Le numéro de port de départ.</p> <p><b>toPort</b></p> <p>Le numéro de port fin.</p> <p><b>source_restriction</b></p> <p>Plage CIDR ou nom d'un groupe de sécurité par lequel le trafic est acheminé. Pour spécifier un groupe de sécurité à partir d'un autre compte (EC2-Classic uniquement, doit être dans la même région), incluez l'ID de compte avant le nom du groupe de sécurité. Utilisez le format suivant : <i>other_account_id/security_group_name</i>. Si vous utilisez <a href="#">Amazon Virtual Private Cloud</a> (Amazon VPC) avec Elastic Beanstalk afin que vos instances soient lancées dans un Virtual Private Cloud (VPC), spécifiez un ID de groupe de sécurité au lieu d'un nom de groupe de sécurité.</p> <p>Exemple : <code>tcp, 22, 22, 54.240.196.185/32</code></p> <p>Exemple : <code>tcp, 22, 22, my-security-group</code></p> <p>Exemple (EC2-Classic): <code>tcp, 22, 22, 123456789012/their-security-group</code></p> <p>Exemple (VPC): <code>tcp, 22, 22, sg-903004f8</code></p>	Aucune	

Name (Nom)	Description	Par défaut	Valeurs valides
BlockDeviceMappings	<p>Attachez des volumes Amazon EBS supplémentaires ou des volumes de stockage d'instance sur toutes les instances dans le groupe Auto Scaling.</p> <p>Lorsque vous mappez des volumes de stockage d'instance, vous devez mapper uniquement le nom du périphérique à un nom de volume. Toutefois, nous vous recommandons, lors du mappage de volumes Amazon EBS, de spécifier en outre certains ou tous les champs suivants (chaque champ doit être séparé par un signe deux points) :</p> <ul style="list-style-type: none"> <li>• ID d'instantané</li> <li>• taille, en Go</li> <li>• Supprimer à la suspension (true ou false)</li> <li>• type de stockage (uniquement pour gp3, gp2, standard, st1, sc1, ou io1)</li> <li>• IOPS (uniquement pour gp3 ou io1)</li> <li>• débit (uniquement pour gp3)</li> </ul> <p>L'exemple suivant attache trois volumes Amazon EBS, un volume vide gp2 de 100 Go et un instantané, un volume io1 vide de 20 Go avec 2000 IOPS provisionnées, et un volume de stockage d'instance ephemeral0. Plusieurs volumes de stockage d'instance peuvent être attachées si le type d'instance assure la prise en charge.</p> <pre>/dev/sdj=:100:true:gp2, /dev/sdh=snap-51eef269, /dev/sdi=:20:true:io1:2000, /dev/sdb=ephemeral0</pre>	Aucune	
RootVolumeType	Type de volume (magnétique, usage général SSD ou IOPS provisionnés (SSD)) à utiliser pour le volume Amazon EBS racine attaché aux instances EC2 de votre environnement.	Varie selon la plateforme.	<p>standard pour stockage magnétique.</p> <p>gp2 ou gp3 pour polyvalent (SSD).</p> <p>io1 pour IOPS provisionnées (SSD).</p>

Name (Nom)	Description	Par défaut	Valeurs valides
RootVolumeSize	<p>Capacité de stockage du volume Amazon EBS racine en Go.</p> <p>Nécessaire si vous définissez RootVolumeType sur IOPS provisionnées (SSD).</p> <p>Par exemple, "64".</p>	<p>Varie selon la plateforme pour stockage magnétique et usage général (SSD).</p> <p>Aucune pour IOPS provisionnées (SSD).</p>	<p>10 sur 16384 Go pour usage général et IOPS provisionnées (SSD).</p> <p>8 sur 1024 Go pour magnétique.</p>
RootVolumeIOPS	<p>Opérations souhaitées d'entrée/de sortie par seconde (IOPS) pour un volume racine IOPS provisionné (SSD) ou un volume racine de SSD gp3 polyvalent.</p> <p>Le rapport maximal d'IOPS selon la taille du volume est de 30 à 1. Par exemple, un volume à 3000 IOPS doit avoir une taille d'au moins 100 Go.</p>	Aucune	<p>100 à 20000 pour les volumes racine de SSD IOPS provisionnés io1.</p> <p>3000 à 16000 pour les volumes racine de SSD gp3 polyvalents.</p>
RootVolumeThroughput	<p>Débit souhaité en mégioctets par seconde (MiB/s) pour approvisionner le volume racine Amazon EBS attaché à l'instance EC2 de votre environnement.</p> <p>Note</p> <p>Cette option s'applique uniquement aux types de stockages gp3.</p>	Aucune	125 sur 1000

## aws:autoscaling:scheduledaction

Configurez les [actions planifiées \(p. 558\)](#) pour le groupe Auto Scaling de votre environnement. Pour chaque action, indiquez un `resource_name` en plus du nom de l'option, de l'espace de noms et de la valeur de chaque paramètre. Pour obtenir des exemples, consultez la section [Espace de noms aws:autoscaling:scheduledaction \(p. 560\)](#).

Espace de nom : **aws:autoscaling:scheduledaction**

Name (Nom)	Description	Par défaut	Valeurs valides
StartTime	Pour les actions exceptionnelles, choisissez la date et l'heure pour exécuter l'action. Pour des actions récurrentes, choisissez le moment auquel l'action doit être activée.	Aucune	<a href="#">Horodatage ISO-8601</a> unique à travers toutes les actions de dimensionnement planifiées.
EndTime	Date et heure futures (du fuseau horaire UTC/GMT) où vous voulez que l'action de dimensionnement planifiée cesse de se répéter. Si vous ne spécifiez pas la valeur EndTime, l'action se produit conformément à l'expression Recurrence.	Aucune	<a href="#">Horodatage ISO-8601</a> unique à travers toutes les actions de dimensionnement planifiées.

Name (Nom)	Description	Par défaut	Valeurs valides
	<p>Exemple : 2015-04-28T04:07:2Z</p> <p>Lorsqu'une action planifiée se termine, Amazon EC2 Auto Scaling ne revient pas automatiquement à ses paramètres précédents. Configurez une seconde action planifiée pour revenir aux paramètres d'origine selon vos besoins.</p>		
MaxSize	Nombre maximal d'instances à appliquer lorsque l'action s'exécute.	Aucune	0 sur 10000
MinSize	Nombre minimal d'instances à appliquer lorsque l'action s'exécute.	Aucune	0 sur 10000
DesiredCapacity	Définissez la capacité initiale souhaitée pour le groupe Auto Scaling. Lorsque l'action planifiée est appliquée, des déclencheurs ajustent la capacité souhaitée en fonction de leurs paramètres.	Aucune	0 sur 10000
Recurrence	La fréquence à laquelle vous souhaitez que l'action planifiée s'exécute. Si vous ne spécifiez pas de récurrence, l'action de dimensionnement s'exécute une seule fois, comme indiqué par la valeur <code>StartTime</code> .	Aucune	Expression <a href="#">Cron</a> .
Suspend	Définissez le paramètre sur <code>true</code> pour désactiver temporairement une action planifiée récurrente.	<code>false</code>	<code>true</code> <code>false</code>

## aws:autoscaling:trigger

Configurez les déclencheurs de mise à l'échelle pour le groupe Auto Scaling de votre environnement.

### Note

Trois options de cet espace de noms déterminent combien de temps la métrique d'un déclencheur peut rester au-delà de ses limites définies avant le lancement du déclencheur. Ces options sont liées comme suit :

`BreachDuration = Period * EvaluationPeriods`

Les valeurs par défaut de ces options (respectivement 5, 5 et 1) correspondent à cette équation. Si vous spécifiez des valeurs incohérentes, Elastic Beanstalk peut modifier l'une d'entre elles afin que l'équation soit toujours satisfaite.

### Espace de nom : **aws:autoscaling:trigger**

Name (Nom)	Description	Par défaut	Valeurs valides
BreachDuration	Durée (en minutes) pendant laquelle une métrique peut excéder la limite définie (comme indiqué dans <code>UpperThreshold</code> et <code>LowerThreshold</code> ) avant que le déclencheur ne soit appelé.	5	1 sur 600

Name (Nom)	Description	Par défaut	Valeurs valides
LowerBreachScaleIncrements	Nombre d'instances Amazon EC2 à supprimer lorsque vous effectuez une activité de dimensionnement.	-1	
LowerThreshold	Si la mesure passe sous cette valeur pendant la durée de la faille, un déclencheur est appelé.	2000000	0 sur 20000000
MeasureName	<p>Métrique utilisée pour votre déclencheur Auto Scaling.</p> <p>Note</p> <p>HealthyHostCount, UnhealthyHostCount et TargetResponseTime ne sont applicables qu'aux environnements dotés d'un équilibrEUR de charge dédié. Il ne s'agit pas de valeurs de métriques valides pour les environnements configurés avec un équilibrEUR de charge partagé. Pour plus d'informations sur les types d'équilibrEURS de charge, consultez la section <a href="#">ÉquilibrEUR de charge pour votre environnement Elastic Beanstalk (p. 562)</a>.</p>	NetworkOut	CPUUtilization NetworkIn NetworkOut DiskWriteOps DiskReadBytes DiskReadOps DiskWriteBytes Latency RequestCount HealthyHostCount UnhealthyHostCount TargetResponseTime
Period	Spécifie la fréquence à laquelle Amazon CloudWatch mesure les métriques pour votre déclencheur. La valeur est le nombre de minutes entre deux périodes consécutives.	5	1 sur 600
EvaluationPeriods	Nombre de périodes d'évaluation consécutives utilisé pour déterminer si une utilisation hors limites est en cours.	1	1 sur 600
Statistic	Statistique que le déclencheur utilise, telle que Average.	Average	Minimum Maximum Sum Average

Name (Nom)	Description	Par défaut	Valeurs valides
Unit	Unité de mesure du déclencheur, telle que <code>Bytes</code> .	Bytes	Seconds Percent Bytes Bits Count Bytes/Second Bits/Second Count/Second None
UpperBreachScaleIncrement	Nombre d'instances Amazon EC2 à ajouter lors d'une activité de dimensionnement.	1	
UpperThreshold	Si la mesure est supérieure à ce nombre pendant la durée de la faille, un déclencheur est appelé.	6000000	0 sur 20000000

## aws:autoscaling:updatepolicy:rollingupdate

Configurer les mises à jour propagées du groupe Auto Scaling de votre environnement

Espace de nom : **aws:autoscaling:updatepolicy:rollingupdate**

Name (Nom)	Description	Par défaut	Valeurs valides
MaxBatchSize	Le nombre d'instances incluses dans chaque lot de la mise à jour propagée.	Un tiers de la taille minimale du groupe Auto Scaling, arrondie à l'entier suivant le plus élevé.	1 sur 10000
MinInstancesInService	Le nombre minimum d'instances qui doivent être en service au sein du groupe Auto Scaling, tandis que d'autres instances sont résiliées.	La taille minimale du groupe Auto Scaling ou une taille inférieure à la taille maximale du groupe Auto Scaling, la valeur la plus basse prévalant.	0 sur 9999
RollingUpdateEnabled	Si <code>true</code> , permet le déploiement de mises à jour pour un environnement. Les mises à jour propagées sont utiles quand	false	true false

Name (Nom)	Description	Par défaut	Valeurs valides
	<p>vous apportez de petites et fréquentes mises à jour à votre application logicielle Elastic Beanstalk et que vous voulez éviter l'interruption des applications.</p> <p>La configuration de cette valeur sur <code>true</code> active automatiquement les options <code>MaxBatchSize</code>, <code>MinInstancesInService</code> et <code>PauseTime</code>. La définition d'une de ces options définit également automatiquement l'option <code>RollingUpdateEnabled</code> sur <code>true</code>. La configuration de cette option sur <code>false</code> désactive les mises à jour propagées.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console et l'interface de ligne de commande EB CLI remplacent cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>		

Name (Nom)	Description	Par défaut	Valeurs valides
RollingUpdateType	<p>Trois types différents : les mises à jour propagées basées sur le temps, les mises à jour propagées basées sur l'état et les mises à jour immuables.</p> <p>Les mises à jour propagées basées sur le temps appliquent une valeur PauseTime entre les lots. Les mises à jour propagées basées sur l'intégrité attendent que de nouvelles instances transmettent les vérifications de l'état avant de passer au lot suivant. <a href="#">Immutable updates (Les mises à jour immuables) (p. 493)</a> lancent un ensemble complet d'instances dans un nouveau groupe Auto Scaling.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console et l'interface de ligne de commande (CLI) EB remplacent cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>	Time	Time Health Immutable

Name (Nom)	Description	Par défaut	Valeurs valides
PauseTime	Délai d'attente (en secondes, minutes ou heures) du service Elastic Beanstalk entre la fin des mises à jour d'un lot d'instances et le début de celles du lot suivant.	Calculée automatiquement en fonction du type d'instance et du conteneur.	PT0S* (0 seconde) à PT1H (1 heure)
Timeout	Durée maximale (en minutes ou heures) pour que toutes les instances d'un lot d'instances passent les vérifications de l'état avant que la mise à jour soit annulée.	PT30M (30 minutes)	PT5M* (5 minutes) à PT1H (1 heure)  *ISO8601 duration format : PT#H#M#S où chaque # correspond au nombre d'heures, de minutes ou de secondes, respectivement.

## aws:ec2:instances

Configurez les instances de votre environnement, y compris les options d'instances Spot. Cet espace de noms complète [aws:autoscaling:launchconfiguration \(p. 681\)](#) et [aws:autoscaling:asg \(p. 680\)](#).

Pour plus d'informations, consultez [the section called “Groupe Auto Scaling” \(p. 547\)](#).

Espace de noms : **aws:ec2:instances**

Name (Nom)	Description	Par défaut	Valeurs valides
EnableSpot	Activez les demandes d'instances Spot pour votre environnement. Lorsqu'elles sont définies sur <code>false</code> , certaines options de cet espace de noms n'ont aucun effet.	<code>false</code>	<code>true</code> <code>false</code>
InstanceTypes	<p>Liste séparée par des virgules de types d'instance que vous souhaitez que votre environnement utilise (par exemple, <code>t2.micro</code>, <code>t3.micro</code>).</p> <p>Lorsque les instances Spot sont désactivées (<code>EnableSpot</code> défini sur <code>false</code>), seul le premier type d'instance de la liste est utilisé.</p> <p>Le premier type d'instance de la liste dans cette option est équivalent à la valeur de l'option <code>InstanceType</code> dans l'espace de noms <a href="#">aws:autoscaling:launchconfiguration (p. 681)</a>. Nous ne recommandons pas d'utiliser cette dernière option, car elle est</p>	<p>Liste de deux types d'instances</p> <p>Varie selon le compte et la région.</p>	<p>Un à dix types d'instance EC2 (au moins deux recommandés)</p> <p>Varie selon le compte, la région et la zone de disponibilité. Vous pouvez obtenir une liste des types d'instance Amazon EC2 filtrés par ces valeurs. Pour de plus amples informations, consultez <a href="#">Types d'instance disponibles</a> dans le Guide de l'utilisateur</p>

Name (Nom)	Description	Par défaut	Valeurs valides
	<p>obsolète. Si vous spécifiez les deux, le premier type d'instance de la liste de l'option <code>InstanceTypes</code> est utilisé et <code>InstanceType</code> est ignoré.</p> <p>Les types d'instance disponibles dépendent des zones de disponibilité et de la région utilisées. Si vous choisissez un sous-réseau, la zone de disponibilité qui contient ce sous-réseau détermine les types d'instance disponibles.</p> <ul style="list-style-type: none"> <li>• Elastic Beanstalk ne prend pas en charge les types d'instance Amazon EC2 Mac et Amazon EC2 Graviton (basées sur ARM) pour le moment.</li> <li>• Pour de plus amples informations sur les familles d'instance et types d'instance Amazon EC2, consultez <a href="#">Types d'instance</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux ou <a href="#">Types d'instance</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.</li> <li>• Pour de plus amples informations sur les types d'instance disponibles dans les régions, consultez <a href="#">Types d'instance disponibles</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux ou <a href="#">Types d'instance disponibles</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.</li> </ul> <p><b>Note</b></p> <p>Certains comptes AWS plus anciens peuvent fournir à Elastic Beanstalk des types d'instance par défaut qui ne prennent pas en charge les instances Spot (par exemple, t1.micro). Si vous activez les demandes d'instances Spot et que vous obtenez une erreur concernant un type d'instance qui ne prend pas en charge les instances Spot, assurez-vous de configurer des types d'instance qui prennent en charge les instances Spot. Pour choisir des</p>		Amazon EC2 pour les instances Linux ou <a href="#">Types d'instance disponibles</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.

Name (Nom)	Description	Par défaut	Valeurs valides
	<p>types d'instance Spot, utilisez <a href="#">Spot Instance Advisor</a>.</p> <p>Lorsque vous mettez à jour la configuration de votre environnement et supprimez un ou plusieurs types d'instance de l'option <code>InstanceTypes</code>, Elastic Beanstalk résilie toutes les instances Amazon EC2 exécutées sur les types d'instance supprimés. Le groupe Auto Scaling de votre environnement lance ensuite de nouvelles instances, si nécessaire pour compléter la capacité souhaitée, en utilisant les types d'instance spécifiés actuels.</p>		
SpotFleetOnDemandBase	<p>Nombre minimal d'instances à la demande que votre groupe Auto Scaling alloue avant de considérer les instances Spot à mesure que votre environnement augmente.</p> <p>Cette option est pertinente uniquement lorsque <code>EnableSpot</code> est défini sur <code>true</code>.</p>	0	0 à option <code>MaxSize</code> dans l'espace de noms <a href="#">aws:autoscaling:asg</a> (p. 680)
SpotFleetOnDemandAboveBasePercentage	<p>Rebase le pourcentage d'instances à la demande dans le cadre de la capacité supplémentaire que votre groupe Auto Scaling alloue au-delà des instances <code>SpotOnDemandBase</code>.</p> <p>Cette option est pertinente uniquement lorsque <code>EnableSpot</code> est défini sur <code>true</code>.</p>	0 pour un environnement d'instance unique  70 pour un environnement à charge équilibrée	0 sur 100
SpotMaxPrice	<p>Prix maximum par heure unitaire, en dollars américains, que vous êtes prêt à payer pour une instance Spot. Pour des recommandations sur les options tarifaires maximales pour les instances Spot, consultez <a href="#">Historique de tarification d'instances Spot</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.</p> <p>Cette option n'est pertinente que lorsque <code>EnableSpot</code> est défini sur <code>true</code>.</p>	Prix à la demande null par type d'instance.  La valeur de l'option dans ce cas est null.	0.001 sur 20.0

## aws:ec2:vpc

Configurez votre environnement pour lancer des ressources dans un [Amazon Virtual Private Cloud](#) (Amazon VPC) personnalisé. Si vous ne configurez pas de paramètres dans cet espace de noms, Elastic Beanstalk lance les ressources dans le VPC par défaut.

Espace de noms : **aws:ec2:vpc**

Name (Nom)	Description	Par défaut	Valeurs valides
VPCId	L'ID de votre VPC Amazon.	Aucune	
Subnets	Les ID du sous-réseau ou des sous-réseaux du groupe Auto Scaling. Si vous avez plusieurs sous-réseaux, spécifiez la valeur sous forme de chaîne unique séparée par des virgules d'ID de sous-réseau (par exemple, "subnet-11111111,subnet-22222222").	Aucune	
ELBSubnets	L'ID du ou des sous-réseaux pour Elastic Load Balancer. Si vous avez plusieurs sous-réseaux, spécifiez la valeur sous forme de chaîne unique séparée par des virgules d'ID de sous-réseau (par exemple, "subnet-11111111,subnet-22222222").	Aucune	
ELBScheme	Spécifiez <code>internal</code> si vous souhaitez créer un équilibrEUR de charge interne dans votre Amazon VPC afin qu'aucun accÈS à votre application Elastic Beanstalk ne soit possible depuis l'extÉRIEUR de votre Amazon VPC. Si vous spÈCifiez une valeur autre que <code>public</code> ou <code>internal</code> , Elastic Beanstalk ignore la valeur.	public	public internal
DBSubnets	Contient les ID des sous-réseaux de base de données. Ce paramÈtre n'est utilisÉ que si vous souhaitez ajouter une instance DB Amazon RDS dans le cadre de votre application. Si vous avez plusieurs sous-réseaux, spÈCifiez la valeur sous forme de chaîne unique séparée par des virgules d'ID de sous-réseau (par exemple, "subnet-11111111,subnet-22222222").	Aucune	
AssociatePublicIPaddresses	<p>S'adresses à l'option <code>public</code>. Il faut lancer des instances avec des adresses IP publiques dans votre VPC Amazon. Les instances avec des adresses IP publiques ne requiÈrent pas de pÉriphÉrique NAT pour communiquer avec Internet. Vous devez dEFinir la valeur sur <code>true</code> si vous souhaitez inclure votre équilibrEUR de charge et des instances dans un seul sous-réseau public.</p> <p>Cette option n'a aucun effet sur un environnement d'instance unique, qui a toujours une seule instance Amazon EC2 avec une adresse IP Elastic. Cette option est pertinente pour les environnements évolutifs avec équilibrage de charge.</p>	Aucune	true false

## aws:elasticbeanstalk:application

Configurez un chemin de vérification de l'état pour votre application. Pour plus d'informations, consultez [Création de rapports d'intégrité de base \(p. 834\)](#).

Espace de noms : **aws:elasticbeanstalk:application**

Name (Nom)	Description	Par défaut	Valeurs valides
URL de vérification d'intégrité de l'application	<p>Chemin d'accès où les demandes de vérification de l'état sont envoyées. Si ce chemin n'est pas défini, l'équilibrEUR de charge tente d'établir une connexion TCP sur le port 80 pour vérifier le statut de l'état de votre application. Définissez sur un chemin d'accès commençant par / pour envoyer une requête HTTP GET à ce chemin d'accès. Vous pouvez également inclure un protocole (HTTP, HTTPS, TCP, ou SSL) et un port avant le chemin d'accès pour vérifier la connectivité HTTPS ou utiliser un port autre que le port par défaut.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console remplace cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>	Aucune	<p>Les valeurs valides sont les suivantes :</p> <p>/ (HTTP GET pour le chemin d'accès racine)</p> <p>/<i>health</i></p> <p>HTTPS:443/</p> <p>HTTPS:443/<i>health</i></p>

L'interface de ligne de commande (CLI) EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Consultez [Valeurs recommandées \(p. 660\)](#) pour plus de détails.

## aws:elasticbeanstalk:application:environment

Configurez les propriétés d'environnement pour votre application.

Espace de noms : **aws:elasticbeanstalk:application:environment**

Name (Nom)	Description	Par défaut	Valeurs valides
Tout nom de variable d'environnement.	Passez en paires clé-valeur.	Aucune	Toute valeur de variable d'environnement.

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## aws:elasticbeanstalk:cloudwatch:logs

Configurez la diffusion des journaux d'instance pour votre application.

Espace de noms : **aws:elasticbeanstalk:cloudwatch:logs**

Name (Nom)	Description	Par défaut	Valeurs valides
StreamLogs	Indique s'il faut créer des groupes dans des CloudWatch Logs pour les journaux de proxy et de déploiement, et s'il faut diffuser les journaux à partir de chaque instance de votre environnement.	false	true false
DeleteOnTermination	Indique s'il faut supprimer les groupes de journaux lorsque l'environnement est suspendu. Si la valeur est false, les journaux sont conservés pendant RetentionInDays jours.	false	true false
RetentionInDays	Le nombre de jours de conservation des événements de journal avant leur expiration.	7	1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400, 545, 731, 1827, 3653

## aws:elasticbeanstalk:cloudwatch:logs:health

Configurez la diffusion des journaux d'intégrité de l'environnement pour votre application.

Espace de noms : **aws:elasticbeanstalk:cloudwatch:logs:health**

Name (Nom)	Description	Par défaut	Valeurs valides
HealthStreaming	Enables environnements pour lesquels la création de rapports d'état amélioré est activée, indique s'il faut créer un groupe dans CloudWatch Logs pour l'état de l'environnement et archiver les données d'état d'environnement Elastic Beanstalk. Pour plus d'informations sur l'activation de l'état amélioré, consultez <a href="#">aws:elasticbeanstalk:healthreporting:system (p. 706)</a> .	false	true false
DeleteOnTermination	Indique s'il faut supprimer le groupe de journaux lorsque l'environnement est suspendu. Si la valeur est false, les données d'intégrité sont conservées pendant RetentionInDays jours.	false	true false
RetentionInDays	Nombre de jours pendant lequel il faut conserver les données d'intégrité archivées avant leur expiration.	7	1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400,

Name (Nom)	Description	Par défaut	Valeurs valides
			545, 731, 1827, 3653

## aws:elasticbeanstalk:command

Configurez la stratégie de déploiement pour votre code d'application. Pour plus d'informations, consultez the section called “Options de déploiement” (p. 479).

Espace de noms : **aws:elasticbeanstalk:command**

Name (Nom)	Description	Par défaut	Valeurs valides
DeploymentPolicy	<p>Sélectionnez une <a href="#">stratégie de déploiement</a> (p. 479) pour les déploiements de version d'application.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration</a> (p. 737). La console remplace cette option par une <a href="#">valeur recommandée</a> (p. 660).</p>	AllAtOnce	AllAtOnce Rolling RollingWithAddition Immutable TrafficSplitting
Timeout	<p>Délai d'attente, en secondes, pour qu'une instance termine l'exécution des commandes.</p> <p>Elastic Beanstalk ajoute 240 secondes (quatre minutes) à la valeur Timeout. Par exemple, le délai d'attente par défaut est de 840 secondes (600+240), ou 14 minutes.</p>	600	1 sur 3600
BatchSizeType	<p>Le type du nombre spécifié dans BatchSize.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration</a> (p. 737). La console et l'interface de ligne de commande EB CLI remplacent cette option par une <a href="#">valeur recommandée</a> (p. 660).</p>	Percentage Fixed	Percentage Fixed

Name (Nom)	Description	Par défaut	Valeurs valides
BatchSize	<p>Pourcentage ou nombre fixe d'instances Amazon EC2 dans le groupe Auto Scaling sur lesquelles réaliser des déploiements simultanément. Les valeurs valides varient en fonction du paramètre BatchSizeType utilisé.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console et l'interface de ligne de commande EB CLI remplacent cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>	100	1 à 100 (Percentage). 1 sur <a href="#">aws:autoscaling:asg::Max (Fixed)</a>
IgnoreHealthCheck	N'annulez pas un déploiement en raison d'un échec de vérifications de l'état.	false	true false

## aws:elasticbeanstalk:environment

Configurez l'architecture et le rôle de service de votre environnement.

Espace de noms : **aws:elasticbeanstalk:environment**

Name (Nom)	Description	Par défaut	Valeurs valides
EnvironmentType	Affectez la valeur SingleInstance pour lancer une instance EC2 sans équilibrEUR de charge.	LoadBalanced	SingleInstance LoadBalanced
ServiceRole	<p>Le nom d'un rôle IAM qu'Elastic Beanstalk utilise pour gérer des ressources pour l'environnement. Spécifiez un nom de rôle (éventuellement précédé d'un chemin personnalisé) ou son ARN.</p> <p><b>Exemples :</b></p> <ul style="list-style-type: none"> <li>aws-elasticbeanstalk-service-role</li> <li><b>custom-path/custom-role</b></li> <li>arn:aws:iam::123456789012:role/ aws-elasticbeanstalk-service-role</li> </ul> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un</p>	Aucune	Nom de rôle IAM, chemin/nom ou ARN

Name (Nom)	Description	Par défaut	Valeurs valides
	environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a> . La console et l'interface de ligne de commande (CLI) EB remplace cette option par une <a href="#">valeur recommandée (p. 660)</a> .		
LoadBalancerType	Type de l'équilibrer de charge pour votre environnement. Pour plus d'informations, consultez <a href="#">the section called “Équilibrer de charge” (p. 562)</a> .	classic	classic application network
LoadBalancerIsShared	Indique si l'équilibrer de charge de l'environnement est dédié ou partagé. Cette option ne peut être définie que pour un Application Load Balancer. Elle ne peut pas être modifiée après la création de l'environnement.  Avec <code>false</code> , l'environnement dispose de son propre équilibrer de charge dédié, créé et géré par Elastic Beanstalk. Avec <code>true</code> , l'environnement utilise un équilibrer de charge partagé, créé par vous et spécifié dans l'option SharedLoadBalancer de l'espace de noms <a href="#">aws:elbv2:loadbalancer (p. 723)</a> .	false	true false

## aws:elasticbeanstalk:environment:process:default

Configurez le processus par défaut de votre environnement.

Espace de noms : **aws:elasticbeanstalk:environment:process:default**

Name (Nom)	Description	Par défaut	Valeurs valides
DeregistrationDelay	Délai d'attente, en secondes, d'achèvement des demandes actives avant l'annulation de l'enregistrement.	20	0 sur 3600
HealthCheckInterval	Intervalle, en secondes, pendant lequel Elastic Load Balancing vérifie l'état des instances Amazon EC2 de votre application.	Avec un équilibrer de charge classique ou d'application : 15  Avec un équilibrer de charge réseau : 30	Avec un équilibrer de charge classique ou d'application : 5 à 300  Avec un équilibrer de charge réseau : 10, 30
HealthCheckPath	Chemin d'accès vers lequel les demandes HTTP pour les vérifications de l'état sont envoyées.	/	Un chemin d'accès routable.

Name (Nom)	Description	Par défaut	Valeurs valides
HealthCheckTimeout	Délai d'attente, en secondes, d'une réponse pendant une vérification de l'état.  Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge d'application.	5	1 sur 60
HealthyThresholdCount	Nombre de demandes consécutives réussies avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	Avec un équilibrEUR de charge classique ou d'application : 3  Avec un équilibrEUR de charge réseau : 5	2 sur 10
MatcherHTTPCode	Liste séparée par des virgules de codes HTTP qui indiquent qu'une instance est en bonne santé.  Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge réseau ou un équilibrEUR de charge d'application.	200	Avec un équilibrEUR de charge d'application : 200 à 499  Avec un équilibrEUR de charge réseau : 200 à 399
Port	Port d'écoute du processus.	80	1 sur 65535
Protocol	Protocole utilisé par le processus.  Avec un équilibrEUR de charge d'application, vous pouvez uniquement définir cette option sur <b>HTTP</b> ou <b>HTTPS</b> .  Avec un équilibrEUR de charge réseau, vous pouvez uniquement définir cette option sur <b>TCP</b> .	Avec un équilibrEUR de charge classique ou d'application : <b>HTTP</b>  Avec un équilibrEUR de charge réseau : <b>TCP</b>	<b>TCP</b> <b>HTTP</b> <b>HTTPS</b>

Name (Nom)	Description	Par défaut	Valeurs valides
StickinessEnabled	Définissez la valeur sur true pour activer les sessions permanentes.  Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge d'application.	'false'	'false' 'true'
StickinessLBCookieDuration	Durée de vie, en secondes, du cookie de session permanente.  Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge d'application.	86400 (une journée)	1 sur 604800
StickinessType	Définissez la valeur sur lb_cookie pour utiliser des cookies pour les sessions permanentes.  Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge d'application.	lb_cookie	lb_cookie
UnhealthyThresholdCount	Nombre de demandes consécutives infructueuses avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	5	2 sur 10

## aws:elasticbeanstalk:environment:process:process\_name

Configurez des processus supplémentaires pour votre environnement.

Espace de nom : **aws:elasticbeanstalk:environment:process:*process\_name***

Name (Nom)	Description	Par défaut	Valeurs valides
DeregistrationDelay	Délai d'attente, en secondes, d'achèvement des demandes actives avant l'annulation de l'enregistrement.	20	0 sur 3600

Name (Nom)	Description	Par défaut	Valeurs valides
HealthCheckInterval	Intervalle, en secondes, pendant lequel Elastic Load Balancing vérifie l'état des instances Amazon EC2 pour votre application.	Avec un équilibreur de charge classique ou d'application : 15  Avec un équilibreur de charge réseau : 30	Avec un équilibreur de charge classique ou d'application : 5 à 300  Avec un équilibreur de charge réseau : 10, 30
HealthCheckPath	Chemin d'accès vers lequel les demandes HTTP pour les vérifications de l'état sont envoyées.	/	Un chemin d'accès routable.
HealthCheckTimeout	Délai d'attente, en secondes, d'une réponse pendant une vérification de l'état.  Cette option est applicable uniquement aux environnements avec un équilibreur de charge d'application.	5	1 sur 60
HealthyThresholdCount	Nombre de demandes consécutives réussies avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	Avec un équilibreur de charge classique ou d'application : 3  Avec un équilibreur de charge réseau : 5	2 sur 10
MatcherHTTPCode	Liste séparée par des virgules de codes HTTP indiquant qu'une instance est saine.  Cette option est applicable uniquement aux environnements avec un équilibreur de charge réseau ou un équilibreur de charge d'application.	200	Avec un équilibreur de charge d'application : 200 à 499  Avec un équilibreur de charge réseau : 200 à 399
Port	Port d'écoute du processus.	80	1 sur 65535

Name (Nom)	Description	Par défaut	Valeurs valides
Protocol	<p>Protocole utilisé par le processus.</p> <p>Avec un équilibrer de charge d'application, vous pouvez uniquement définir cette option sur <b>HTTP</b> ou <b>HTTPS</b>.</p> <p>Avec un équilibrer de charge réseau, vous pouvez uniquement définir cette option sur <b>TCP</b>.</p>	<p>Avec un équilibrer de charge classique ou d'application : <b>HTTP</b></p> <p>Avec un équilibrer de charge réseau : <b>TCP</b></p>	<b>TCP</b> <b>HTTP</b> <b>HTTPS</b>
StickinessEnabled	<p>Définissez la valeur sur <code>true</code> pour activer les sessions permanentes.</p> <p>Cette option est applicable uniquement aux environnements avec un équilibrer de charge d'application.</p>	' <code>false</code> '	' <code>false</code> ' ' <code>true</code> '
StickinessLBCookieDuration	<p>Durée de vie, en secondes, du cookie de session permanente.</p> <p>Cette option est applicable uniquement aux environnements avec un équilibrer de charge d'application.</p>	86400 (une journée)	1 sur 604800
StickinessType	<p>Définissez la valeur sur <code>lb_cookie</code> pour utiliser des cookies pour les sessions permanentes.</p> <p>Cette option est applicable uniquement aux environnements avec un équilibrer de charge d'application.</p>	<code>lb_cookie</code>	<code>lb_cookie</code>
UnhealthyThresholdCount	Nombre de demandes consécutives infructueuses avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	5	2 sur 10

## aws:elasticbeanstalk:environment:proxy:staticfiles

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

### Note

Cet espace de noms s'applique aux branches de plateforme basées sur Amazon Linux 2. Si votre environnement utilise une version de plateforme basée sur une AMI Amazon Linux (antérieure à Amazon Linux 2), reportez-vous à la section [the section called “Options spécifiques à une plateforme” \(p. 727\)](#) pour les espaces de noms de fichiers statiques spécifiques à la plateforme.

Espace de nom : **aws:elasticbeanstalk:environment:proxy:staticfiles**

Name (Nom)	Valeur
Chemin d'accès où le serveur proxy sert les fichiers. Démarrez la valeur par /.  Par exemple, spécifiez /images pour traiter les fichiers au niveau de <b>subdomain.eleasticbeanstalk.com/images</b> .	Nom du dossier contenant les fichiers.  Par exemple, spécifiez staticimages pour traiter les fichiers depuis un dossier nommé staticimages au niveau supérieur de votre solution groupée de fichiers source.

## aws:elasticbeanstalk:healthreporting:system

Configurez des rapports améliorés sur l'état pour votre environnement.

Espace de nom : **aws:elasticbeanstalk:healthreporting:system**

Name (Nom)	Description	Par défaut	Valeurs valides
SystemType	Système de rapport d'état ( <a href="#">de base (p. 834)</a> ou <a href="#">amélioré (p. 837)</a> ). Les rapports améliorés sur l'état de santé nécessitent un <a href="#">rôle de service (p. 21)</a> , une version 2 de la plateforme ou une <a href="#">version de plateforme (p. 31)</a> ultérieure.  Note  Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a> . La console et l'interface de ligne de commande (CLI) EB remplacent cette option par une <a href="#">valeur recommandée (p. 660)</a> .	basic	basic enhanced
ConfigDocument	Document JSON décrivant les métriques d'environnement et d'instance à publier sur CloudWatch.	Aucune	

Name (Nom)	Description	Par défaut	Valeurs valides
EnhancedHealthAuth	<p>Permet à l'API interne qu'Elastic Beanstalk utilise à communiquer des informations d'état améliorées depuis les instances de votre environnement vers le service Elastic Beanstalk.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">the section called “Rôles d'intégrité améliorée” (p. 843)</a>.</p> <p><b>Note</b></p> <p>Cette option ne s'applique qu'aux rapports améliorés sur l'état (c'est-à-dire quand SystemType est défini sur enhanced).</p>	true	true false
HealthCheckSuccessThreshold	<p>Abaisse le seuil de réussite des vérifications de l'état des instances.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console remplace cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>	ok	Ok Warning Degraded Severe

## aws:elasticbeanstalk:hostmanager

Configurez les instances EC2 dans votre environnement pour télécharger des journaux ayant subi une rotation dans Amazon S3.

Espace de nom : **aws:elasticbeanstalk:hostmanager**

Name (Nom)	Description	Par défaut	Valeurs valides
LogPublicationControl	Copie les fichiers journaux des instances Amazon EC2 de votre application dans le compartiment Amazon S3 associé à votre application.	false	true false

## aws:elasticbeanstalk:managedactions

Configurez les mises à jour gérées de la plateforme pour votre environnement.

Espace de noms : **aws:elasticbeanstalk:managedactions**

Name (Nom)	Description	Par défaut	Valeurs valides
ManagedActionsEnabled	Activez <a href="#">Mises à jour de la plateforme gérée (p. 506)</a> .	true	true false

Name (Nom)	Description	Par défaut	Valeurs valides
	Lorsque vous définissez cela sur <code>true</code> , vous devez également spécifier une <a href="#">PreferredStartTime</a> et <a href="#">UpdateLevel</a> (p. 708).		
PreferredStartTime	Configurez une fenêtre de maintenance pour les actions gérées au format UTC.  Par exemple, "Tue:09:00".	Aucune	Jour et heure au format <i>jour:heure:minutes</i>  .
ServiceRoleForManagedUpdates	<b>Notes</b> d'un rôle IAM utilisé par Elastic Beanstalk pour effectuer des mises à jour de plateforme gérées pour votre environnement.  Vous pouvez utiliser soit le même rôle que celui que vous avez spécifié pour l'option <code>ServiceRole</code> de l'espace de noms <a href="#">aws:elasticbeanstalk:environment</a> , soit le <a href="#">rôle lié au service de mises à jour gérées de votre compte</a> (p. 941). Dans ce dernier cas, si le compte n'a pas encore de rôle lié au service de mises à jour gérées, Elastic Beanstalk le crée.	Aucune	Identique à <code>ServiceRole</code> ou <code>AWSServiceRoleForElasticBeanstalk</code>

## aws:elasticbeanstalk:managedactions:platformupdate

Configurez les mises à jour gérées de la plateforme pour votre environnement.

Espace de noms : **aws:elasticbeanstalk:managedactions:platformupdate**

Name (Nom)	Description	Par défaut	Valeurs valides
UpdateLevel	Le plus haut niveau de mise à jour à appliquer aux mises à jour de plateforme gérées. Les plateformes sont avec la version <i>majeure.mineure.correctif</i> . Par exemple, 2.0.8 dispose d'une version majeure de 2, d'une version mineure de 0 et d'une version corrective de 8.	Aucune	<code>patch</code> pour les mises à jour de version corrective uniquement.  <code>minor</code> pour les mises à jour des deux versions : mineure et corrective.
InstanceRefreshEnabled	Activez le remplacement d'instance hebdomadaire.	<code>false</code>	<code>true</code> <code>false</code>

Name (Nom)	Description	Par défaut	Valeurs valides
	Nécessite la définition de <code>ManagedActionsEnabled</code> sur <code>true</code> .		

## aws:elasticbeanstalk:monitoring

Configurez votre environnement pour résilier les instances EC2 dont les vérifications d'état ont échoué.

Espace de nom : **aws:elasticbeanstalk:monitoring**

Name (Nom)	Description	Par défaut	Valeurs valides
Automatically Terminate Unhealthy Instances	<p>Mettez fin à une instance en cas d'échec de vérifications d'intégrité.</p> <p><b>Note</b></p> <p>Cette option est uniquement prise en charge sur les <a href="#">environnements hérités (p. 507)</a>. Il détermine l'état d'une instance en fonction de son accessibilité et d'autres métriques basées sur l'instance. Elastic Beanstalk ne permet pas d'arrêter automatiquement des instances en fonction de l'état de l'application.</p>	<code>true</code>	<code>true</code> <code>false</code>

## aws:elasticbeanstalk:sns:topics

Configurez les notifications pour votre environnement.

Espace de nom : **aws:elasticbeanstalk:sns:topics**

Name (Nom)	Description	Par défaut	Valeurs valides
Notification Endpoint	<p>Point de terminaison dans lequel vous souhaitez être informé des événements importants qui affectent votre application.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console remplace cette</p>	Aucune	

Name (Nom)	Description	Par défaut	Valeurs valides
	option par une valeur recommandée (p. 660).		
Notification Protocol	Protocole utilisé pour envoyer des notifications à votre point de terminaison.	email	http https email email-json sns
Notification Topic ARN	Amazon Resource Name (ARN) pour la rubrique à laquelle vous avez souscrit.	Aucune	
Notification Topic Name	Nom de la rubrique à laquelle vous avez souscrit.	Aucune	

## aws:elasticbeanstalk:sqsd

Configurez la file d'attente Amazon SQS pour un environnement de travail.

Espace de nom : **aws:elasticbeanstalk:sqsd**

Name (Nom)	Description	Par défaut	Valeurs valides
WorkerQueueURL	URL de la file d'attente à partir de laquelle le démon de la couche d'environnement de travail lit les messages.  <b>Note</b>  Lorsque vous ne spécifiez pas de valeur, la file d'attente créée automatiquement par Elastic Beanstalk est une file d'attente Amazon SQS <b>standard</b> . Lorsque vous indiquez une valeur, vous pouvez fournir l'URL d'une file d'attente standard ou celle d'une file d'attente <b>FIFO</b> Amazon SQS. Sachez que si vous spécifiez une file d'attente FIFO, les <b>tâches périodiques</b> (p. 525) ne sont pas prises en charge.	génération automatique	Si vous ne spécifiez pas de valeur, Elastic Beanstalk crée automatiquement une file d'attente.
HttpPath	Chemin d'accès relatif à l'application vers laquelle les messages HTTP POST sont envoyés.	/	
MimeType	Type MIME du message envoyé dans la demande HTTP POST.	application/json	application/json

Name (Nom)	Description	Par défaut	Valeurs valides
			<b>application/x-www-form-urlencoded</b> <b>application/xml</b> <b>text/plain</b> Type MIME personnalisé.
HttpConnections	<p>Nombre maximal de connexions simultanées à toute application au sein d'une instance Amazon EC2.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console remplace cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>	50	1 sur 100
ConnectTimeout	Délai d'attente, en secondes, de connexion réussie à une application.	5	1 sur 60
InactivityTimeout	<p>Délai d'attente, en secondes, d'une réponse sur une connexion existante à une application.</p> <p>Le message est traité à nouveau jusqu'à ce que le démon reçoive une réponse 200 (OK) à partir de l'application dans la couche d'environnement de travail ou lorsque la <code>RetentionPeriod</code> expire.</p>	299	1 sur 36000
VisibilityTimeout	<p>La quantité de temps, en secondes, durant laquelle un message entrant à partir de la file d'attente Amazon SQS est verrouillé afin d'être traité. Une fois que la durée configurée est écoulée, alors le message est à nouveau rendu visible dans la file d'attente permettant à tout autre démon de le lire.</p>	300	0 sur 43200
ErrorVisibilityTimeout	<p>La quantité de temps, en secondes, qui s'écoule avant qu'Elastic Beanstalk renvoie un message à la file d'attente Amazon SQS après l'échec d'une tentative de traitement avec une erreur explicite.</p>	2 secondes	0 à 43200 secondes
RetentionPeriod	Délai d'attente, en secondes, pendant lequel un message est valide et activement traité.	345600	60 sur 1209600

Name (Nom)	Description	Par défaut	Valeurs valides
MaxRetries	Nombre maximum de tentatives d'envoi du message par Elastic Beanstalk à l'application web qui le traite avant de passer le message à la file d'attente de lettres mortes.	10	1 sur 100

## aws:elasticbeanstalk:trafficsplitting

Configurez les déploiements avec répartition du trafic pour votre environnement.

Cet espace de noms s'applique lorsque vous définissez l'option `DeploymentPolicy` de l'espace de noms [aws:elasticbeanstalk:command \(p. 699\)](#) sur `TrafficSplitting`. Pour de plus amples informations sur les stratégies de déploiement, veuillez consulter [the section called “Options de déploiement” \(p. 479\)](#).

Espace de nom : **aws:elasticbeanstalk:trafficsplitting**

Name (Nom)	Description	Par défaut	Valeurs valides
NewVersionPercent	Pourcentage initial du trafic client entrant qu'Elastic Beanstalk déplace vers les instances d'environnement exécutant la nouvelle version de l'application que vous déployez.	10	1 sur 100
EvaluationTime	Période, en minutes, qu'Elastic Beanstalk attend après un déploiement sain initial avant de déplacer tout le trafic client entrant vers la nouvelle version de l'application que vous déployez.	5	3 sur 600

## aws:elasticbeanstalk:xray

Exécutez le démon AWS X-Ray pour relayer les informations de trace de votre application [X-Ray intégrée \(p. 638\)](#).

Espace de nom : **aws:elasticbeanstalk:xray**

Name (Nom)	Description	Par défaut	Valeurs valides
XRayEnabled	Définissez sur <code>true</code> pour exécuter le démon X-Ray sur les instances dans votre environnement.	<code>false</code>	<code>true</code> <code>false</code>

## aws:elb:healthcheck

Configurez les vérifications de l'état pour un Classic Load Balancer.

Espace de nom : **aws:elb:healthcheck**

Name (Nom)	Description	Par défaut	Valeurs valides
HealthyThreshold	Nombre de demandes consécutives réussies avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	3	2 sur 10
Interval	Intervalle pendant lequel Elastic Load Balancing vérifie l'état des instances Amazon EC2 de votre application.	10	5 sur 300
Timeout	Délai d'attente, en secondes, pendant lequel Elastic Load Balancing attend une réponse avant de considérer l'instance non réactive.	5	2 sur 60
UnhealthyThreshold	Nombre de demandes consécutives infructueuses avant qu'Elastic Load Balancing ne modifie le statut de l'état de l'instance.	5	2 sur 10
(obsolète) Target	Destination d'une instance backend vers laquelle les vérifications de l'état sont envoyées. Utilisez <a href="#">Application Healthcheck URL</a> dans l'espace de noms <a href="#">aws:elasticbeanstalk:application</a> (p. 697) à la place.	TCP:80	Cible au format <b>PROTOCOL:PORT/PATH</b>

## aws:elb:loadbalancer

Configurez le Classic Load Balancer de votre environnement.

Plusieurs options de cet espace de noms ne sont plus prises en charge et sont remplacées par des options spécifiques à l'écouteur dans l'espace de noms [aws:elb:listener](#) (p. 714). Avec ces options qui ne sont plus prises en charge, vous ne pouvez configurer que deux écouteurs (un sécurisé et un non sécurisé) sur les ports standard.

Espace de nom : **aws:elb:loadbalancer**

Nom	Description	Par défaut	Valeurs valides
CrossZone	<p>Configurez l'équilibrer de charge pour acheminer le trafic uniformément sur toutes les instances dans toutes les zones de disponibilité et non uniquement dans chaque zone.</p> <p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration</a> (p. 737). La console et l'interface de ligne de commande EB CLI remplacent cette option par une <a href="#">valeur recommandée</a> (p. 660).</p>	false	true false
SecurityGroups	Attribuez un ou plusieurs groupes de sécurité que vous avez créés à l'équilibrer de charge.	Aucune	Un ou plusieurs ID

Nom	Description	Par défaut	Valeurs valides
			de groupes de sécurité.
ManagedSecurityGroup	<p>Attribuez un groupe de sécurité existant à l'équilibrer de charge de votre environnement au lieu d'en créer un. Pour utiliser ce paramètre, mettez à jour le paramètre <code>SecurityGroups</code> de cet espace de noms afin d'inclure l'ID de votre groupe de sécurité et supprimez l'ID du groupe de sécurité créé automatiquement, le cas échéant.</p> <p>Pour autoriser le trafic à partir de l'équilibrer de charge vers les instances EC2 de votre environnement, Elastic Beanstalk ajoute une règle au groupe de sécurité des instances, qui autorise le trafic entrant depuis le groupe de sécurité géré.</p>	Aucune	Un ID de groupe de sécurité.
(obsolète) LoadBalancerHTTPPort	Port d'écoute pour l'écouteur non sécurisé.	80  80	OFF  HTTP
(obsolète) LoadBalancerPortProtocol	Protocole à utiliser sur l'écouteur non sécurisé.		HTTP  TCP
(obsolète) LoadBalancerHTTPSPort	Port d'écoute pour l'écouteur sécurisé.	OFF  443  8443	OFF  443  8443
(obsolète) LoadBalancerSSLPorProtocol	Protocole à utiliser sur l'écouteur sécurisé.	HTTPS	HTTPS  SSL
(obsolète) SSLCertificateId	Amazon Resource Name (ARN) d'un certificat SSL à lier à l'écouteur sécurisé.	Aucune	

## aws:elb:listener

Configurez l'écouteur par défaut (port 80) sur un Classic Load Balancer.

Espace de nom : **aws:elb:listener**

Nom	Description	Par défaut	Valeurs valides
ListenerProtocol	Le protocole utilisé par l'écouteur.	HTTP	HTTP TCP
InstancePort	Le port que cet écouteur utilise pour communiquer avec les instances EC2.	80	1 sur 65535
InstanceProtocol	Le protocole que cet écouteur utilise pour communiquer avec les instances EC2.	HTTP lorsque	HTTP ou HTTPS

Nom	Description	Par défaut	Valeurs valides
	<p>Il doit se trouver sur la même couche de protocole Internet que le <code>ListenerProtocol</code>. Il doit avoir également le même niveau de sécurité que tout autre écouteur utilisant le même <code>InstancePort</code> que cet écouteur.</p> <p>Par exemple, si le <code>ListenerProtocol</code> est <code>HTTPS</code> (couche d'application, via une connexion sécurisée), vous pouvez définir le <code>InstanceProtocol</code> sur <code>HTTP</code> (également au niveau de la couche d'application, via une connexion non sécurisée). En outre, si vous définissez le <code>InstancePort</code> sur <code>80</code>, vous devez définir le <code>InstanceProtocol</code> sur <code>HTTP</code> dans tous les autres écouteurs avec le <code>InstancePort</code> défini sur <code>80</code>.</p>	<code>ListenerProtocol</code> lorsque <code>ListenerProtocol</code> est <code>HTTP</code> <code>TCP</code> lorsque <code>ListenerProtocol</code> est <code>TCP</code>	<code>localhost</code> <code>ListenerProtocol</code> est <code>HTTP</code> ou <code>HTTPS</code> <code>ListenerProtocol</code> est <code>TCP</code> <code>TCP</code> ou <code>SSL</code> <code>localhost</code> <code>ListenerProtocol</code> est <code>TCP</code> ou <code>SSL</code>
<code>PolicyNames</code>	Une liste de noms de stratégie séparés par des virgules à appliquer au port pour cet écouteur. Nous vous recommandons d'utiliser l'option <code>LoadBalancerPorts</code> de l'espace de noms <a href="#">aws:elb:policies (p. 716)</a> à la place.	Aucune	
<code>ListenerEnabled</code>	Indique si cet écouteur est activé. Si vous spécifiez <code>false</code> , l'écouteur n'est pas inclus dans l'équilibrage de charge.	<code>true</code>	<code>true</code> <code>false</code>

## aws:elb:listener:listener\_port

Configurez des écouteurs supplémentaires sur un Classic Load Balancer.

Espace de nom : **aws:elb:listener:*listener\_port***

Nom	Description	Par défaut	Valeurs valides
<code>ListenerProtocol</code>	Le protocole utilisé par l'écouteur.	<code>HTTP</code>	<code>HTTP</code> <code>HTTPS</code> <code>TCP</code> <code>SSL</code>
<code>InstancePort</code>	Le port que cet écouteur utilise pour communiquer avec les instances EC2.	Le même que <i>listener_port</i>	1 sur 65535
<code>InstanceProtocol</code>	<p>Le protocole que cet écouteur utilise pour communiquer avec les instances EC2.</p> <p>Il doit se trouver sur la même couche de protocole Internet que le <code>ListenerProtocol</code>. Il doit avoir également le même niveau de sécurité que tout autre écouteur utilisant le même <code>InstancePort</code> que cet écouteur.</p> <p>Par exemple, si le <code>ListenerProtocol</code> est <code>HTTPS</code> (couche d'application, via une connexion sécurisée), vous pouvez définir le</p>	<code>HTTP</code> lorsque <code>ListenerProtocol</code> est <code>HTTP</code> ou <code>HTTPS</code> <code>TCP</code> lorsque <code>ListenerProtocol</code> est <code>TCP</code> ou <code>SSL</code>	<code>HTTP</code> ou <code>HTTPS</code> <code>localhost</code> <code>ListenerProtocol</code> est <code>HTTP</code> ou <code>HTTPS</code> <code>localhost</code> <code>ListenerProtocol</code> est <code>TCP</code> ou <code>SSL</code> <code>localhost</code> <code>ListenerProtocol</code> est <code>TCP</code> ou <code>SSL</code>

Nom	Description	Par défaut	Valeurs valides
	InstanceProtocol sur HTTP (également au niveau de la couche d'application, via une connexion non sécurisée). En outre, si vous définissez le InstancePort sur 80, vous devez définir le InstanceProtocol sur HTTP dans tous les autres écouteurs avec le InstancePort défini sur 80.		est TCP ou SSL
PolicyNames	Une liste de noms de stratégie séparés par des virgules à appliquer au port pour cet écouteur. Nous vous conseillons d'utiliser à la place l'option LoadBalancerPorts de l'espace de noms <a href="#">aws:elb:policies (p. 716)</a> .	Aucune	
SSLCertificateId	Amazon Resource Name (ARN) d'un certificat SSL à lier au processus d'écoute.	Aucune	
ListenerEnabled	Indique si cet écouteur est activé. Si vous spécifiez false, l'écouteur n'est pas inclus dans l'équilibreur de charge.	true si une autre option est définie. false dans le cas contraire.	true false

## aws:elb:policies

Modifiez l'adhérence par défaut et les stratégies globales d'équilibreur de charge pour un Classic Load Balancer.

Espace de nom : **aws:elb:policies**

Name (Nom)	Description	Par défaut	Valeurs valides
ConnectionDrainingEnabled	Spécifie si l'équilibreur de charge gère les connexions existantes vers les instances qui sont devenues défectueuses ou désinscrites pour terminer les requêtes en cours.  Note  Si vous utilisez la console Elastic Beanstalk ou l'interface de ligne de commande (CLI) EB pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a> . La console et l'interface de ligne de commande EB CLI remplacent cette option par une <a href="#">valeur recommandée (p. 660)</a> .	false	true false
ConnectionDrainingTimeout	Nombre maximum de secondes pendant lesquels l'équilibreur de charge maintient des connexions existantes sur une instance au cours du drainage de la connexion avant de forcer la fermeture des connexions.	20	1 sur 3600

Name (Nom)	Description	Par défaut	Valeurs valides
	<p><b>Note</b></p> <p>Si vous utilisez la console Elastic Beanstalk pour créer un environnement, vous ne pouvez pas définir cette option dans un <a href="#">fichier de configuration (p. 737)</a>. La console remplace cette option par une <a href="#">valeur recommandée (p. 660)</a>.</p>		
ConnectionSettingIdleTimeOut	Durée, en secondes, pendant laquelle l'équilibrEUR de charge attend que les données soit envoyées ou reçues via la connexion. Si aucune donnée n'a été envoyée ou reçue une fois cette période écoulée, l'équilibrEUR de charge ferme la connexion.	60	1 sur 3600
LoadBalancerPorts	Une liste séparée par des virgules des ports d'écoute auxquels la stratégie par défaut (AWSEB-ELB-StickinessPolicy) s'applique.	Aucune	Vous pouvez utiliser :all pour indiquer tous les ports d'écoute
Stickiness Cookie Expiration	La quantité de temps, en secondes, durant laquelle chaque cookie est valide. Utilise la stratégie par défaut (AWSEB-ELB-StickinessPolicy).	0	0 sur 1000000
Stickiness Policy	Lie la session d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la session soient envoyées à la même instance de serveur. Utilise la stratégie par défaut (AWSEB-ELB-StickinessPolicy).	false	true false

## aws:elb:policies:policy\_name

Créez des stratégies globales d'équilibrEUR de charge supplémentaires pour un Classic Load Balancer.

Espace de nom : **aws:elb:policies:policy\_name**

Name (Nom)	Description	Par défaut	Valeurs valides
CookieName	Le nom du cookie généré par l'application qui contrôle les durées de vie de la session d'une stratégie AppCookieStickinessPolicyType. Cette stratégie peut être associée uniquement à des auditeurs HTTP/HTTPS.	Aucune	
InstancePorts	Une liste des ports d'instance séparés par des virgules auxquels cette stratégie s'applique.	Aucune	Une liste de ports ou :all
LoadBalancerPorts	Une liste des ports d'écoute séparés par des virgules auxquels cette stratégie s'applique.	Aucune	Une liste de ports ou :all

Name (Nom)	Description	Par défaut	Valeurs valides
ProxyProtocol	Pour une stratégie <code>ProxyProtocolPolicyType</code> , spécifie s'il convient d'inclure l'adresse IP et le port de la demande d'origine pour les messages TCP. Cette stratégie peut être associée uniquement aux écouteurs TCP/SSL.	Aucune	<code>true</code> <code>false</code>
PublicKey	Le contenu d'une clé publique pour une stratégie <code>PublicKeyPolicyType</code> à utiliser lors de l'authentification du ou des serveurs backend. Cette stratégie ne peut pas être appliquée directement aux serveurs backend ou aux écouteurs. Elle doit faire partie d'une stratégie <code>BackendServerAuthenticationPolicyType</code> .	Aucune	
PublicKeyPolicyNames	Liste séparée par des virgules de noms de stratégie (à partir des stratégies <code>PublicKeyPolicyType</code> ) pour une stratégie <code>BackendServerAuthenticationPolicyType</code> qui contrôle l'authentification sur un ou plusieurs serveurs backend. Cette stratégie peut être associée uniquement à des serveurs backend qui utilisent HTTPS/SSL.	Aucune	
SSLProtocols	Une liste séparée par des virgules de protocoles SSL devant être activés pour une stratégie <code>SSLNegotiationPolicyType</code> qui définit les chiffrements et les protocoles acceptés par l'équilibrEUR de charge. Cette stratégie peut être associée uniquement aux écouteurs HTTPS/SSL.	Aucune	
SSLReferencePolicy	Nom d'une stratégie de sécurité prédéfinie qui respecte les bonnes pratiques de sécurité AWS et que vous souhaitez activer pour une stratégie <code>SSLNegotiationPolicyType</code> qui définit les chiffrements et les protocoles acceptés par l'équilibrEUR de charge. Cette stratégie peut être associée uniquement aux écouteurs HTTPS/SSL.	Aucune	
Stickiness Cookie Expiration	La quantité de temps, en secondes, durant laquelle chaque cookie est valide.	0	0 sur 1000000
Stickiness Policy	Lie la session d'un utilisateur à une instance de serveur spécifique afin que toutes les demandes provenant de l'utilisateur pendant la session soient envoyées à la même instance de serveur.	<code>false</code>	<code>true</code> <code>false</code>

## aws:elbv2:listener:default

Configurez l'écouteur par défaut (port 80) sur un Application Load Balancer ou un Network Load Balancer.

Cet espace de noms ne s'applique pas à un environnement qui utilise un équilibrEUR de charge partagé.  
Les équilibrEURS de charge partagés n'ont pas d'écouteur par défaut.

Espace de nom : **aws:elbv2:listener:default**

Name (Nom)	Description	Par défaut	Valeurs valides
DefaultProcess	Nom du <a href="#">processus (p. 701)</a> vers lequel transférer le trafic lorsqu'aucune règle ne correspond à la demande.	default	Un nom de processus.
ListenerEnabled	Définissez la valeur sur <code>false</code> pour désactiver l'écouteur. Vous pouvez utiliser cette option pour désactiver l'écouteur par défaut sur le port 80.	true	<code>true</code> <code>false</code>
Protocol	Protocole du trafic à traiter.	Avec un équilibrEUR de charge d'application : <code>HTTP</code>  Avec un équilibrEUR de charge réseau : <code>TCP</code>	Avec un équilibrEUR de charge d'application : <code>HTTP, HTTPS</code>  Avec un équilibrEUR de charge réseau : <code>TCP</code>
Rules	Liste des <a href="#">règles (p. 721)</a> à appliquer à l'écouteur  Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge Application Load Balancer.	Aucune	Liste séparée par des virgules de noms de règles.
SSLCertificateArns	Amazon Resource Name (ARN) du certificat SSL à lier à l'écouteur.  Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge Application Load Balancer.	Aucune	L'ARN d'un certificat stocké dans IAM ou ACM.
SSLPolicy	Spécifiez une stratégie de sécurité à appliquer à l'écouteur.  Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge Application Load Balancer.	Aucune (valeur par défaut ELB)	Le nom d'une stratégie de sécurité d'équilibrEUR de charge.

Name (Nom)	Description	Par défaut	Valeurs valides
	charge Application Load Balancer.		

## aws:elbv2:listener:listener\_port

Configurez des écouteurs supplémentaires sur un Application Load Balancer ou un Network Load Balancer.

### Note

Pour un Application Load Balancer partagé, vous ne pouvez spécifier que l'option `Rule`. Les autres options ne s'appliquent pas aux équilibriseurs de charge partagés.

Espace de nom : **aws:elbv2:listener:*listener\_port***

Name (Nom)	Description	Par défaut	Valeurs valides
DefaultProcess	Nom du <a href="#">processus (p. 701)</a> vers lequel transférer le trafic lorsqu'aucune règle ne correspond à la demande.	default	Un nom de processus.
ListenerEnabled	Définissez la valeur sur <code>false</code> pour désactiver l'écouteur. Vous pouvez utiliser cette option pour désactiver l'écouteur par défaut sur le port 80.	true	true  false
Protocol	Protocole du trafic à traiter.	Avec un équilibrEUR de charge d'application :  <code>HTTP</code>  Avec un équilibrEUR de charge réseau :  <code>TCP</code>	Avec un équilibrEUR de charge d'application :  <code>HTTP, HTTPS</code>  Avec un équilibrEUR de charge réseau :  <code>TCP</code>
Rules	Liste des <a href="#">règles (p. 721)</a> à appliquer à l'écouteur  Cette option est applicable uniquement aux environnements avec un Application Load Balancer.  Si votre environnement utilise un Application Load Balancer partagé et que vous ne spécifiez pas cette option pour un processus d'écoute, Elastic Beanstalk associe automatiquement la	Aucune	Liste séparée par des virgules de noms de règles.

Name (Nom)	Description	Par défaut	Valeurs valides
	règle <code>default</code> à un écouteur de port 80.		
SSLCertificateArns	<p>Amazon Resource Name (ARN) du certificat SSL à lier à l'écouteur.</p> <p>Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge Application Load Balancer.</p>	Aucune	L'ARN d'un certificat stocké dans IAM ou ACM.
SSLPolicy	<p>Spécifiez une stratégie de sécurité à appliquer à l'écouteur.</p> <p>Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge Application Load Balancer.</p>	Aucune (valeur par défaut ELB)	Le nom d'une stratégie de sécurité d'équilibrEUR de charge.

## aws:elbv2:listenerrule:rule\_name

Définissez les règles d'écouteur pour un Application Load Balancer. Si une requête correspond aux noms d'hôte ou aux chemins dans une règle, l'équilibrEUR de charge les transfère vers le processus en question. Pour utiliser une règle, ajoutez-la à un écouteur avec l'option `Rules` dans le namespace [aws:elbv2:listener:listener\\_port](#) (p. 720).

### Note

Cet espace de noms n'est pas applicable aux environnements avec un équilibrEUR de charge réseau.

Espace de nom : **aws:elbv2:listenerrule:rule\_name**

Name (Nom)	Description	Par défaut	Valeurs valides
HostHeaders	Liste de noms d'hôte à faire correspondre. Par exemple, <code>my.example.com</code> .	ÉquilibrEUR de charge dédié : Aucun ÉquilibrEUR de charge partagé : CNAME de l'environnement	Chaque nom peut contenir jusqu'à 128 caractères. Un modèle peut inclure à la fois des lettres majuscules et minuscules, des chiffres, des traits d'union (-) et jusqu'à trois caractères génériques (* correspond à zéro ou plus ; ? correspond exactement à un

Name (Nom)	Description	Par défaut	Valeurs valides
			<p>caractère). Vous pouvez répertorier plusieurs noms, tous séparés par une virgule. L'Application Load Balancer prend en charge jusqu'à cinq règles <code>HostHeader</code> et <code>PathPattern</code> combinées.</p> <p>Pour plus d'informations, consultez <a href="#">Conditions de l'hôte</a> dans le Guide de l'utilisateur des Application Load Balancers.</p>
PathPatterns	<p>Modèles de chemin d'accès à faire correspondre (par exemple, <code>/img/*</code>).</p> <p>Cette option est applicable uniquement aux environnements avec un équilibrEUR de charge d'application.</p>	Aucune	<p>Chaque modèle peut contenir jusqu'à 128 caractères. Un modèle peut inclure des lettres majuscules et minuscules, des chiffres, des tirets (-) et jusqu'à trois caractères génériques (* correspond à zéro ou plus ; ? correspond exactement à un caractère). Vous pouvez ajouter plusieurs modèles de chemin d'accès séparés par des virgules. L'Application Load Balancer prend en charge jusqu'à cinq règles <code>HostHeader</code> et <code>PathPattern</code> combinées.</p> <p>Pour plus d'informations, consultez <a href="#">Conditions de chemin d'accès</a> dans le Guide de l'utilisateur des Application Load Balancers.</p>

Name (Nom)	Description	Par défaut	Valeurs valides
Priority	<p>Priorité de cette règle lorsque plusieurs règles correspondent. Le nombre inférieur est prioritaire. Deux règles ne peuvent pas avoir la même priorité.</p> <p>Avec un équilibrage de charge partagé, Elastic Beanstalk traite les priorités des règles comme relatives dans les environnements de partage et les mappe aux priorités absolues lors de la création.</p>	1	1 sur 1000
Process	Nom du <a href="#">processus (p. 701)</a> vers lequel transférer le trafic lorsque cette règle correspond à la demande.	default	Un nom de processus.

## aws:elbv2:loadbalancer

Configurez un Application Load Balancer.

Pour un équilibrage de charge partagé, seules les options `SharedLoadBalancer` et `SecurityGroups` sont valides.

### Note

Cet espace de noms n'est pas applicable aux environnements avec un Network Load Balancer.

Espace de nom : **aws:elbv2:loadbalancer**

Name (Nom)	Description	Par défaut	Valeurs valides
AccessLogsS3Bucket	Compartiment Amazon S3 où les journaux d'accès sont stockés. Le compartiment doit se trouver dans la même région que l'environnement et autoriser l'accès en écriture à l'équilibrage de charge.	Aucune	Un nom de compartiment.
AccessLogsS3Enabled	Activez le stockage des journaux d'accès.	false	true false
AccessLogsS3Prefix	Préfixe à ajouter aux noms des journaux d'accès. Par défaut, l'équilibrage de charge place les journaux dans un répertoire nommé AWSLogs dans le compartiment que vous spécifiez. Spécifiez un préfixe pour placer le répertoire AWSLogs dans un autre répertoire.	Aucune	
IdleTimeout	Délai d'attente, en secondes, de la fin d'une demande avant de fermer les connexions au client et à l'instance.	Aucune	1 sur 3600

Name (Nom)	Description	Par défaut	Valeurs valides
ManagedSecurityGroup	<p>Attribuez un groupe de sécurité existant à l'équilibrEUR de charge de votre environnement au lieu d'en créer un. Pour utiliser ce paramètre, mettez à jour le paramètre <b>SecurityGroups</b> dans cet espace de noms afin d'inclure l'ID de votre groupe de sécurité et de supprimer l'ID du groupe de sécurité créé automatiquement, le cas échéant.</p> <p>Pour autoriser le trafic de l'équilibrEUR de charge vers les instances EC2 de votre environnement, Elastic Beanstalk ajoute une règle au groupe de sécurité des instances, qui autorise le trafic entrant depuis le groupe de sécurité géré.</p>	Le groupe de sécurité qu'Elastic Beanstalk crée pour votre équilibrEUR de charge.	Un ID de groupe de sécurité.
SecurityGroups	<p>Liste des groupes de sécurité à lier à l'équilibrEUR de charge.</p> <p>Pour un équilibrEUR de charge partagé, si vous ne spécifiez pas cette valeur, Elastic Beanstalk vérifie si un groupe de sécurité existant qu'il gère est déjà attaché à l'équilibrEUR de charge. Si aucun groupe n'est lié à l'équilibrEUR de charge, Elastic Beanstalk crée un groupe de sécurité et le lie à l'équilibrEUR de charge. Elastic Beanstalk supprime ce groupe de sécurité lorsque le dernier environnement partageant l'équilibrEUR de charge s'arrête.</p> <p>Les groupes de sécurité de l'équilibrEUR de charge sont utilisés pour configurer la règle d'entrée du groupe de sécurité de l'instance Amazon EC2.</p>	Le groupe de sécurité qu'Elastic Beanstalk crée pour votre équilibrEUR de charge.	Liste, séparée par des virgules, des identifiants de groupes de sécurité.

Name (Nom)	Description	Par défaut	Valeurs valides
SharedLoadBalancer	<p>Amazon Resource Name (ARN) de l'équilibreur de charge partagé. Cette option s'applique uniquement à un Application Load Balancer. Elle est requise lorsque l'option <code>LoadBalancerIsShared</code> de l'espace de noms <a href="#">aws:elasticbeanstalk:environment (p. 700)</a> est définie sur <code>true</code>. Vous ne pouvez pas modifier l'ARN de l'équilibreur de charge partagé après la création de l'environnement.</p> <p>Critères d'une valeur valide :</p> <ul style="list-style-type: none"> <li>Il doit s'agir d'un équilibreur de charge actif valide dans la région AWS où se trouve l'environnement.</li> <li>Il doit se trouver dans le même Amazon Virtual Private Cloud (Amazon VPC) que l'environnement.</li> <li>Il ne peut pas s'agir d'un équilibreur de charge créé par Elastic Beanstalk en tant qu'équilibreur de charge dédié pour un autre environnement. Vous pouvez identifier ces équilibreurs de charge dédiés par le préfixe <code>awseb-</code>.</li> </ul> <p>Exemple :</p> <pre>arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/FrontEndLB/0dbf78d8ad96abbc</pre>	Aucune	ARN d'un équilibreur de charge valide répondant aux critères décrits ici.

## aws:rds:dbinstance

Configurez une instance de base de données Amazon RDS attachée.

Espace de nom : **aws:rds:dbinstance**

Name (Nom)	Description	Par défaut	Valeurs valides
DBAllocatedStorage	taille de stockage de base de données allouée, spécifiée en giga-octets.	MySQL: 5 Oracle : 10  sqlserver-se: 200  sqlserver-ex: 30	MySQL: 5-1024  Oracle: 10-1024  sqlserver: ne peut pas être modifié

Name (Nom)	Description	Par défaut	Valeurs valides
		sqlserver-web: 30	
DBDeletionPolicy	<p>Spécifie s'il faut conserver, supprimer ou créer un instantané de l'instance de base de données lors de la résiliation d'un environnement.</p> <p>Cette option fonctionne conjointement avec HasCoupledDatabase, qui est également une option de cet espace de noms.</p> <p><b>Warning</b></p> <p>La suppression d'une instance DB entraîne une perte permanente de données.</p>	Delete	Delete Retain Snapshot
DBEngine	Le nom du moteur de base de données à utiliser pour cette instance.	mysql	mysql oracle-se1 sqlserver-ex sqlserver-web sqlserver-se postgres
DBEngineVersion	Le numéro de version du moteur de base de données.	5.5	
DBInstanceClass	Le type de l'instance de base de données	db.t2.micro (db.m1.large pour un environnement qui ne s'exécute pas dans un VPC Amazon)	Pour plus d'informations, consultez la section <a href="#">Classes d'instances de bases de données</a> dans le Guide de l'utilisateur Amazon Relational Database Service.
DBPassword	Le nom du mot de passe utilisateur principal pour l'instance de base de données.	Aucune	
DBSnapshotIdentifier	L'identificateur du Snapshot DB d'origine à partir duquel effectuer la restauration.	Aucune	
DBUser	Le nom de l'utilisateur principal pour l'instance DB.	eboot	

Name (Nom)	Description	Par défaut	Valeurs valides
HasCoupledDatabase	<p>Spécifie si une instance de base de données est couplée à votre environnement. Si elle bascule sur <code>true</code>, Elastic Beanstalk crée une nouvelle instance de base de données couplée à votre environnement. Si elle bascule sur <code>false</code>, Elastic Beanstalk initie le découplage de l'instance de base de données de votre environnement.</p> <p>Cette option fonctionne conjointement avec <code>DBDeletionPolicy</code>, qui est également une option de cet espace de noms.</p> <p><b>Note</b></p> <p>Remarque : si vous basculez cette valeur sur <code>true</code> après avoir découpé la base de données précédente, Elastic Beanstalk crée une nouvelle base de données avec les paramètres d'option de base de données précédents. Toutefois, pour maintenir la sécurité de votre environnement, il ne conserve pas les paramètres <code>DBUser</code> et <code>DBPassword</code> existants. Vous devez à nouveau spécifier <code>DBUser</code> et <code>DBPassword</code>.</p>	<code>false</code>	<code>true</code> <code>false</code>
MultiAZDatabase	Spécifie si un déploiement multi-AZ d'une instance de base de données doit être créé. Pour plus d'informations sur les déploiements multi-AZ avec Amazon Relational Database Service (RDS), consultez <a href="#">Régions et zones de disponibilité</a> dans le Guide de l'utilisateur Amazon Relational Database Service.	<code>false</code>	<code>true</code> <code>false</code>

## Options spécifiques à une plateforme

Certaines plateformes Elastic Beanstalk définissent des espaces de noms d'options spécifiques à la plateforme. Ces espaces de noms et leurs options sont répertoriés ci-dessous pour chaque plate-forme.

### Note

Auparavant, dans les versions de plateforme basées sur l'AMI Amazon Linux (antérieure à Amazon Linux 2), les deux fonctionnalités suivantes et leurs espaces de noms respectifs étaient considérés comme des fonctionnalités spécifiques à la plateforme et étaient répertoriés ici par plateforme :

- Configuration du proxy pour les fichiers statiques – [`aws:elasticbeanstalk:environment:proxy:staticfiles` \(p. 706\)](#)
- AWS X-Ray Prise en charge – [`aws:elasticbeanstalk:xray` \(p. 712\)](#)

Dans les versions de plateforme Amazon Linux 2, Elastic Beanstalk implémente ces fonctionnalités de manière cohérente sur toutes les plateformes de prise en charge.

L'espace de noms associé est désormais répertorié dans la page [the section called “Options générales” \(p. 679\)](#). Nous n'en avons gardé mention sur cette page que pour les plates-formes qui avaient des espaces de noms différents.

#### Plates-formes

- [Options de la plateforme Docker \(p. 728\)](#)
- [Options de la plateforme Go \(p. 728\)](#)
- [Options de la plateforme Java SE \(p. 729\)](#)
- [Java avec options de plateforme Tomcat \(p. 729\)](#)
- [Historique de la plateforme .NET Core sous Linux \(p. 731\)](#)
- [Options de la plateforme .NET \(p. 731\)](#)
- [Options de plateforme Node.js \(p. 731\)](#)
- [Options de la plateforme PHP \(p. 733\)](#)
- [Options de la plateforme Python \(p. 734\)](#)
- [Options de la plateforme Ruby \(p. 735\)](#)

## Options de la plateforme Docker

Les options de configuration suivantes spécifiques à Docker s'appliquent aux plateformes Docker et Docker préconfigurées.

#### Note

Ces options de configuration ne s'appliquent pas à

- la plateforme Docker (Amazon Linux 2) avec Docker Compose
- la plateforme Multicontainer Docker (AMI Amazon Linux)

Espace de noms : **aws:elasticbeanstalk:environment:proxy**

Name (Nom)	Description	Par défaut	Valeurs valides
ProxyServer	Spécifie le serveur web à utiliser comme proxy.	nginx	nginx  none – Amazon Linux AM et Docker avec DC uniquement

## Options de la plateforme Go

### Options de plateforme AMI Amazon Linux (pré-Amazon Linux 2)

Espace de noms : **aws:elasticbeanstalk:container:golang:staticfiles**

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

Name (Nom)	Valeur
Chemin d'accès où le serveur proxy va servir les fichiers.  Exemple : /images pour traiter les fichiers à l'emplacement <b>subdomain.eleasticbeanstalk.com/images</b> .	Nom du dossier contenant les fichiers.  Exemple : staticimages pour traiter les fichiers depuis un dossier nommé staticimages au niveau supérieur de votre bundle de fichiers source.

## Options de la plateforme Java SE

### Options de plateforme AMI Amazon Linux (pré-Amazon Linux 2)

Espace de noms : **aws:elasticbeanstalk:container:java:staticfiles**

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

Name (Nom)	Valeur
Chemin d'accès où le serveur proxy va servir les fichiers.  Exemple : /images pour traiter les fichiers à l'emplacement <b>subdomain.eleasticbeanstalk.com/images</b> .	Nom du dossier contenant les fichiers.  Exemple : staticimages pour traiter les fichiers depuis un dossier nommé staticimages au niveau supérieur de votre bundle de fichiers source.

## Java avec options de plateforme Tomcat

Espace de noms : **aws:elasticbeanstalk:application:environment**

Name (Nom)	Description	Par défaut	Valeurs valides
JDBC_CONNECTION_STRING	Chaîne de connexion à une base de données externe.	Non applicable	Non applicable

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

Espace de noms : **aws:elasticbeanstalk:container:tomcat:jvmoptions**

Name (Nom)	Description	Par défaut	Valeurs valides
JVM Options	Passez des options de ligne de commande à la JVM au démarrage.	Non applicable	Non applicable

Name (Nom)	Description	Par défaut	Valeurs valides
Xmx	Tailles maximum des segments de mémoire de la JVM.	256m	Non applicable
XX:MaxPermSize	<p>Section du segment de mémoire de la JVM qui est utilisée pour stocker des définitions de classe et des métadonnées associées.</p> <p><b>Note</b></p> <p>Cette option s'applique uniquement aux versions Java antérieures à Java 8 et n'est pas prise en charge sur les plateformes Elastic Beanstalk Tomcat basées sur Amazon Linux 2.</p>	64m	Non applicable
Xms	Tailles initiales du segment de mémoire de la JVM.	256m	Non applicable
<i>optionName</i>	Spécifiez les options JVM arbitraires en plus de celles définies par la plate-forme Tomcat.	Non applicable	Non applicable

Espace de noms : **aws:elasticbeanstalk:environment:proxy**

Name (Nom)	Description	Par défaut	Valeurs valides
GzipCompression	<p>Définissez la valeur sur <code>false</code> pour désactiver la compression des réponses.</p> <p>Valable uniquement sur les versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux).</p>	true	true false
ProxyServer	<p>Définissez le proxy à utiliser sur les instances de votre environnement. Si vous définissez cette option sur <code>apache</code>, Elastic Beanstalk utilise <a href="#">Apache 2.4</a>.</p> <p>Définissez cette option sur <code>apache/2.2</code> si votre application n'est pas prête à migrer à partir d'<a href="#">Apache 2.2</a> en raison des paramètres de configuration de proxy incompatibles. Cette valeur n'est valide que sur les versions de plateforme AMI Amazon Linux (antérieure à Amazon Linux).</p> <p>Définissez cette option sur <code>nginx</code> pour utiliser <a href="#">nginx</a>. Il s'agit de la version par défaut de la plateforme Amazon Linux 2.</p> <p>Pour plus d'informations, consultez <a href="#">Configuration du serveur proxy de votre environnement Tomcat (p. 126)</a>.</p>	nginx (Amazon Linux 2)  apache (AMI Amazon Linux)	apache  apache/2.2 – AMI Amazon Linux uniquement  nginx

## Historique de la plateforme .NET Core sous Linux

Espace de noms : **aws:elasticbeanstalk:environment:proxy**

Name (Nom)	Description	Par défaut	Valeurs valides
ProxyServer	Spécifie le serveur web à utiliser comme proxy.	nginx	nginx none

## Options de la plateforme .NET

Espace de nom : **aws:elasticbeanstalk:container:dotnet:apppool**

Name (Nom)	Description	Par défaut	Valeurs valides
Target Runtime	Choisissez la version .NET Framework pour votre application.	4.0	2.0 4.0
Enable 32-bit Applications	Réglez le paramètre sur <code>True</code> pour exécuter des applications 32 bits.	False	True False

## Options de plateforme Node.js

Espace de noms : **aws:elasticbeanstalk:environment:proxy**

Name (Nom)	Description	Par défaut	Valeurs valides
ProxyServer	Définissez le proxy à utiliser sur les instances de votre environnement.	nginx	apache nginx

## Options de plateforme AMI Amazon Linux (pré-Amazon Linux 2)

Espace de nom : **aws:elasticbeanstalk:container:nodejs**

Name (Nom)	Description	Par défaut	Valeurs valides
NodeCommand	Commande utilisée pour démarrer l'application Node.js. Si une chaîne vide est spécifiée, <code>app.js</code> est utilisé, puis <code>server.js</code> , et enfin <code>npm start</code> dans cet ordre.	""	Non applicable
NodeVersion	Version de Node.js. Par exemple, <code>4.4.6</code>  Les versions Node.js prises en charge varient selon les versions de plateforme Node.js. Pour obtenir la liste des versions prises en	varie	varie

Name (Nom)	Description	Par défaut	Valeurs valides
	<p>charge, accédez à <a href="#">Node.js</a> dans le document Plateformes AWS Elastic Beanstalk.</p> <p><b>Note</b></p> <p>Lorsque la prise en charge de la version de Node.js que vous utilisez est supprimée de la plateforme, vous devez modifier ou supprimer le paramètre de version avant de procéder à une <a href="#">mise à jour de la plateforme (p. 496)</a>. Cela peut se produire lorsqu'une faille de sécurité est identifiée pour une ou plusieurs versions de Node.js</p> <p>Lorsque cela se produit, toute tentative de mise à jour vers une nouvelle version de la plateforme qui ne prend pas en charge le paramètre <a href="#">NodeVersion (p. 731)</a> configuré échoue. Pour éviter d'avoir besoin de créer un nouvel environnement, remplacez l'option de configuration NodeVersion par une version de Node.js qui est prise en charge à la fois par l'ancienne version de plateforme et par la nouvelle, ou bien <a href="#">supprimez le paramètre de l'option (p. 671)</a>, puis effectuez la mise à jour de la plateforme.</p>		
GzipCompression	Spécifie si la compression gzip est activée. Si le serveur proxy est défini sur none, la compression gzip est désactivée.	false	true false
ProxyServer	Spécifie le serveur web qui doit être utilisé pour les connexions proxy à Node.js. Si ProxyServer est défini sur none, les mappages de fichiers statiques ne prennent pas effet et la compression gzip est désactivée.	nginx	apache nginx none

#### Espace de nom : `aws:elasticbeanstalk:container:nodejs:staticfiles`

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

##### Note

Les paramètres de fichiers statiques ne s'appliquent pas si `aws:elasticbeanstalk:container:nodejs::ProxyFiles` est défini sur none.

Name (Nom)	Valeur
Chemin d'accès où le serveur proxy va servir les fichiers.  Exemple : /images pour traiter les fichiers à l'emplacement <b>subdomain.eleasticbeanstalk.com/images</b> .	Nom du dossier contenant les fichiers.  Exemple : staticimages pour traiter les fichiers depuis un dossier nommé staticimages au niveau supérieur de votre bundle de fichiers source.

## Options de la plateforme PHP

Espace de nom : **aws:elasticbeanstalk:container:php:phpini**

Name (Nom)	Description	Par défaut	Valeurs valides
document_root	Spécifiez le répertoire enfant de votre projet qui est traité comme la racine web destinée au public.	/	Une chaîne vide est traitée comme /, ou spécifiez une chaîne commençant par /
memory_limit	Volume de mémoire alloué à l'environnement PHP.	256M	Non applicable
zlib.output_compression	Spécifie si PHP doit utiliser ou non la compression pour la sortie.	Off	On Off true false
allow_url_fopen	Spécifie si les fonctions du fichier PHP sont autorisées à extraire des données des sites distants, comme les sites web ou les serveurs FTP.	On	On Off true false
display_errors	Spécifie si les messages d'erreur doivent faire partie de la sortie.	Off	On Off
max_execution_time	Définit la durée maximale, en secondes, durant laquelle un script est autorisé à s'exécuter avant qu'il soit résilié par l'environnement.	60	0 à 9223372036854775807 (PHP_INT_MAX)
composer_options	Définit des options personnalisées à utiliser lors de l'installation de dépendances à l'aide de Composer via composer.phar install. Pour plus d'informations et pour connaître les options disponibles, consultez <a href="http://getcomposer.org/doc/03-cli.md#install">http://getcomposer.org/doc/03-cli.md#install</a> .	Non applicable	Non applicable

Espace de noms : **aws:elasticbeanstalk:environment:proxy**

Name (Nom)	Description	Par défaut	Valeurs valides
ProxyServer	Définissez le proxy à utiliser sur les instances de votre environnement.	nginx	apache nginx

Note

Pour plus d'informations sur la plateforme PHP, consultez [Utilisation de la plateforme PHP Elastic Beanstalk \(p. 293\)](#).

## Options de la plateforme Python

Espace de noms : **aws:elasticbeanstalk:application:environment**

Name (Nom)	Description	Par défaut	Valeurs valides
DJANGO_SETTINGS_MODULE	Spécifie le fichier de paramètres à utiliser.	Non applicable	Non applicable

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

Espace de noms : **aws:elasticbeanstalk:container:python**

Name (Nom)	Description	Par défaut	Valeurs valides
WSGIPath	Le fichier qui contient l'application WSGI. Ce fichier doit avoir une <code>application</code> joignable.	Sur les versions de plateforme Python Amazon Linux 2 : <code>application</code>  Sur les versions de la plateforme AMI Python Amazon Linux : <code>application.py</code>	Non applicable
NumProcesses	Le nombre de processus de démon qui doit être démarré pour le groupe de processus lorsque vous exécutez des applications WSGI.	1	Non applicable
NumThreads	Le nombre de threads à créer pour gérer des demandes dans chaque processus de démon au sein du groupe de processus lorsque vous exécutez des applications WSGI.	15	Non applicable

Espace de noms : **aws:elasticbeanstalk:environment:proxy**

Name (Nom)	Description	Par défaut	Valeurs valides
ProxyServer	Définissez le proxy à utiliser sur les instances de votre environnement.	nginx	apache nginx

Options de plateforme AMI Amazon Linux (pré-Amazon Linux 2)

Espace de noms : **aws:elasticbeanstalk:container:python:staticfiles**

Vous pouvez utiliser l'espace de noms suivant pour configurer le serveur proxy afin de servir des fichiers statiques. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application. Cela réduit le nombre de demandes que votre application doit traiter.

Mappez un chemin servi par le serveur proxy à un dossier dans le code source qui contient les ressources statiques. Chaque option que vous définissez dans cet espace de noms mappe un chemin d'accès différent.

Par défaut, le serveur proxy d'un environnement Python sert tous les fichiers dans un dossier nommé `static` sur le chemin d'accès `/static`.

Espace de noms : **aws:elasticbeanstalk:container:python:staticfiles**

Name (Nom)	Valeur
Chemin d'accès où le serveur proxy va servir les fichiers.  Exemple : <code>/images</code> pour traiter les fichiers à l'emplacement <code>subdomain.eleasticbeanstalk.com/images</code> .	Nom du dossier contenant les fichiers.  Exemple : <code>staticimages</code> pour traiter les fichiers depuis un dossier nommé <code>staticimages</code> au niveau supérieur de votre bundle de fichiers source.

## Options de la plateforme Ruby

Espace de noms : **aws:elasticbeanstalk:application:environment**

Name (Nom)	Description	Par défaut	Valeurs valides
RAILS_SKIP_MIGRATIONS	Spécifie si il convient d'exécuter <code>rake db:migrate</code> au nom d'applications utilisateurs ; ou si cette valeur doit être ignorée. Ceci n'est applicable qu'aux applications Rails 3.	false	true false
RAILS_SKIP_ASSET_COMPILATION	Spécifie si le conteneur doit exécuter <code>rake assets:precompile</code> au nom des applications utilisateurs ; ou si cette valeur doit être ignorée. Cela est également applicable uniquement aux applications Rails 3.	false	true false

Name (Nom)	Description	Par défaut	Valeurs valides
BUNDLE_WITHOUT	Une liste séparée par deux points (:) de groupes à ignorer lors de l'installation de dépendances à partir d'un Gemfile.	test:development	Non applicable
RACK_ENV	Spécifie dans quelle étape de l'environnement une application peut être exécutée. Des exemples d'environnements courants incluent le développement, la production, le test.	production	Non applicable

Pour plus d'informations, consultez [Propriétés de l'environnement et autres paramètres de logiciel \(p. 632\)](#).

## Options personnalisées

Utilisez l'espace de noms `aws:elasticbeanstalk:customoption` pour définir des options et des valeurs qui peuvent être lues dans les blocs Resources des autres fichiers de configuration. Utilisez des options personnalisées pour collecter les paramètres spécifiés par les utilisateurs dans un fichier de configuration unique.

Par exemple, vous disposez peut-être d'un fichier de configuration complexe qui définit une ressource pouvant être configurée par l'utilisateur qui lance l'environnement. Si vous utilisez `Fn::GetOptionSetting` pour récupérer la valeur d'une option personnalisée, vous pouvez placer la définition de cette option dans un autre fichier de configuration, où l'utilisateur pourra plus facilement la trouver et la modifier.

En outre, comme il s'agit d'options de configuration, les options personnalisées peuvent être définies au niveau de l'API pour remplacer les valeurs définies dans un fichier de configuration. Pour plus d'informations, consultez la section [Priorité \(p. 659\)](#).

Les options personnalisées sont définies comme toute autre option :

```
option_settings:
  aws:elasticbeanstalk:customoption:
    option name: option value
```

Par exemple, le fichier de configuration suivant crée une option nommée `ELBAlarmEmail` et lui attribue la valeur `someone@example.com` :

```
option_settings:
  aws:elasticbeanstalk:customoption:
    ELBAlarmEmail: someone@example.com
```

Autre exemple : un fichier de configuration peut définir une rubrique SNS qui lit l'option avec `Fn::GetOptionSetting` pour renseigner la valeur de l'attribut `Endpoint` :

```
Resources:
  MySNSTopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint:
            Fn::GetOptionSetting:
              OptionName: ELBAlarmEmail
              DefaultValue: nobody@example.com
```

Protocol: email

Vous trouverez d'autres exemples d'extraits utilisant la fonction Fn::GetOptionSetting sur la page [Ajout et personnalisation des ressources de l'environnement Elastic Beanstalk \(p. 759\)](#).

## Personnalisation d'environnement avancée avec fichiers de configuration (.ebextensions)

Vous pouvez ajouter des fichiers de configuration AWS Elastic Beanstalk (.ebextensions) au code source de votre application web pour configurer votre environnement et personnaliser les ressources AWS qu'il contient. Les fichiers de configuration sont des documents au format YAML, or JSON, avec une extension de fichier .config que vous placez dans un dossier nommé .ebextensions et que vous déployez dans le bundle de fichiers source de votre application.

Example .ebextensions/network-load-balancer.config

Dans cet exemple on effectue une modification de configuration simple. Une option de configuration est modifiée de sorte à définir le type de l'équilibreur de charge de votre environnement sur Network Load Balancer.

```
option_settings:
  aws:elasticbeanstalk:environment:
    LoadBalancerType: network
```

Nous vous recommandons d'utiliser un format YAML pour vos fichiers de configuration, car il est plus lisible que JSON. YAML prend en charge des commentaires, des commandes sur plusieurs lignes, plusieurs alternatives pour l'utilisation de guillemets, et bien plus encore. Toutefois, vous pouvez effectuer une modification de configuration dans les fichiers de configuration Elastic Beanstalk comme avec YAML ou JSON.

### Conseil

Lorsque vous développez ou testez de nouveaux fichiers de configuration, lancez un environnement propre qui exécute l'application par défaut et déployez les fichiers dans cet environnement. Des fichiers de configuration au format erroné empêcheront irrémédiablement le démarrage d'un nouvel environnement.

La section `option_settings` d'un fichier de configuration définit des valeurs pour des [options de configuration \(p. 658\)](#). Les options de configuration vous permettent de configurer votre environnement Elastic Beanstalk, les ressources AWS qu'il contient et le logiciel qui exécute votre application. Les fichiers de configuration sont seulement une parmi de multiples façons de définir les options de configuration.

La section [Resources \(p. 759\)](#) vous permet de personnaliser davantage les ressources dans l'environnement de votre application et de définir des ressources AWS supplémentaires au-delà de la fonctionnalité fournie par les options de configuration. Vous pouvez ajouter et configurer toutes les ressources prises en charge par AWS CloudFormation, qui sont utilisées par Elastic Beanstalk pour créer des environnements.

Les autres sections d'un fichier de configuration (`packages`, `sources`, `files`, `users`, `groups`, `commands`, `container_commands` et `services`) vous permettent de configurer les instances EC2 qui sont lancées dans votre environnement. Chaque fois qu'un serveur est démarré dans votre environnement, Elastic Beanstalk exécute les opérations définies dans ces sections pour préparer le système d'exploitation et le système de stockage pour votre application.

Pour obtenir des exemples de fichiers .ebextensions couramment utilisés, veuillez consulter le [référentiel de fichiers de configuration Elastic Beanstalk](#).

## Prérequis

- Emplacement – Placez tous vos fichiers de configuration dans un même dossier nommé `.ebextensions`, à la racine de votre bundle de fichiers source. Comme les dossiers commençant par un point peuvent être masqués par des navigateurs de fichiers, assurez-vous que le dossier est ajouté lorsque vous créez votre bundle de fichiers source. Pour obtenir des instructions, veuillez consulter [Création d'une solution groupée d'application \(p. 415\)](#).
- Attribution d'un nom – Les fichiers de configuration doivent porter l'extension de fichier `.config`.
- Mise en forme – Les fichiers de configuration doivent être conformes aux spécifications YAML ou JSON.

Lorsque vous utilisez YAML, utilisez des espaces pour mettre retrait les clés à différents niveaux d'imbrication. Pour plus d'informations sur YAML, consultez [YAML Ain't Markup Language \(YAML™\) Version 1.1](#).

- Unicité – Utilisez chaque clé une seule fois dans chaque fichier de configuration.

### Avertissement

Si vous utilisez une clé (par exemple, `option_settings`) deux fois dans le même fichier de configuration, l'une des sections sera supprimée. Combinez les sections dupliquées en une seule section, ou placez-les dans des fichiers de configuration distincts.

Le processus de déploiement varie légèrement selon le client que vous utilisez pour gérer vos environnements. Consultez les sections suivantes pour obtenir des détails :

- [Console Elastic Beanstalk \(p. 666\)](#)
- [INTERFACE DE LIGNE DE COMMANDE EB \(p. 668\)](#)
- [AWS CLI \(p. 669\)](#)

## Rubriques

- [Paramètres d'option \(p. 738\)](#)
- [Personnalisation du logiciel sur des serveurs Linux \(p. 740\)](#)
- [Personnalisation du logiciel sur des serveurs Windows \(p. 753\)](#)
- [Ajout et personnalisation des ressources de l'environnement Elastic Beanstalk \(p. 759\)](#)

# Paramètres d'option

Vous pouvez utiliser la clé `option_settings` pour modifier la configuration d'Elastic Beanstalk et définir des variables qui peuvent être récupérées depuis votre application à l'aide des variables d'environnement. Certains espaces de noms vous permettent d'étendre le nombre de paramètres et spécifient les noms de paramètres. Pour une liste des options de configuration et des espaces de noms, consultez [Options de configuration \(p. 658\)](#).

Les paramètres d'option peuvent également être appliqués directement dans un environnement au cours de la création de l'environnement ou d'une mise à jour de l'environnement. Les paramètres appliqués directement à l'environnement remplacent ceux pour les mêmes options dans les fichiers de configuration. Si vous supprimez des paramètres de la configuration d'un environnement, les paramètres dans les fichiers de configuration entrent en vigueur. Pour de plus amples informations, veuillez consulter [Précedence \(p. 659\)](#).

## Syntaxe

La syntaxe standard pour les paramètres d'option est un ensemble d'objets, chacun ayant un `namespace`, `option_name` et une clé `value`.

```
option_settings:  
  - namespace: namespace  
    option_name: option name  
    value: option value  
  - namespace: namespace  
    option_name: option name  
    value: option value
```

La clé `namespace` est facultative. Si vous ne spécifiez pas un espace de noms, la valeur par défaut utilisée est `aws:elasticbeanstalk:application:environment` :

```
option_settings:  
  - option_name: option name  
    value: option value  
  - option_name: option name  
    value: option value
```

Elastic Beanstalk prend également en charge une syntaxe raccourcie pour les paramètres d'option qui vous permet de spécifier des options sous forme de paires clé-valeur sous l'espace de noms :

```
option_settings:  
  namespace:  
    option_name: option value  
    option_name: option value
```

## Exemples

Les exemples suivants définissent une option spécifique à la plateforme Tomcat dans l'espace de noms `aws:elasticbeanstalk:container:tomcat:jvmoptions` et une propriété d'environnement nommée `MYPARAMETER`.

Au format YAML standard :

Example `.ebextensions/options.config`

```
option_settings:  
  - namespace: aws:elasticbeanstalk:container:tomcat:jvmoptions  
    option_name: Xmx  
    value: 256m  
  - option_name: MYPARAMETER  
    value: parametervalue
```

Au format raccourci :

Example `.ebextensions/options.config`

```
option_settings:  
  aws:elasticbeanstalk:container:tomcat:jvmoptions:  
    Xmx: 256m  
  aws:elasticbeanstalk:application:environment:  
    MYPARAMETER: parametervalue
```

Dans JSON :

Example `.ebextensions/options.config`

```
{
```

```
"option_settings": [
  {
    "namespace": "aws:elasticbeanstalk:container:tomcat:jvmoptions",
    "option_name": "Xmx",
    "value": "256m"
  },
  {
    "option_name": "MYPARAMETER",
    "value": "parametervalue"
  }
]
```

## Personnalisation du logiciel sur des serveurs Linux

Vous pouvez souhaiter personnaliser et configurer le logiciel dont dépend votre application. Vous pouvez ajouter des commandes à exécuter pendant le provisionnement de l'instance, définir des utilisateurs et des groupes Linux et télécharger ou créer directement des fichiers sur vos instances d'environnement. Ces fichiers peuvent être des dépendances requises par l'application (par exemple, des packages supplémentaires provenant du référentiel yum) ou des fichiers de configuration en remplacement d'un fichier de configuration de proxy par exemple, pour remplacer des paramètres spécifiques qui sont définis par défaut par Elastic Beanstalk.

Cette section décrit le type d'informations que vous pouvez inclure dans un fichier de configuration afin de personnaliser le logiciel sur vos instances EC2 exécutant Linux. Pour obtenir des informations générales sur la personnalisation et la configuration de vos environnements Elastic Beanstalk, veuillez consulter [Configuration d'environnements Elastic Beanstalk \(p. 532\)](#). Pour plus d'informations sur la personnalisation des logiciels sur vos instances EC2 exécutant Windows, consultez [Personnalisation du logiciel sur des serveurs Windows \(p. 753\)](#).

### Remarques

- Sur les plateformes Amazon Linux 2, au lieu de fournir des fichiers et des commandes dans des fichiers de configuration .ebextensions, nous vous recommandons vivement d'utiliser Buildfile, Profile et les hooks de plateforme dès que possible, pour configurer et exécuter du code personnalisé sur vos instances d'environnement pendant le provisionnement d'instance. Pour plus d'informations sur ces mécanismes, consultez [the section called “Extension des plateformes Linux” \(p. 34\)](#).
- YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Les fichiers de configuration prennent en charge les clés suivantes qui affectent le serveur Linux sur lequel votre application s'exécute.

### Clés

- [Packages \(p. 741\)](#)
- [Groupes \(p. 742\)](#)
- [Utilisateurs \(p. 742\)](#)
- [Sources \(p. 743\)](#)
- [Dépôt de \(p. 744\)](#)
- [Commandes \(p. 746\)](#)
- [Services \(p. 747\)](#)
- [Commandes de conteneur \(p. 748\)](#)
- [Exemple : utilisation de métriques personnalisées Amazon CloudWatch \(p. 750\)](#)

Les clés sont traitées dans l'ordre dans lequel elles sont répertoriées ici.

Observez les [événements \(p. 879\)](#) de votre environnement pendant le développement et les tests des fichiers de configuration. Elastic Beanstalk ignore un fichier de configuration qui contient des erreurs de validation, comme une clé non valide, et ne traite aucune autre clé dans le même fichier. Lorsque cela se produit, Elastic Beanstalk ajoute un événement d'avertissement dans le journal des événements.

## Packages

Vous pouvez utiliser la clé `packages` pour télécharger et installer des applications et des composants prépackagés.

### Syntaxe

```
packages:  
  name of package manager:  
    package name: version  
    ...  
  name of package manager:  
    package name: version  
    ...  
  ...
```

Vous pouvez spécifier plusieurs packages sous chaque clé de gestionnaire de package.

### Formats de packages pris en charge

Actuellement, Elastic Beanstalk prend en charge les gestionnaires de package suivants : yum, rubygems, python et rpm. Les packages sont traités dans l'ordre suivant : rpm, yum, puis rubygems et python. Il n'y a pas de classement entre rubygems et python. Au sein de chaque gestionnaire de packages, l'ordre des packages d'installation n'est pas garanti. Utilisez un gestionnaire de package pris en charge par votre système d'exploitation.

#### Note

Elastic Beanstalk prend en charge deux gestionnaires de package sous-jacents pour Python : pip et easy\_install. Toutefois, dans la syntaxe du fichier de configuration, vous devez spécifier `python` comme nom du gestionnaire de package. Lorsque vous utilisez un fichier de configuration pour spécifier un gestionnaire de package Python, Elastic Beanstalk utilise Python 2.7. Si votre application repose sur une autre version de Python, vous pouvez indiquer que les packages doivent être installés dans un fichier `requirements.txt`. Pour de plus amples informations, veuillez consulter [Spécification des dépendances à l'aide d'un fichier Requirements \(p. 363\)](#).

### Spécification des versions

Au sein de chaque gestionnaire de package, chaque package est spécifié sous la forme d'un nom de package et d'une liste de versions. La version peut être une chaîne, une liste de versions, ou une chaîne ou une liste vide. Une chaîne ou une liste vide indique que vous souhaitez la dernière version. Pour le gestionnaire rpm, la version est spécifiée sous la forme d'un chemin d'accès à un fichier sur disque ou d'une URL. Les chemins relatifs ne sont pas pris en charge.

Si vous spécifiez une version d'un package, Elastic Beanstalk tente d'installer cette version, même si une version plus récente du package est déjà installée sur l'instance. Si une version plus récente est déjà installée, le déploiement échoue. Certains gestionnaires de package prennent en charge plusieurs versions, mais pas tous. Veuillez vérifier la documentation de votre gestionnaire de package pour plus d'informations. Si vous ne spécifiez aucune version et qu'une version du package est déjà installée, Elastic Beanstalk n'installe pas une nouvelle version, car il part du principe que vous souhaitez conserver et utiliser la version existante.

## Exemple d'extrait

L'extrait suivant spécifie une URL de la version pour rpm, demande la dernière version à yum et la version 0.10.2 de chef à rubygems.

```
packages:  
  yum:  
    libmemcached: []  
    ruby-devel: []  
    gcc: []  
  rpm:  
    epel: http://download.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm  
  rubygems:  
    chef: '0.10.2'
```

## Groupes

Vous pouvez utiliser la clé `groups` pour créer des groupes Linux/UNIX et pour attribuer des ID de groupe. Pour créer un groupe, ajoutez une nouvelle paire clé-valeur qui mappe un nouveau nom de groupe à un ID de groupe facultatif. La clé des groupes peut contenir un ou plusieurs noms de groupe. Le tableau suivant répertorie les clés disponibles.

### Syntaxe

```
groups:  
  name of group: {}  
  name of group:  
    gid: "group id"
```

### Options

#### gid

Numéro d'identification d'un groupe.

Si un ID de groupe est spécifié et que le nom du groupe existe déjà, la création du groupe échoue. Si un autre groupe est associé à l'ID de groupe spécifié, le système d'exploitation peut refuser la création du groupe.

## Exemple d'extrait

L'extrait suivant spécifie un groupe nommé `groupOne` sans attribuer d'ID de groupe, et un groupe nommé `groupTwo` ayant spécifié 45 comme valeur d'ID de groupe.

```
groups:  
  groupOne: {}  
  groupTwo:  
    gid: "45"
```

## Utilisateurs

Vous pouvez utiliser la clé `users` pour créer des utilisateurs Linux/UNIX sur l'instance EC2.

### Syntaxe

```
users:
```

```
name of user:  
groups:  
  - name of group  
uid: "id of the user"  
homeDir: "user's home directory"
```

## Options

**uid**

ID d'un utilisateur. Le processus de création échoue si le nom d'utilisateur existe avec un autre ID d'utilisateur. Si l'ID d'utilisateur est déjà affecté à un utilisateur existant, le système d'exploitation peut refuser la demande de création.

**groups**

Liste de noms de groupes. L'utilisateur est ajouté à chaque groupe de la liste.

**homeDir**

Répertoire de base de l'utilisateur.

Les utilisateurs sont créés en tant qu'utilisateurs du système non interactif avec le shell /sbin/nologin. Ce paramètre est intégré à la conception et ne peut pas être modifié.

## Exemple d'extrait

```
users:  
myuser:  
groups:  
  - group1  
  - group2  
uid: "50"  
homeDir: "/tmp"
```

## Sources

Vous pouvez utiliser la clé sources pour télécharger un fichier d'archives à partir d'une URL publique et le décompresser dans un répertoire cible de l'instance EC2.

## Syntaxe

```
sources:  
target directory: location of archive file
```

## Formats pris en charge

Les formats pris en charge sont les suivants : tar, tar+gzip, tar+bz2 et zip. Vous pouvez faire référence à des emplacements externes tels qu'Amazon Simple Storage Service (Amazon S3) (par exemple, <https://mybucket.s3.amazonaws.com/myobject>) tant que l'URL est publiquement accessible.

## Exemple d'extrait

L'exemple suivant télécharge un fichier .zip public à partir d'un compartiment Amazon S3 et le décomprime dans /etc/myapp:

```
sources:  
/etc/myapp: https://mybucket.s3.amazonaws.com/myobject
```

### Note

Plusieurs extractions ne doivent pas réutiliser le même chemin cible. L'extraction d'une autre source vers le même chemin cible remplacera le contenu au lieu de l'ajouter.

## Dépôt de

Vous pouvez utiliser la clé `files` pour créer des fichiers sur l'instance EC2. Le contenu peut être soit en ligne dans le fichier de configuration, ou bien le contenu peut être extrait d'une URL. Les fichiers sont écrits sur disque par ordre lexicographique.

Vous pouvez utiliser la clé `files` pour télécharger des fichiers privés depuis Amazon S3 en fournissant un profil d'instance pour l'autorisation.

Si le chemin d'accès au fichier que vous spécifiez existe déjà sur l'instance, le fichier existant est conservé avec l'extension `.bak` ajoutée à son nom.

### Syntaxe

```
files:  
  "target file location on disk":  
    mode: "six-digit octal value"  
    owner: name of owning user for file  
    group: name of owning group for file  
    source: URL  
    authentication: authentication name:  
  
  "target file location on disk":  
    mode: "six-digit octal value"  
    owner: name of owning user for file  
    group: name of owning group for file  
    content: |  
      # this is my  
      # file content  
    encoding: encoding format  
    authentication: authentication name:
```

### Options

#### content

Contenu de la chaîne à ajouter dans le fichier. Spécifiez `content` ou `source`, mais pas les deux.

#### source

URL du fichier à télécharger. Spécifiez `content` ou `source`, mais pas les deux.

#### encoding

Format d'encodage de la chaîne spécifiée via l'option `content`.

Valeurs valides : `plain` | `base64`

#### group

Groupe Linux auquel appartient le fichier.

#### owner

Utilisateur Linux auquel appartient le fichier.

#### mode

Valeur octale à six chiffres qui représente le mode de ce fichier. Non pris en charge pour les systèmes Windows. Utilisez les trois premiers chiffres pour les liens symboliques et les trois derniers chiffres

pour définir les autorisations. Pour créer un lien symbolique, spécifiez 120`xxx`, où `xxx` définit les autorisations du fichier cible. Pour définir des autorisations pour un fichier, utilisez les trois derniers chiffres, tels que 000644.

#### authentication

Le nom d'une [méthode d'authentification AWS CloudFormation](#) à utiliser. Vous pouvez ajouter des méthodes d'authentification aux métadonnées du groupe Auto Scaling via la clé Resources. Vous trouverez un exemple ci-dessous.

### Exemple d'extrait

```
files:
  "/home/ec2-user/myfile" :
    mode: "000755"
    owner: root
    group: root
    source: http://foo.bar/myfile

  "/home/ec2-user/myfile2" :
    mode: "000755"
    owner: root
    group: root
    content: |
      this is my
      file content
```

Exemple d'utilisation d'un lien symbolique. Cela crée un lien `/tmp/myfile2.txt` qui renvoie vers le fichier existant `/tmp/myfile1.txt`.

```
files:
  "/tmp/myfile2.txt" :
    mode: "120400"
    content: "/tmp/myfile1.txt"
```

L'exemple suivant utilise la clé Resources pour ajouter une méthode d'authentification nommée S3Auth et l'utilise pour télécharger un fichier privé à partir d'un compartiment Amazon S3 :

```
Resources:
  AWSEBAutoScalingGroup:
    Metadata:
      AWS::CloudFormation::Authentication:
        S3Auth:
          type: "s3"
          buckets: ["elasticbeanstalk-us-west-2-123456789012"]
          roleName:
            Fn::GetOptionSetting:
              Namespace: "aws:autoscaling:launchconfiguration"
              OptionName: "IamInstanceProfile"
              DefaultValue: "aws-elasticbeanstalk-ec2-role"

    files:
      "/tmp/data.json" :
        mode: "000755"
        owner: root
        group: root
        authentication: "S3Auth"
        source: https://elasticbeanstalk-us-west-2-123456789012.s3-us-west-2.amazonaws.com/
data.json
```

## Commandes

Vous pouvez utiliser la clé `commands` pour exécuter des commandes sur l'instance EC2. Les commandes exécutées avant l'application et le serveur web sont définies et le fichier de version de l'application est extrait.

Les commandes spécifiées s'exécutent en tant qu'utilisateur racine et sont traitées dans l'ordre alphabétique par nom. Commandes exécutées dans le répertoire racine (par défaut). Pour exécuter des commandes à partir d'un autre répertoire, utilisez l'option `cwd`.

Pour résoudre les problèmes associés à vos commandes, vous pouvez trouver leur sortie dans les [journaux d'instance \(p. 884\)](#).

### Syntaxe

```
commands:  
  command_name:  
    command: command to run  
    cwd: working directory  
    env:  
      variable name: variable value  
    test: conditions for command  
    ignoreErrors: true
```

### Options

#### command

Tableau ([collection de séquence de blocs](#) en syntaxe YAML) ou chaîne spécifiant la commande à exécuter. Remarques importantes :

- Si vous utilisez une chaîne, vous n'avez pas besoin de placer la totalité de la chaîne entre guillemets. Si vous utilisez des guillemets, utilisez des caractères d'échappement pour les occurrences littérales du même type de guillemets.
- Si vous utilisez un tableau, vous n'avez pas besoin de caractères d'espacement ou d'inclure des paramètres de commande entre guillemets. Chaque élément de tableau est un argument de commande unique. N'utilisez pas un tableau pour spécifier plusieurs commandes.

Les exemples suivants sont tous équivalents :

```
commands:  
  command1:  
    command: git commit -m "This is a comment."  
  command2:  
    command: "git commit -m \"This is a comment.\""  
  command3:  
    command: 'git commit -m "This is a comment."'  
  command4:  
    command:  
      - git  
      - commit  
      - -m  
      - This is a comment.
```

Pour spécifier plusieurs commandes, utilisez un [bloc littéral scalaire](#), comme illustré dans l'exemple suivant.

```
commands:  
  command block:  
    command: |
```

```
git commit -m "This is a comment."  
git push
```

#### env

(Facultatif) Définit des variables d'environnement pour la commande. Cette propriété remplace, plutôt que d'ajouter, l'environnement existant.

#### cwd

(Facultatif) Le répertoire de travail. Si cette option n'est pas spécifiée, les commandes sont exécutées à partir du répertoire racine (/).

#### test

(Facultatif) Commande qui doit renvoyer la valeur true (code de sortie 0) afin qu'Elastic Beanstalk traite la commande (par exemple, un script shell) contenue dans la clé command.

#### ignoreErrors

(Facultatif) Valeur booléenne qui détermine si les autres commandes doivent s'exécuter si la commande contenue dans la clé command échoue (renvoie une valeur non nulle). Définissez cette valeur sur true si vous voulez continuer à exécuter des commandes même si la commande échoue. Définissez-la sur false si vous souhaitez arrêter l'exécution des commandes si la commande échoue. La valeur par défaut est false.

## Exemple d'extrait

L'exemple d'extrait suivant exécute un script Python.

```
commands:  
  python_install:  
    command: myscript.py  
    cwd: /home/ec2-user  
    env:  
      myvarname: myvarvalue  
    test: "[ -x /usr/bin/python ]"
```

## Services

Vous pouvez utiliser la clé services pour définir les services qui doivent être démarrés ou arrêtés lors du lancement de l'instance. La clé services vous permet également de spécifier des dépendances sur des sources, des packages et des fichiers de sorte que si un redémarrage est nécessaire en raison d'une installation de fichiers, Elastic Beanstalk prenne en charge le redémarrage du service.

## Syntaxe

```
services:  
  sysvinit:  
    name of service:  
      enabled: "true"  
      ensureRunning: "true"  
      files:  
        - "file name"  
      sources:  
        - "directory"  
      packages:  
        name of package manager:  
          "package name[: version]"  
      commands:  
        - "name of command"
```

## Options

### ensureRunning

Définissez cette option sur `true` afin de garantir que le service s'exécute après qu'Elastic Beanstalk a terminé.

Définissez cette option sur `false` afin de garantir que le service ne s'exécute pas après qu'Elastic Beanstalk a terminé.

Ignorez cette clé pour n'apporter aucune modification à l'état du service.

### enabled

Définissez cette option sur `true` afin de garantir que le service soit démarré automatiquement lors du démarrage.

Définissez cette option sur `false` afin de garantir que le service ne soit pas démarré automatiquement lors du démarrage.

Ignorez cette clé pour n'apporter aucune modification à cette propriété.

### files

Liste de fichiers. Si Elastic Beanstalk en change un directement via le bloc de fichiers, le service est redémarré.

### sources

Liste de répertoires. Si Elastic Beanstalk développe une archive dans l'un de ces répertoires, le service est redémarré.

### packages

Une carte du gestionnaire de package sur une liste de noms de package. Si Elastic Beanstalk installe ou met à jour l'un de ces packages, le service est redémarré.

### commands

Liste de noms de commandes. Si Elastic Beanstalk exécute la commande spécifiée, le service est redémarré.

## Exemple d'extrait

Voici un exemple d'extrait :

```
services:  
  sysvinit:  
    myservice:  
      enabled: true  
      ensureRunning: true
```

## Commandes de conteneur

Vous pouvez utiliser la clé `container_commands` pour exécuter des commandes qui affectent le code source de votre application. Les commandes de conteneur s'exécutent après que l'application et le serveur web ont été configurés et que l'archive de version d'application a été extraite, mais avant que la version d'application soit déployée. Les commandes non-conteneur et autres opérations de personnalisation sont effectuées avant le code source d'application en cours d'extraction.

Les commandes spécifiées s'exécutent en tant qu'utilisateur racine et sont traitées dans l'ordre alphabétique par nom. Les commandes de conteneur sont exécutées depuis le répertoire intermédiaire, où votre code source est extrait avant d'être déployé sur le serveur d'applications. Toutes les modifications

apportées à votre code source dans le répertoire intermédiaire avec une commande conteneur seront incluses lorsque la source est déployée sur son emplacement final.

Pour résoudre les problèmes associés à vos commandes de conteneur, vous pouvez trouver leur sortie dans les [journaux d'instance \(p. 884\)](#).

Vous pouvez utiliser `leader_only` pour exécuter uniquement la commande sur une seule instance, ou configurer un `test` pour exécuter uniquement la commande lorsqu'une commande `test` évalue sur `true`. Les commandes de conteneur principales uniquement sont exécutées uniquement au cours de la création et du déploiement de l'environnement, tandis que d'autres commandes et opérations de personnalisation de serveur sont effectuées chaque fois qu'une instance est mise en service ou mise à jour. Les commandes de conteneur principales uniquement ne sont pas exécutées pour lancer des modifications de configuration, par exemple, une modification d'ID d'AMI ou de type d'instance.

## Syntaxe

```
container_commands:  
  name_of_container_command:  
    command: "command to run"  
    leader_only: true  
  name_of_container_command:  
    command: "command to run"
```

## Options

### command

Une chaîne ou un tableau de chaînes à exécuter.

### env

(Facultatif) Définissez les variables d'environnement avant d'exécuter la commande, en ignorant toute valeur existante.

### cwd

(Facultatif) Le répertoire de travail. Par défaut, il s'agit du répertoire intermédiaire de l'application décompressée.

### leader\_only

(Facultatif) Exécutez uniquement la commande sur une seule instance choisie par Elastic Beanstalk. Les commandes de conteneur principales uniquement sont exécutées avant d'autres commandes de conteneur. Une commande peut être principale uniquement ou disposer d'un `test`, mais pas les deux (`leader_only` a la priorité).

### test

(Facultatif) Exécutez une commande `test` qui doit renvoyer la valeur `true` afin d'exécuter la commande de conteneur. Une commande peut être principale uniquement ou disposer d'un `test`, mais pas les deux (`leader_only` a la priorité).

### ignoreErrors

(Facultatif) Ne manquez pas les déploiements si la commande de conteneur renvoie une valeur différente de 0 (réussite). Définissez sur `true` pour activer.

## Exemple d'extrait

Voici un exemple d'extrait :

```
container_commands:  
  collectstatic:
```

```
command: "django-admin.py collectstatic --noinput"
01syncdb:
    command: "django-admin.py syncdb --noinput"
    leader_only: true
02migrate:
    command: "django-admin.py migrate"
    leader_only: true
99customize:
    command: "scripts/customize.sh"
```

## Exemple : utilisation de métriques personnalisées Amazon CloudWatch

Amazon CloudWatch est un service web qui vous permet de surveiller, de gérer et de publier différentes métriques, et de configurer des actions d'alarme basées sur les données des métriques. Vous pouvez définir des métriques personnalisées pour votre propre utilisation, qui seront transmises à Amazon CloudWatch par Elastic Beanstalk. Une fois que vos métriques personnalisées sont incluses dans Amazon CloudWatch, vous pouvez les afficher dans la console Amazon CloudWatch.

Les scripts de surveillance d'Amazon CloudWatch pour Linux démontrent comment produire des métriques personnalisées Amazon CloudWatch et en tirer parti. Ces scripts comportent un exemple entièrement fonctionnel qui présente les métriques d'utilisation de la mémoire, des échanges et de l'espace disque, pour une instance Linux Amazon Elastic Compute Cloud (Amazon EC2). Pour de plus amples informations sur les scripts de surveillance Amazon CloudWatch, veuillez consulter [Scripts de supervision Amazon CloudWatch pour Linux](#) dans le Manuel du développeur Amazon CloudWatch.

### Note

Les [rapports améliorés sur l'état \(p. 837\)](#) d'Elastic Beanstalk assurent une prise en charge native de la publication d'un large éventail de métriques d'instance et d'environnement dans CloudWatch. Pour de plus amples informations, veuillez consulter [Publication de métriques personnalisées Amazon CloudWatch pour un environnement \(p. 862\)](#).

### Rubriques

- [Fichier de configuration .Ebextensions \(p. 750\)](#)
- [Autorisations \(p. 751\)](#)
- [Affichage des métriques dans la console CloudWatch \(p. 752\)](#)

## Fichier de configuration .Ebextensions

Cet exemple utilise des commandes et des paramètres d'option dans un fichier de configuration .ebextensions pour télécharger, installer et exécuter des scripts de surveillance fournis par Amazon CloudWatch.

Pour utiliser cet exemple, enregistrez-le dans un fichier nommé `cloudwatch.config`, dans un répertoire nommé `.ebextensions` au niveau supérieur de votre répertoire de projet. Déployez ensuite votre application via la console Elastic Beanstalk (en incluant le répertoire `.ebextensions` dans votre [bundle de fichiers source \(p. 415\)](#) ou via l'[interface de ligne de commande EB \(p. 1027\)](#)).

Pour plus d'informations sur les fichiers de configuration, consultez [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

`.ebextensions/cloudwatch.config`

```
packages:
  yum:
    perl-DateTime: []
    perl-Sys-Syslog: []
```

```
perl-LWP-Protocol-https: []
perl-Switch: []
perl-URI: []
perl-Bundle-LWP: []

sources:
    /opt/cloudwatch: https://aws-cloudwatch.s3.amazonaws.com/downloads/
CloudWatchMonitoringScripts-1.2.1.zip

container_commands:
    01-setupcron:
        command: |
            echo '*/5 * * * * root perl /opt/cloudwatch/aws-scripts-mon/mon-put-instance-data.pl
`{"Fn::GetOptionSetting" : { "OptionName" : "CloudWatchMetrics", "DefaultValue" : "--mem-
util --disk-space-util --disk-path=/ " }}` >> /var/log/cwpump.log 2>&1' > /etc/cron.d/cwpump
    02-changeperm:
        command: chmod 644 /etc/cron.d/cwpump
    03-changeperm:
        command: chmod u+x /opt/cloudwatch/aws-scripts-mon/mon-put-instance-data.pl

option_settings:
    "aws:autoscaling:launchconfiguration" :
        IamInstanceProfile : "aws-elasticbeanstalk-ec2-role"
    "aws:elasticbeanstalk:customoption" :
        CloudWatchMetrics : "--mem-util --mem-used --mem-avail --disk-space-util --disk-space-
used --disk-space-avail --disk-path=/ --auto-scaling"
```

Après avoir vérifié le fonctionnement du fichier de configuration, vous pouvez économiser l'espace disque en remplaçant la commande de redirection du fichier journal (`>> /var/log/cwpump.log 2>&1`) par `/dev/null` (`> /dev/null`).

## Autorisations

Pour publier des métriques Amazon CloudWatch personnalisées, les instances de votre environnement doivent être autorisées à utiliser CloudWatch. Vous pouvez accorder des autorisations aux instances de votre environnement en les ajoutant au [profil d'instance \(p. 22\)](#) de l'environnement. Vous pouvez ajouter des autorisations au profil d'instance avant ou après le déploiement de votre application.

Pour accorder des autorisations de publication des métriques CloudWatch

1. Ouvrez la console IAM à l'adresse <https://console.aws.amazon.com/iam/>.
2. Dans le panneau de navigation, choisissez Rôles.
3. Sélectionnez le rôle de profil d'instance de votre environnement. Par défaut, lorsque vous créez un environnement via la console Elastic Beanstalk ou l'[interface de ligne de commande EB \(p. 1027\)](#), il s'agit de `aws-elasticbeanstalk-ec2-role`.
4. Choisissez l'onglet Autorisations.
5. Sous Inline Policies (Stratégies en ligne), dans la section Autorisations, choisissez Create Role Policy (Créer une stratégie de rôle).
6. Choisissez Custom Policy, puis Select.
7. Complétez les champs suivants, puis choisissez Apply Policy (Appliquer la stratégie) :

Nom de la politique

Nom de la stratégie.

Document de stratégie

Copiez et collez le texte suivant dans le document de stratégie :

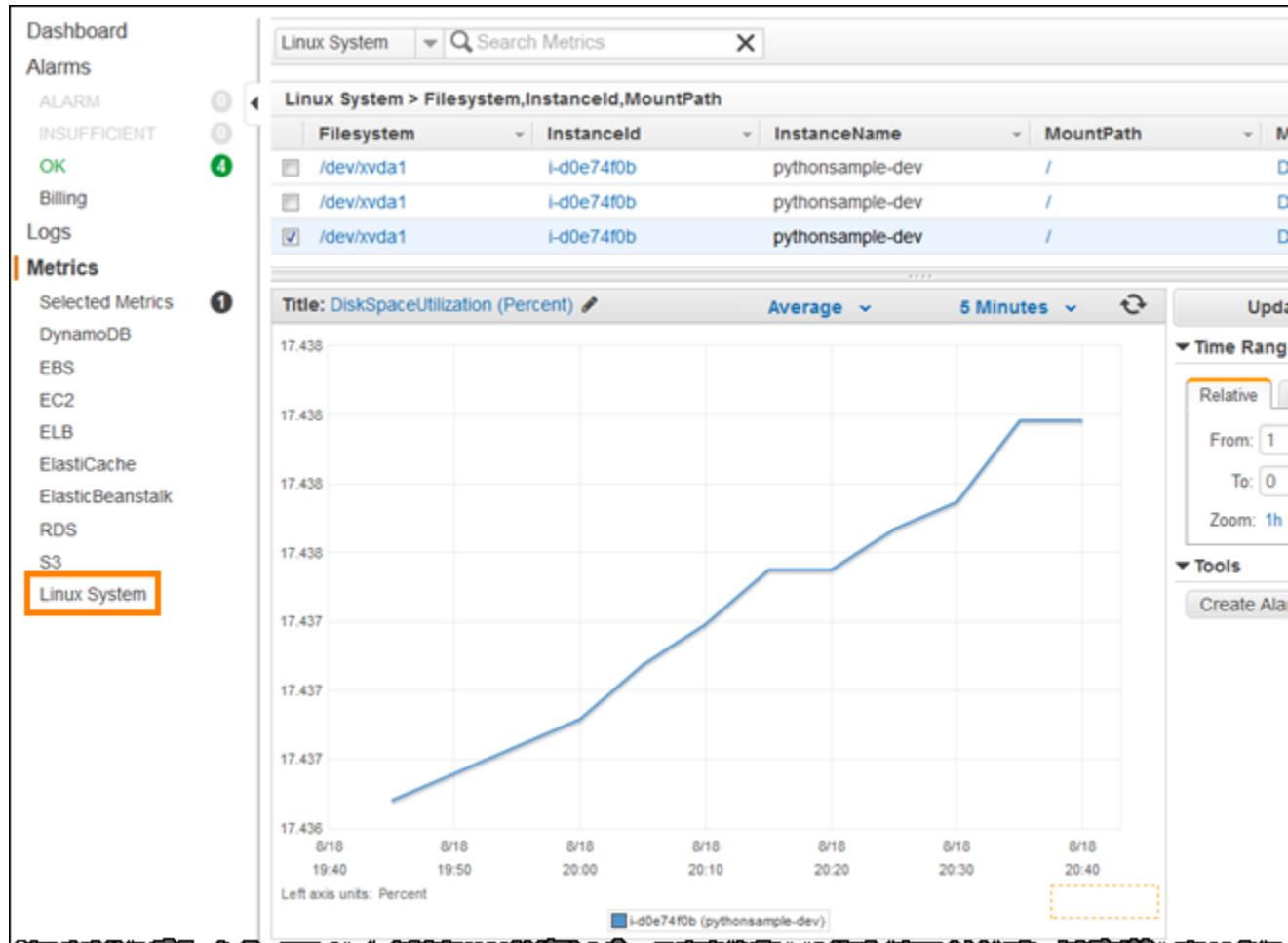
```
{
```

```
"Version": "2012-10-17",
"Statement": [
    {
        "Action": [
            "cloudwatch:PutMetricData",
            "ec2:DescribeTags"
        ],
        "Effect": "Allow",
        "Resource": [
            "*"
        ]
    }
]
```

Pour de plus amples informations sur la gestion des stratégies, veuillez consulter [Utilisation des stratégies](#) dans le Guide de l'utilisateur IAM.

## Affichage des métriques dans la console CloudWatch

Une fois que vous avez déployé le fichier de configuration CloudWatch dans votre environnement, consultez la [console Amazon CloudWatch](#) pour visualiser vos métriques. Les métriques personnalisées comportent le préfixe Système Linux.



## Personnalisation du logiciel sur des serveurs Windows

Vous pouvez souhaiter personnaliser et configurer le logiciel dont dépend votre application. Ces fichiers pourraient être soit des dépendances requises par l'application, par exemple, des services ou des packages supplémentaires qui doivent être exécutés. Pour obtenir des informations générales sur la personnalisation et la configuration de vos environnements Elastic Beanstalk, veuillez consulter [Configuration d'environnements Elastic Beanstalk \(p. 532\)](#).

### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Les fichiers de configuration prennent en charge les clés suivantes qui affectent le serveur Windows sur lequel votre application s'exécute.

### Clés

- [Packages \(p. 753\)](#)
- [Sources \(p. 754\)](#)
- [Dépôt de \(p. 754\)](#)
- [Commandes \(p. 756\)](#)
- [Services \(p. 757\)](#)
- [Commandes de conteneur \(p. 758\)](#)

Les clés sont traitées dans l'ordre dans lequel elles sont répertoriées ici.

### Note

Des versions antérieures (sans version) des versions de plateforme .NET ne traitent pas les fichiers de configuration dans l'ordre correct. Pour de plus amples informations, veuillez consulter [Migration entre les principales versions de la plateforme Windows Server pour Elastic Beanstalk \(p. 198\)](#).

Observez les [événements \(p. 879\)](#) de votre environnement pendant le développement et les tests des fichiers de configuration. Elastic Beanstalk ignore un fichier de configuration qui contient des erreurs de validation, comme une clé non valide, et ne traite aucune autre clé dans le même fichier. Lorsque cela se produit, Elastic Beanstalk ajoute un événement d'avertissement dans le journal des événements.

## Packages

Utilisez la clé `packages` pour télécharger, et installer des applications et des composants prépackagés.

Dans les environnements Windows, Elastic Beanstalk prend en charge le téléchargement et l'installation des packages MSI. (Les environnements Linux prennent en charge d'autres gestionnaires de packages. Pour de plus amples informations, veuillez consulter [Packages \(p. 741\)](#) sur la page Personnalisation du logiciel sur des serveurs Linux.)

Vous pouvez faire référence à des emplacements externes, comme un objet Amazon Simple Storage Service (Amazon S3), tant que l'URL est publiquement accessible.

Si vous spécifiez plusieurs packages `msi:`, leur ordre d'installation n'est pas garanti.

## Syntaxe

Spécifiez le nom de votre choix comme nom du package et une URL vers un emplacement de fichier MSI comme valeur. Vous pouvez spécifier plusieurs packages sous la clé `msi:`.

```
packages:  
  msi:  
    package name: package url  
    ...
```

## Exemples

L'exemple suivant indique une URL permettant de télécharger mysql à partir <https://dev.mysql.com/>.

```
packages:  
  msi:  
    mysql: https://dev.mysql.com/get/Downloads/Connector-Net/mysql-connector-net-8.0.11.msi
```

L'exemple suivant indique un objet Amazon S3 comme emplacement de fichier MSI.

```
packages:  
  msi:  
    mymsi: https://mybucket.s3.amazonaws.com/myobject.msi
```

## Sources

Utilisez la clé **sources** pour télécharger un fichier d'archives à partir d'une URL publique et le décompresser dans un répertoire cible de l'instance EC2.

### Syntaxe

```
sources:  
  target directory: location of archive file
```

### Formats pris en charge

Dans les environnements Windows, Elastic Beanstalk prend en charge le format .zip. (Les environnements Linux prennent en charge d'autres formats. Pour de plus amples informations, veuillez consulter [Sources \(p. 743\)](#) sur la page Personnalisation du logiciel sur des serveurs Linux.)

Vous pouvez faire référence à des emplacements externes, comme un objet Amazon Simple Storage Service (Amazon S3), tant que l'URL est publiquement accessible.

### Exemple

L'exemple suivant télécharge un fichier .zip public à partir d'un compartiment Amazon S3 et le décomprime dans c:/myproject/myapp.

```
sources:  
  "c:/myproject/myapp": https://mybucket.s3.amazonaws.com/myobject.zip
```

## Dépôt de

Utilisez la clé **files** pour créer des fichiers sur l'instance EC2. Le contenu peut être en ligne dans le fichier de configuration, ou issu d'une URL. Les fichiers sont écrits sur disque par ordre lexicographique. Pour télécharger des fichiers privés depuis Amazon S3, fournissez un profil d'instance pour l'autorisation.

### Syntaxe

```
files:
```

```
"target file location on disk":  
  source: URL  
  authentication: authentication name:  
  
"target file location on disk":  
  content: |  
    this is my content  
  encoding: encoding format
```

## Options

**content**

(Facultatif) Une chaîne.

**source**

(Facultatif) L'URL à partir de laquelle le fichier est chargé. Cette option ne peut pas être spécifiée avec la clé de contenu.

**encoding**

(Facultatif) Le format d'encodage. Cette option est utilisée uniquement pour une valeur de clé de contenu fournie. La valeur par défaut est `plain`.

Valeurs valides : `plain` | `base64`

**authentication**

(Facultatif) Le nom d'une [méthode d'authentification AWS CloudFormation](#) à utiliser. Vous pouvez ajouter des méthodes d'authentification aux métadonnées du groupe Auto Scaling via la clé Resources.

## Exemples

L'exemple suivant illustre les deux manières de fournir un contenu d'un fichier : depuis une URL, ou en ligne dans le fichier de configuration.

```
files:  
  "c:\\targetdirectory\\targetfile.txt":  
    source: http://foo.bar/myfile  
  
  "c:/targetdirectory/targetfile.txt":  
    content: |  
      # this is my file  
      # with content
```

### Note

Si vous utilisez une barre oblique inverse (\) dans votre chemin d'accès, vous devez la faire précéder d'une autre barre oblique inverse (caractère d'échappement), comme illustré dans l'exemple précédent.

L'exemple suivant utilise la clé Resources pour ajouter une méthode d'authentification nommée S3Auth et l'utilise pour télécharger un fichier privé à partir d'un compartiment Amazon S3 :

```
files:  
  "c:\\targetdirectory\\targetfile.zip":  
    source: https://elasticbeanstalk-us-east-2-123456789012.s3.amazonaws.com/prefix/  
myfile.zip  
    authentication: S3Auth
```

```
Resources:  
AWSEBAutoScalingGroup:  
  Metadata:  
    AWS::CloudFormation::Authentication:  
      S3Auth:  
        type: "s3"  
        buckets: ["elasticbeanstalk-us-east-2-123456789012"]  
        roleName:  
          Fn::GetOptionSetting:  
            Namespace: "aws:autoscaling:launchconfiguration"  
            OptionName: "IamInstanceProfile"  
            DefaultValue: "aws-elasticbeanstalk-ec2-role"
```

## Commandes

Utilisez la clé `commands` pour exécuter des commandes sur l'instance EC2. Les commandes sont traitées dans l'ordre alphabétique par nom et elles s'exécutent avant la configuration de l'application et du serveur web, avant que le fichier de version d'application soit extrait.

Les commandes spécifiées s'exécutent en tant qu'utilisateur Administrateur.

Pour résoudre les problèmes associés à vos commandes, vous pouvez trouver leur sortie dans les [journaux d'instance](#) (p. 884).

### Syntaxe

```
commands:  
  command name:  
    command: command to run
```

### Options

#### command

Soit un tableau, soit une chaîne spécifiant la commande à exécuter. Si vous utilisez un tableau, vous n'avez pas besoin de caractères d'espacement ou d'inclure des paramètres de commande entre guillemets.

#### cwd

(Facultatif) Le répertoire de travail. Par défaut, Elastic Beanstalk tente de trouver l'emplacement du répertoire de votre projet. S'il ne le trouve pas, il utilise `c:\Windows\System32` comme valeur par défaut.

#### env

(Facultatif) Définit des variables d'environnement pour la commande. Cette propriété remplace, plutôt que d'ajouter, l'environnement existant.

#### ignoreErrors

(Facultatif) Valeur booléenne qui détermine si les autres commandes doivent s'exécuter si la commande contenue dans la clé `command` échoue (renvoie une valeur non nulle). Définissez cette valeur sur `true` si vous voulez continuer à exécuter des commandes même si la commande échoue. Définissez-la sur `false` si vous souhaitez arrêter l'exécution des commandes si la commande échoue. La valeur par défaut est `false`.

#### test

(Facultatif) Commande qui doit renvoyer la valeur `true` (code de sortie 0) afin qu'Elastic Beanstalk traite la commande contenue dans la clé `command`.

#### waitForCompletion

(Facultatif) Secondes d'attente une fois que la commande est terminée avant d'exécuter la commande suivante. Si le système a besoin d'un redémarrage une fois la commande terminée, le système redémarre une fois le nombre de secondes spécifié écoulé. Si le système redémarre suite à une commande, Elastic Beanstalk récupère jusqu'au moment qui suit la commande dans le fichier de configuration. La valeur par défaut est de **60** secondes. Vous pouvez également spécifier **forever**, mais le système doit redémarrer avant que vous puissiez exécuter une autre commande.

## Exemple

L'exemple suivant enregistre la sortie de la commande `set` dans le fichier spécifié. S'il existe une commande suivante, Elastic Beanstalk exécute cette commande immédiatement à la fin de cette commande. Si cette commande exige un redémarrage, Elastic Beanstalk redémarre l'instance immédiatement à la fin de la commande.

```
commands:  
  test:  
    command: set > c:\\myapp\\set.txt  
    waitForCompletion: 0
```

## Services

Utilisez la clé `services` pour définir les services qui doivent être démarrés ou arrêtés lors du lancement de l'instance. La clé `services` vous permet également de spécifier des dépendances sur des sources, des packages et des fichiers de sorte que si un redémarrage est nécessaire en raison d'une installation de fichiers, Elastic Beanstalk prenne en charge le redémarrage du service.

## Syntaxe

```
services:  
  windows:  
    name of service:  
      files:  
        - "file name"  
      sources:  
        - "directory"  
      packages:  
        name of package manager:  
          "package name[: version]"  
      commands:  
        - "name of command"
```

## Options

#### ensureRunning

(Facultatif) Définissez cette option sur `true` afin de garantir que le service s'exécute après qu'Elastic Beanstalk a terminé.

Définissez cette option sur `false` afin de garantir que le service ne s'exécute pas après qu'Elastic Beanstalk a terminé.

Ignorez cette clé pour n'apporter aucune modification à l'état du service.

#### enabled

(Facultatif) Définissez cette option sur `true` afin de garantir que le service soit démarré automatiquement lors du démarrage.

Définissez cette option sur `false` afin de garantir que le service ne soit pas démarré automatiquement lors du démarrage.

Ignorez cette clé pour n'apporter aucune modification à cette propriété.

#### files

Liste de fichiers. Si Elastic Beanstalk en charge un directement via le bloc de fichiers, le service est redémarré.

#### sources

Liste de répertoires. Si Elastic Beanstalk développe une archive dans l'un de ces répertoires, le service est redémarré.

#### packages

Une carte du gestionnaire de package sur une liste de noms de package. Si Elastic Beanstalk installe ou met à jour l'un de ces packages, le service est redémarré.

#### commands

Liste de noms de commandes. Si Elastic Beanstalk exécute la commande spécifiée, le service est redémarré.

## Exemple

```
services:  
  windows:  
    myservice:  
      enabled: true  
      ensureRunning: true
```

## Commandes de conteneur

Utilisez la clé `container_commands` pour exécuter des commandes qui affectent le code source de votre application. Les commandes de conteneur s'exécutent après que l'application et le serveur web ont été configurés et que l'archive de version d'application a été extraite, mais avant que la version d'application soit déployée. Les commandes non-conteneur et autres opérations de personnalisation sont effectuées avant le code source d'application en cours d'extraction.

Les commandes de conteneur sont exécutées depuis le répertoire intermédiaire, où votre code source est extrait avant d'être déployé sur le serveur d'applications. Toutes les modifications apportées à votre code source dans le répertoire intermédiaire avec une commande conteneur seront incluses lorsque la source est déployée sur son emplacement final.

Pour résoudre les problèmes associés à vos commandes de conteneur, vous pouvez trouver leur sortie dans les [journaux d'instance \(p. 884\)](#).

Utilisez l'option `leader_only` pour exécuter uniquement la commande sur une seule instance, ou configurer un `test` pour exécuter uniquement la commande lorsqu'une commande `test` a la valeur `true`. Les commandes de conteneur principales uniquement sont exécutées uniquement au cours de la création et du déploiement de l'environnement, tandis que d'autres commandes et opérations de personnalisation de serveur sont effectuées chaque fois qu'une instance est mise en service ou mise à jour. Les commandes de conteneur principales uniquement ne sont pas exécutées pour lancer des modifications de configuration, par exemple, une modification d'ID d'AMI ou de type d'instance.

## Syntaxe

```
container_commands:  
  name of container_command:
```

command: *command to run*

## Options

command

Une chaîne ou un tableau de chaînes à exécuter.

env

(Facultatif) Définissez les variables d'environnement avant d'exécuter la commande, en ignorant toute valeur existante.

cwd

(Facultatif) Le répertoire de travail. Par défaut, il s'agit du répertoire intermédiaire de l'application décompressée.

leader\_only

(Facultatif) Exécutez uniquement la commande sur une seule instance choisie par Elastic Beanstalk. Les commandes de conteneur principales uniquement sont exécutées avant d'autres commandes de conteneur. Une commande peut être principale uniquement ou disposer d'un test, mais pas les deux (`leader_only` a la priorité).

test

(Facultatif) Exécutez une commande test qui doit renvoyer la valeur `true` afin d'exécuter la commande de conteneur. Une commande peut être principale uniquement ou disposer d'un test, mais pas les deux (`leader_only` a la priorité).

ignoreErrors

(Facultatif) Ne manquez pas les déploiements si la commande de conteneur renvoie une valeur différente de 0 (réussite). Définissez sur `true` pour activer.

waitForCompletion

(Facultatif) Secondes d'attente une fois que la commande est terminée avant d'exécuter la commande suivante. Si le système a besoin d'un redémarrage une fois la commande terminée, le système redémarre une fois le nombre de secondes spécifié écoulé. Si le système redémarre suite à une commande, Elastic Beanstalk récupère jusqu'au moment qui suit la commande dans le fichier de configuration. La valeur par défaut est de `60` secondes. Vous pouvez également spécifier `forever`, mais le système doit redémarrer avant que vous puissiez exécuter une autre commande.

## Exemple

L'exemple suivant enregistre la sortie de la commande `set` dans le fichier spécifié. Elastic Beanstalk exécute la commande sur une instance et redémarre l'instance immédiatement à la fin de la commande.

```
container_commands:  
  foo:  
    command: set > c:\\myapp\\set.txt  
    leader_only: true  
    waitAfterCompletion: 0
```

# Ajout et personnalisation des ressources de l'environnement Elastic Beanstalk

Vous pouvez personnaliser les ressources de votre environnement qui font partie de votre environnement Elastic Beanstalk. Par exemple, vous pouvez ajouter une file d'attente Amazon SQS et une alarme relative

à la longueur de la file d'attente, ou ajouter un cluster Amazon ElastiCache. Vous pouvez facilement personnaliser votre environnement en même temps que vous déployez votre version d'application, en incluant un fichier de configuration dans votre bundle source.

Vous pouvez utiliser la clé **Ressources** dans un [fichier de configuration \(p. 737\)](#) pour créer et personnaliser les ressources AWS de votre environnement. Les ressources définies dans les fichiers de configuration sont ajoutées au modèle AWS CloudFormation utilisé pour lancer votre environnement. Tous les [types de ressourcesAWS CloudFormation](#) sont pris en charge.

#### Note

Chaque fois que vous ajoutez une ressource qui n'est pas gérée par Elastic Beanstalk, assurez-vous d'ajouter une stratégie d'utilisateur avec les autorisations appropriées à vos utilisateurs AWS Identity and Access Management (IAM). Les [stratégies utilisateur gérées \(p. 946\)](#) fournies par Elastic Beanstalk ne couvrent que les autorisations pour les ressources gérées par Elastic Beanstalk.

Par exemple, le fichier de configuration suivant ajoute un hook de cycle de vie Auto Scaling au groupe Auto Scaling par défaut créé par Elastic Beanstalk :

**~/my-app/.ebextensions/as-hook.config**

```
Resources:
  hookrole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument: {
        "Version" : "2012-10-17",
        "Statement": [ {
          "Effect": "Allow",
          "Principal": {
            "Service": [ "autoscaling.amazonaws.com" ]
          },
          "Action": [ "sts:AssumeRole" ]
        } ]
      }
    Policies: [ {
      "PolicyName": "SNS",
      "PolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [ {
          "Effect": "Allow",
          "Resource": "*",
          "Action": [
            "sns:Publish"
          ]
        } ]
      }
    } ]
  hooktopic:
    Type: AWS::SNS::Topic
    Properties:
      Subscription:
        - Endpoint: "my-email@example.com"
          Protocol: email
  lifecyclehook:
    Type: AWS::AutoScaling::LifecycleHook
    Properties:
      AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
      LifecycleTransition: autoscaling:EC2_INSTANCE_TERMINATING
```

```
NotificationTargetARN: { "Ref" : "hooktopic" }
RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn" ] }
```

Cet exemple définit trois ressources : `hookrole`, `hooktopic` et `lifecyclehook`. Les deux premières ressources correspondent à un rôle IAM, qui autorise Amazon EC2 Auto Scaling à publier des messages dans Amazon SNS, et à une rubrique SNS, qui transmet des messages vers une adresse e-mail à partir du groupe Auto Scaling. Elastic Beanstalk crée ces ressources avec les propriétés et les types spécifiés.

La ressource finale, `lifecyclehook`, correspond au hook du cycle de vie lui-même :

```
lifecyclehook:
  Type: AWS::AutoScaling::LifecycleHook
  Properties:
    AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
    LifecycleTransition: autoscaling:EC2_INSTANCE_TERMINATING
    NotificationTargetARN: { "Ref" : "hooktopic" }
    RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn" ] }
```

La définition du hook de cycle de vie utilise deux [fonctions \(p. 764\)](#) pour renseigner les valeurs relatives aux propriétés du hook. `{ "Ref" : "AWSEBAutoScalingGroup" }` récupère le nom du groupe Auto Scaling créé par Elastic Beanstalk pour l'environnement. `AWSEBAutoScalingGroup` est l'un des [noms de ressource \(p. 761\)](#) standard fournis par Elastic Beanstalk.

Pour `AWS::IAM::Role`, `Ref` renvoie uniquement le nom du rôle, pas l'ARN. Pour obtenir l'ARN du paramètre `RoleARN`, vous devez utiliser une autre fonction intrinsèque, `Fn::GetAtt`, qui permet d'obtenir un attribut à partir d'une ressource. `RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn" ] }` obtient l'attribut `Arn` depuis la ressource `hookrole`.

`{ "Ref" : "hooktopic" }` permet d'obtenir l'ARN de la rubrique Amazon SNS créée précédemment dans le fichier de configuration. La valeur renvoyée par `Ref` varie selon le type de ressource et se trouve dans la rubrique du Guide de l'utilisateur AWS CloudFormation [dédiée au type de ressource AWS::SNS::Topic](#).

## Modification des ressources créées par Elastic Beanstalk pour votre environnement

Les ressources créées par Elastic Beanstalk pour votre environnement ont des noms. Vous pouvez utiliser ces noms pour obtenir des informations sur les ressources avec une [fonction \(p. 764\)](#), ou modifier les propriétés sur les ressources pour personnaliser leur comportement.

Les environnements de serveur Web disposent des ressources suivantes.

### Environnements de serveur web

- `AWSEBAutoScalingGroup` ([AWS::AutoScaling::AutoScalingGroup](#)) – Groupe Auto Scaling attaché à votre environnement.
- Une des deux ressources suivantes.
  - `AWSEBAutoScalingLaunchConfiguration` ([AWS::AutoScaling::LaunchConfiguration](#)) – Configuration de lancement attachée au groupe Auto Scaling de votre environnement.
  - `AWSEBEC2LaunchTemplate` ([AWS::EC2::LaunchTemplate](#)) – Modèle de lancement Amazon EC2 utilisé par le groupe Auto Scaling de votre environnement.

#### Note

Si votre environnement utilise une fonctionnalité qui exige des modèles de lancement Amazon EC2 et que votre stratégie utilisateur ne dispose pas des autorisations requises, la

création ou la mise à jour de l'environnement peut échouer. Utilisez la stratégie d'utilisateur gérée [AdministratorAccess-AWSElasticBeanstalk](#) (p. 946) ou ajoutez les autorisations requises à votre stratégie personnalisée (p. 947).

- **AWSEBEnvironmentName** ([AWS::ElasticBeanstalk::Environment](#)) – Votre environnement.
- **AWSEBSecurityGroup** ([AWS::EC2::SecurityGroup](#)) – Groupe de sécurité attaché à votre groupe Auto Scaling.
- **AWSEBRDSDatabase** ([AWS::RDS::DBInstance](#)) – Instance de base de données Amazon RDS attachée à votre environnement (le cas échéant).

Dans un environnement à charge équilibrée, vous pouvez accéder à des ressources supplémentaires liées à l'équilibrage de charge. Les équilibreurs de charge classiques disposent d'une ressource pour le programme d'équilibrage de charge et d'une autre pour le groupe de sécurité attaché à celui-ci. Les équilibrateurs de charge d'application et de réseau ont des ressources supplémentaires pour l'écouteur par défaut, la règle de l'écouteur et le groupe cible de l'équilibrage de charge.

#### Environnements à charge équilibrée

- **AWSEBLoadBalancer** ([AWS::ElasticLoadBalancing::LoadBalancer](#)) – Équilibrage de charge Classic Load Balancer de votre environnement.
- **AWSEBV2LoadBalancer** ([AWS::ElasticLoadBalancingV2::LoadBalancer](#)) – Équilibrage de charge Application Load Balancer ou Network Load Balancer de votre environnement.
- **AWSEBLoadBalancerSecurityGroup** ([AWS::EC2::SecurityGroup](#)) – Dans un [Amazon Virtual Private Cloud](#) (Amazon VPC) personnalisé uniquement, nom du groupe de sécurité créé par Elastic Beanstalk pour l'équilibrage de charge. Dans un VPC par défaut ou dans EC2-Classic, Elastic Load Balancing attribue un groupe de sécurité par défaut à l'équilibrage de charge.
- **AWSEBV2LoadBalancerListener** ([AWS::ElasticLoadBalancingV2::Listener](#)) – Écouteur qui permet à l'équilibrage de charge de vérifier les demandes de connexion et de les transférer à un ou plusieurs groupes cibles.
- **AWSEBV2LoadBalancerListenerRule** ([AWS::ElasticLoadBalancingV2::ListenerRule](#)) – Définit les demandes pour lesquelles l'écouteur Elastic Load Balancing effectue une action et précise cette action.
- **AWSEBV2LoadBalancerTargetGroup** ([AWS::ElasticLoadBalancingV2::TargetGroup](#)) – Groupe cible Elastic Load Balancing qui achemine les demandes vers une ou plusieurs cibles enregistrées, telles que les instances Amazon EC2.

Les environnements de travail disposent de ressources pour la file d'attente SQS qui place en mémoire tampon les demandes entrantes, et d'une table Amazon DynamoDB utilisée par les instances pour effectuer le choix principal.

#### Environnements de travail

- **AWSEBWorkerQueue** ([AWS::SQS::Queue](#)) – File d'attente Amazon SQS à partir de laquelle le démon extrait les demandes devant être traitées.
- **AWSEBWorkerDeadLetterQueue** ([AWS::SQS::Queue](#)) – File d'attente Amazon SQS qui stocke les messages ne pouvant pas être livrés ou n'ayant pas été traités avec succès par le démon.
- **AWSEBWorkerCronLeaderRegistry** ([AWS::DynamoDB::Table](#)) – Table Amazon DynamoDB qui constitue le registre interne utilisé par le démon pour des tâches périodiques.

## Autres clés de modèle AWS CloudFormation

Nous avons déjà présenté les clés de fichier de configuration de AWS CloudFormation, comme `Resources`, `files` et `packages`. Elastic Beanstalk ajoute le contenu des fichiers de configuration au modèle AWS CloudFormation qui prend en charge votre environnement ; vous pouvez donc utiliser

d'autres sections AWS CloudFormation pour effectuer des tâches avancées dans vos fichiers de configuration.

#### Clés

- [Paramètres \(p. 763\)](#)
- [Outputs \(p. 763\)](#)
- [Mappages \(p. 764\)](#)

## Paramètres

Les paramètres constituent une alternative aux [options personnalisées \(p. 736\)](#) d'Elastic Beanstalk que vous pouvez utiliser pour définir des valeurs utilisées dans d'autres endroits de vos fichiers de configuration. À l'instar des options personnalisées, vous pouvez utiliser des paramètres pour rassembler des valeurs configurables par l'utilisateur dans un seul endroit. Contrairement aux options personnalisées, vous ne pouvez pas utiliser l'API d'Elastic Beanstalk pour définir des valeurs des paramètres ; de même, le nombre de paramètres que vous pouvez définir dans un modèle est limité par AWS CloudFormation.

Vous pouvez souhaiter utiliser des paramètres afin que vos fichiers de configuration soient considérés comme des modèles AWS CloudFormation. Si vous utilisez des paramètres et non des options personnalisées, vous pouvez utiliser le fichier de configuration pour créer la même ressource dans AWS CloudFormation comme sa propre pile. Par exemple, vous pouvez disposer d'un fichier de configuration qui ajoute un système de fichiers Amazon EFS à votre environnement à des fins de test, puis utiliser le même fichier pour créer un système de fichiers indépendant non rattaché au cycle de vie de votre environnement pour une utilisation en production.

L'exemple suivant illustre l'utilisation de paramètres pour rassembler des valeurs configurables par l'utilisateur au début d'un fichier de configuration.

#### Example [Loadbalancer-accesslogs-existingbucket.config](#) – Paramètres

```
Parameters:  
  bucket:  
    Type: String  
    Description: "Name of the Amazon S3 bucket in which to store load balancer logs"  
    Default: "DOC-EXAMPLE-BUCKET"  
  bucketprefix:  
    Type: String  
    Description: "Optional prefix. Can't start or end with a /, or contain the word AWSLogs"  
    Default: ""
```

## Outputs

Vous pouvez utiliser un bloc `Outputs` pour exporter des informations sur les ressources créées vers AWS CloudFormation. Vous pouvez ensuite utiliser la fonction `Fn::ImportValue` pour extraire la valeur dans un modèle AWS CloudFormation en dehors d'Elastic Beanstalk.

L'exemple suivant montre comment créer une rubrique Amazon SNS et exporter son ARN vers AWS CloudFormation avec le nom `NotificationTopicArn`.

#### Example [sns-topic.config](#)

```
Resources:  
  NotificationTopic:  
    Type: AWS::SNS::Topic
```

```
Outputs:  
  NotificationTopicArn:  
    Description: Notification topic ARN  
    Value: { "Ref" : "NotificationTopic" }  
    Export:  
      Name: NotificationTopicArn
```

Dans un fichier de configuration pour un autre environnement, ou dans un modèle AWS CloudFormation hors d'Elastic Beanstalk, vous pouvez utiliser la fonction Fn::ImportValue pour obtenir l'ARN exporté. Cet exemple affecte la valeur exportée à une propriété d'environnement nommée TOPIC\_ARN.

### Example env.config

```
option_settings:  
  aws:elasticbeanstalk:application:environment:  
    TOPIC_ARN: `'{ "Fn::ImportValue" : "NotificationTopicArn" }`'
```

## Mappages

Vous pouvez utiliser un mappage pour stocker des paires clé/valeur organisées par espace de noms. Un mappage peut vous aider à organiser les valeurs que vous utilisez dans l'ensemble de vos configurations, ou à modifier une valeur de paramètre en fonction d'une autre valeur. Par exemple, la configuration suivante définit la valeur d'un paramètre d'ID de compte en fonction de la région en cours.

### Example Loadbalancer-accesslogs-newbucket.config – Mappages

```
Mappings:  
  Region2ELBAccountId:  
    us-east-1:  
      AccountId: "111122223333"  
    us-west-2:  
      AccountId: "444455556666"  
    us-west-1:  
      AccountId: "123456789012"  
    eu-west-1:  
      AccountId: "777788889999"  
...  
  Principal:  
    AWS:  
      ? "Fn::FindInMap"  
      :  
        - Region2ELBAccountId  
        - Ref: "AWS::Region"  
        - AccountId
```

## Fonctions

Vous pouvez utiliser des fonctions dans vos fichiers de configuration pour renseigner les valeurs des propriétés de ressource à l'aide des informations issues d'autres ressources ou de paramètres d'option de configuration Elastic Beanstalk. Elastic Beanstalk prend en charge les fonctions AWS CloudFormation (Ref, Fn::GetAtt, Fn::Join) et une fonction spécifique à Elastic Beanstalk, Fn::GetOptionSetting.

### Fonctions

- [Réf \(p. 765\)](#)
- [Fn::GetAtt \(p. 765\)](#)
- [Fn::Join \(p. 765\)](#)

- [Fn::GetOptionSetting \(p. 766\)](#)

## Réf

Utilisez `Ref` pour récupérer la représentation de chaîne par défaut d'une ressource AWS. La valeur renvoyée par `Ref` dépend du type de ressource et parfois d'autres facteurs également. Par exemple, un groupe de sécurité ([AWS::EC2::SecurityGroup](#)) renvoie soit le nom, soit l'ID du groupe de sécurité, selon que le groupe de sécurité se trouve dans un [Amazon Virtual Private Cloud \(Amazon VPC\)](#) par défaut, dans EC2-Classic ou dans un VPC personnalisé.

```
{ "Ref" : "resource name" }
```

### Note

Pour de plus amples informations sur chaque type de ressource, y compris la ou les valeurs de renvoi de `Ref`, veuillez consulter [Référence des types de ressources AWS](#) dans le Guide de l'utilisateur AWS CloudFormation.

À partir de l'exemple de [hook de cycle de vie Auto Scaling \(p. 759\)](#) :

```
Resources:  
  lifecyclehook:  
    Type: AWS::AutoScaling::LifecycleHook  
    Properties:  
      AutoScalingGroupName: { "Ref" : "AWSEBAutoScalingGroup" }
```

Vous pouvez également utiliser `Ref` pour récupérer la valeur d'un paramètre AWS CloudFormation défini ailleurs dans le même fichier ou dans un fichier de configuration différent.

## Fn::GetAtt

Utilisez `Fn::GetAtt` pour récupérer la valeur d'un attribut sur une ressource AWS.

```
{ "Fn::GetAtt" : [ "resource name", "attribute name" ] }
```

À partir de l'exemple de [hook de cycle de vie Auto Scaling \(p. 759\)](#) :

```
Resources:  
  lifecyclehook:  
    Type: AWS::AutoScaling::LifecycleHook  
    Properties:  
      RoleARN: { "Fn::GetAtt" : [ "hookrole", "Arn" ] }
```

Pour de plus amples informations, veuillez consulter [Fn::GetAtt](#).

## Fn::Join

Utilisez `Fn::Join` pour associer des chaînes avec un délimiteur. Les chaînes peuvent être codées en dur ou utiliser le résultat de `Fn::GetAtt` ou `Ref`.

```
{ "Fn::Join" : [ "delimiter", [ "string1", "string2" ] ] }
```

Pour de plus amples informations, veuillez consulter [Fn::Join](#).

## Fn::GetOptionSetting

Utilisez Fn::GetOptionSetting pour récupérer la valeur d'un paramètre d'[option de configuration](#) (p. 658) appliquée à l'environnement.

```
"Fn::GetOptionSetting":  
  Namespace: "namespace"  
  OptionName: "option name"  
  DefaultValue: "default value"
```

À partir de l'exemple [stockage de clés privées](#) (p. 828) :

```
Resources:  
  AWSEBAutoScalingGroup:  
    Metadata:  
      AWS::CloudFormation::Authentication:  
        S3Auth:  
          type: "s3"  
          buckets: ["elasticbeanstalk-us-west-2-123456789012"]  
          roleName:  
            "Fn::GetOptionSetting":  
              Namespace: "aws:autoscaling:launchconfiguration"  
              OptionName: "IamInstanceProfile"  
              DefaultValue: "aws-elasticbeanstalk-ec2-role"
```

## Exemples de ressources personnalisées

Voici une liste d'exemples de fichiers de configuration que vous pouvez utiliser pour personnaliser vos environnements Elastic Beanstalk :

- [DynamoDB, CloudWatch et SNS](#)
- [Elastic Load Balancing et CloudWatch](#)
- [ElastiCache](#)
- [RDS et CloudWatch](#)
- [SQS, SNS et CloudWatch](#)

Les sous-rubriques de cette page fournissent des exemples étendus pour l'ajout et la configuration de ressources personnalisées dans un environnement Elastic Beanstalk.

### Exemples

- [Exemple : ElastiCache \(p. 766\)](#)
- [Exemple : SQS, CloudWatch et SNS \(p. 773\)](#)
- [Exemple : DynamoDB, CloudWatch et SNS \(p. 775\)](#)

### Exemple : ElastiCache

Les exemples suivants permettent d'ajouter un cluster Amazon ElastiCache aux plateformes EC2-Classic et EC2-VPC ([Amazon Virtual Private Cloud](#) (Amazon VPC) par défaut et personnalisé). Pour de plus amples informations sur ces plateformes et sur la façon d'identifier celles qui sont prises en charge par EC2 pour votre région et votre compte AWS, veuillez consulter <https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-supported-platforms.html>. Consultez ensuite la section de cette rubrique qui s'applique à votre plateforme.

- [Plateformes EC2-classic \(p. 767\)](#)

- EC2-VPC (par défaut) (p. 768)
- EC2-VPC (personnalisé) (p. 770)

## Plateformes EC2-classic

Cet exemple ajoute un cluster Amazon ElastiCache à un environnement avec des instances lancées dans la plateforme EC2-Classic. Toutes les propriétés répertoriées dans cet exemple correspondent aux propriétés requises minimales qui doivent être définies pour chaque type de ressource. Vous pouvez télécharger l'exemple sur la page [Exemple ElastiCache](#).

### Note

Cet exemple crée des ressources AWS, qui peuvent éventuellement vous être facturées. Pour de plus amples informations sur la tarification AWS, veuillez consulter <http://aws.amazon.com/pricing/>. Certains services font partie du niveau d'offre gratuite d'AWS. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour de plus amples informations, veuillez consulter <http://aws.amazon.com/free/>.

Pour utiliser cet exemple, procédez comme suit :

1. Créez un répertoire [.ebextensions](#) (p. 737) dans le répertoire de niveau supérieur de votre bundle de fichiers source.
2. Créez deux fichiers de configuration avec l'extension `.config`, puis placez-les dans votre répertoire `.ebextensions`. Un fichier de configuration définit les ressources et l'autre fichier de configuration définit les options.
3. Déployez votre application sur Elastic Beanstalk.

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Créez un fichier de configuration (par exemple, `elasticache.config`) qui définit les ressources. Dans cet exemple, nous créons le cluster ElastiCache en spécifiant le nom de la ressource de cluster ElastiCache (`MyElasticCache`), en déclarant son type, puis en configurant les propriétés pour le cluster. L'exemple fait référence au nom de la ressource de groupe de sécurité ElastiCache qui est créée et définie dans ce fichier de configuration. Ensuite, nous créons un groupe de sécurité ElastiCache. Nous définissons le nom de cette ressource, nous déclarons son type, puis nous ajoutons une description du groupe de sécurité. Enfin, nous définissons les règles de trafic entrant pour le groupe de sécurité ElastiCache afin de n'autoriser l'accès qu'à partir des instances incluses dans le groupe de sécurité ElastiCache (`MyCacheSecurityGroup`) et dans le groupe de sécurité Elastic Beanstalk (`AWSEBSecurityGroup`). Le nom du paramètre, `AWSEBSecurityGroup`, est un nom de ressource fixe fourni par Elastic Beanstalk. Vous devez ajouter `AWSEBSecurityGroup` aux règles de trafic entrant de votre groupe de sécurité ElastiCache pour que votre application Elastic Beanstalk se connecte aux instances de votre cluster ElastiCache.

```
#This sample requires you to create a separate configuration file that defines the custom
option settings for CacheCluster properties.

Resources:
  MyElasticCache:
    Type: AWS::ElastiCache::CacheCluster
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName : CacheNodeType
          DefaultValue: cache.m1.small
      NumCacheNodes:
```

```
Fn::GetOptionSetting:  
  OptionName : NumCacheNodes  
  DefaultValue: 1  
Engine:  
  Fn::GetOptionSetting:  
    OptionName : Engine  
    DefaultValue: memcached  
CacheSecurityGroupNames:  
  - Ref: MyCacheSecurityGroup  
MyCacheSecurityGroup:  
  Type: AWS::ElastiCache::SecurityGroup  
  Properties:  
    Description: "Lock cache down to webserver access only"  
MyCacheSecurityGroupIngress:  
  Type: AWS::ElastiCache::SecurityGroupIngress  
  Properties:  
    CacheSecurityGroupName:  
      Ref: MyCacheSecurityGroup  
    EC2SecurityGroupName:  
      Ref: AWSEBSecurityGroup
```

Pour plus d'informations sur les ressources utilisées dans cet exemple de fichier de configuration, consultez les références suivantes :

- [AWS::ElastiCache::CacheCluster](#)
- [AWS::ElastiCache::SecurityGroup](#)
- [AWS::ElastiCache::SecurityGroupIngress](#)

Créez un fichier de configuration distinct nommé `options.config` et définissez les paramètres d'option personnalisés.

```
option_settings:  
  "aws:elasticbeanstalk:customoption":  
    CacheNodeType : cache.m1.small  
    NumCacheNodes : 1  
    Engine : memcached
```

Avec ces lignes de code, Elastic Beanstalk récupère les valeurs des propriétés CacheNodeType, NumCacheNodes et Engine à partir des valeurs CacheNodeType, NumCacheNodes et Engine figurant dans un fichier de configuration (`options.config` dans notre exemple). Celui-ci contient une section `option_settings` dont la section `aws:elasticbeanstalk:customoption` inclut une paire nom-valeur qui contient la valeur réelle à utiliser. Dans l'exemple ci-dessus, cela signifie que les éléments `cache.m1.small`, `1` et `memcached` seraient utilisés pour les valeurs. Pour de plus amples informations sur `Fn::GetOptionSetting`, veuillez consulter [Fonctions \(p. 764\)](#).

### EC2-VPC (par défaut)

Cet exemple ajoute un cluster Amazon ElastiCache à un environnement avec des instances lancées dans la plateforme EC2-VPC. Plus précisément, les informations de cette section s'appliquent à un scénario où EC2 lance des instances dans le VPC par défaut. Toutes les propriétés figurant dans cet exemple correspondent aux propriétés requises minimales qui doivent être définies pour chaque type de ressource. Pour de plus amples informations sur les VPC par défaut, veuillez consulter [Vos VPC et sous-réseaux par défaut](#).

#### Note

Cet exemple crée des ressources AWS, qui peuvent éventuellement vous être facturées. Pour de plus amples informations sur la tarification AWS, veuillez consulter <http://aws.amazon.com/>

[pricing](#). Certains services font partie du niveau d'offre gratuite d'AWS. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour de plus amples informations, veuillez consulter <http://aws.amazon.com/free/>.

Pour utiliser cet exemple, procédez comme suit :

1. Créez un répertoire [.ebextensions](#) (p. 737) dans le répertoire de niveau supérieur de votre bundle de fichiers source.
2. Créez deux fichiers de configuration avec l'extension `.config`, puis placez-les dans votre répertoire `.ebextensions`. Un fichier de configuration définit les ressources et l'autre fichier de configuration définit les options.
3. Déployez votre application sur Elastic Beanstalk.

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Ensuite, attribuez le nom `elasticache.config` au fichier de configuration de ressources. Pour créer le cluster ElastiCache, cet exemple spécifie le nom de la ressource de cluster ElastiCache (`MyElasticCache`), déclare son type, puis configure les propriétés pour le cluster. L'exemple fait référence à l'ID de la ressource de groupe de sécurité que nous créons et définissons dans ce fichier de configuration.

Ensuite, nous créons un groupe de sécurité EC2. Nous définissons le nom de cette ressource, déclarons son type, ajoutons une description et définissons les règles de trafic entrant pour le groupe de sécurité afin de n'autoriser l'accès qu'à partir des instances incluses dans le groupe de sécurité Elastic Beanstalk (`AWSEBSecurityGroup`). (Le nom du paramètre, `AWSEBSecurityGroup`, est un nom de ressource fixe fourni par Elastic Beanstalk. Vous devez ajouter `AWSEBSecurityGroup` aux règles de trafic entrant de votre groupe de sécurité ElastiCache pour que votre application Elastic Beanstalk se connecte aux instances de votre cluster ElastiCache.)

Les règles de trafic entrant pour le groupe de sécurité EC2 définissent également le protocole IP et les numéros de port sur lesquels les nœuds de cache peuvent accepter des connexions. Pour Redis, le numéro de port par défaut est 6379.

```
#This sample requires you to create a separate configuration file that defines the custom
option settings for CacheCluster properties.

Resources:
  MyCacheSecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
    Properties:
      GroupDescription: "Lock cache down to webserver access only"
      SecurityGroupIngress :
        - IpProtocol : "tcp"
          FromPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
          ToPort :
            Fn::GetOptionSetting:
              OptionName : "CachePort"
              DefaultValue: "6379"
              SourceSecurityGroupName:
                Ref: "AWSEBSecurityGroup"
  MyElasticCache:
    Type: "AWS::ElastiCache::CacheCluster"
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
```

```
    OptionName : "CacheNodeType"
    DefaultValue : "cache.t2.micro"
  NumCacheNodes:
    Fn::GetOptionSetting:
      OptionName : "NumCacheNodes"
      DefaultValue : "1"
  Engine:
    Fn::GetOptionSetting:
      OptionName : "Engine"
      DefaultValue : "redis"
  VpcSecurityGroupIds:
    -
      Fn::GetAtt:
        - MyCacheSecurityGroup
        - GroupId

Outputs:
  ElastiCache:
    Description : "ID of ElastiCache Cache Cluster with Redis Engine"
    Value :
      Ref : "MyElastiCache"
```

Pour plus d'informations sur les ressources utilisées dans cet exemple de fichier de configuration, consultez les références suivantes :

- [AWS::ElastiCache::CacheCluster](#)
- [AWS::EC2::SecurityGroup](#)

Ensuite, attribuez le nom `options.config` au fichier de configuration des options et définissez les paramètres d'option personnalisés.

```
option_settings:
  "aws:elasticbeanstalk:customoption":
    CacheNodeType : cache.t2.micro
    NumCacheNodes : 1
    Engine : redis
    CachePort : 6379
```

Avec ces lignes de code, Elastic Beanstalk récupère les valeurs des propriétés `CacheNodeType`, `NumCacheNodes`, `Engine` et `CachePort` à partir des valeurs `CacheNodeType`, `NumCacheNodes`, `Engine` et `CachePort` figurant dans un fichier de configuration (`options.config` dans notre exemple). Ce fichier inclut une section `aws:elasticbeanstalk:customoption` (sous `option_settings`) qui contient des paires nom-valeur incluant les valeurs réelles à utiliser. Dans l'exemple précédent, les éléments `cache.t2.micro`, `1`, `redis` et `6379` seraient utilisés pour les valeurs. Pour de plus amples informations sur `Fn::GetOptionSetting`, veuillez consulter [Fonctions \(p. 764\)](#).

## EC2-VPC (personnalisé)

Si vous créez un VPC personnalisé sur la plateforme EC2-VPC et que vous le spécifiez en tant que VPC dans lequel EC2 lance les instances, le processus d'ajout d'un cluster Amazon ElastiCache à votre environnement diffère de celui d'un VPC par défaut. La différence principale est que vous devez créer un groupe de sous-réseaux pour le cluster ElastiCache. Toutes les propriétés figurant dans cet exemple correspondent aux propriétés requises minimales qui doivent être définies pour chaque type de ressource.

### Note

Cet exemple crée des ressources AWS, qui peuvent éventuellement vous être facturées. Pour de plus amples informations sur la tarification AWS, veuillez consulter <http://aws.amazon.com/pricing/>. Certains services font partie du niveau d'offre gratuite d'AWS. Si vous êtes un nouveau

client, vous pouvez essayer ces services gratuitement. Pour de plus amples informations, veuillez consulter <http://aws.amazon.com/free/>.

Pour utiliser cet exemple, procédez comme suit :

1. Créez un répertoire [.ebextensions \(p. 737\)](#) dans le répertoire de niveau supérieur de votre bundle de fichiers source.
2. Créez deux fichiers de configuration avec l'extension `.config`, puis placez-les dans votre répertoire `.ebextensions`. Un fichier de configuration définit les ressources et l'autre fichier de configuration définit les options.
3. Déployez votre application sur Elastic Beanstalk.

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Ensuite, attribuez le nom `elasticache.config` au fichier de configuration de ressources. Pour créer le cluster ElastiCache, cet exemple spécifie le nom de la ressource de cluster ElastiCache (`MyElasticCache`), déclare son type, puis configure les propriétés pour le cluster. Dans cet exemple, les propriétés font référence au nom du groupe de sous-réseaux pour le cluster ElastiCache ainsi qu'à l'ID de la ressource de groupe de sécurité que nous créons et définissons dans ce fichier de configuration.

Ensuite, nous créons un groupe de sécurité EC2. Nous définissons le nom de cette ressource, déclarons son type, ajoutons une description et l'ID de VPC, et définissons les règles de trafic entrant pour le groupe de sécurité afin de n'autoriser l'accès qu'à partir des instances incluses dans le groupe de sécurité Elastic Beanstalk (`AWSEBSecurityGroup`). (Le nom du paramètre, `AWSEBSecurityGroup`, est un nom de ressource fixe fourni par Elastic Beanstalk. Vous devez ajouter `AWSEBSecurityGroup` aux règles de trafic entrant de votre groupe de sécurité ElastiCache pour que votre application Elastic Beanstalk se connecte aux instances de votre cluster ElastiCache.)

Les règles de trafic entrant pour le groupe de sécurité EC2 définissent également le protocole IP et les numéros de port sur lesquels les nœuds de cache peuvent accepter des connexions. Pour Redis, le numéro de port par défaut est 6379. Enfin, cet exemple crée un groupe de sous-réseaux pour le cluster ElastiCache. Nous définissons le nom de cette ressource, déclarons son type et ajoutons la description et l'ID du sous-réseau dans le groupe de sous-réseaux.

#### Note

Nous vous recommandons d'utiliser des sous-réseaux privés pour le cluster ElastiCache. Pour de plus amples informations sur un VPC avec un sous-réseau privé, veuillez consulter [https://docs.aws.amazon.com/vpc/latest/userguide/VPC\\_Scenario2.html](https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Scenario2.html).

```
#This sample requires you to create a separate configuration file that defines the custom
option settings for CacheCluster properties.

Resources:
  MyElasticCache:
    Type: "AWS::ElastiCache::CacheCluster"
    Properties:
      CacheNodeType:
        Fn::GetOptionSetting:
          OptionName : "CacheNodeType"
          DefaultValue : "cache.t2.micro"
      NumCacheNodes:
        Fn::GetOptionSetting:
          OptionName : "NumCacheNodes"
          DefaultValue : "1"
      Engine:
        Fn::GetOptionSetting:
```

```

        OptionName : "Engine"
        DefaultValue : "redis"
    CacheSubnetGroupName:
        Ref: "MyCacheSubnets"
    VpcSecurityGroupIds:
        - Ref: "MyCacheSecurityGroup"
MyCacheSecurityGroup:
    Type: "AWS::EC2::SecurityGroup"
    Properties:
        GroupDescription: "Lock cache down to webserver access only"
    VpcId:
        Fn::GetOptionSetting:
            OptionName : "VpcId"
    SecurityGroupIngress :
        - IpProtocol : "tcp"
        FromPort :
            Fn::GetOptionSetting:
                OptionName : "CachePort"
                DefaultValue: "6379"
        ToPort :
            Fn::GetOptionSetting:
                OptionName : "CachePort"
                DefaultValue: "6379"
        SourceSecurityGroupId:
            Ref: "AWSEBSecurityGroup"
MyCacheSubnets:
    Type: "AWS::ElastiCache::SubnetGroup"
    Properties:
        Description: "Subnets for ElastiCache"
    SubnetIds:
        Fn::GetOptionSetting:
            OptionName : "CacheSubnets"
Outputs:
    ElastiCache:
        Description : "ID of ElastiCache Cache Cluster with Redis Engine"
        Value :
            Ref : "MyElastiCache"

```

Pour plus d'informations sur les ressources utilisées dans cet exemple de fichier de configuration, consultez les références suivantes :

- [AWS::ElastiCache::CacheCluster](#)
- [AWS::EC2::SecurityGroup](#)
- [AWS::ElastiCache::SubnetGroup](#)

Ensuite, attribuez le nom `options.config` au fichier de configuration des options et définissez les paramètres d'option personnalisés.

#### Note

Dans l'exemple suivant, remplacez les exemples de valeurs `CacheSubnets` et `VpcId` par vos propres sous-réseaux et VPC.

```

option_settings:
    "aws:elasticbeanstalk:customoption":
        CacheNodeType : cache.t2.micro
        NumCacheNodes : 1
        Engine : redis
        CachePort : 6379
        CacheSubnets:
            - subnet-1a1a1a1a

```

```
- subnet-2b2b2b2b
- subnet-3c3c3c3c
VpcId: vpc-4d4d4d4d
```

Avec ces lignes de code, Elastic Beanstalk récupère les valeurs des propriétés CacheNodeType, NumCacheNodes, Engine, CachePort, CacheSubnets et VpcId à partir des valeurs CacheNodeType, NumCacheNodes, Engine, CachePort, CacheSubnets et VpcId figurant dans un fichier de configuration (`options.config` dans notre exemple). Ce fichier inclut une section `aws:elasticbeanstalk:customoption` (sous `option_settings`) qui contient des paires nom-valeur incluant des exemples de valeurs. Dans l'exemple ci-dessus, les éléments `cache.t2.micro`, `1, redis, 6379, subnet-1a1a1a1a`, `subnet-2b2b2b2b`, `subnet-3c3c3c3c` et `vpc-4d4d4d4d` seraient utilisés pour les valeurs. Pour plus d'informations sur `Fn::GetOptionSetting`, consultez Fonctions (p. 764).

## Exemple : SQS, CloudWatch et SNS

Dans cet exemple, une file d'attente Amazon SQS et une alarme relative à la longueur de la file d'attente sont ajoutées à l'environnement. Les propriétés que vous voyez dans cet exemple correspondent aux propriétés minimales requises que vous devez définir pour chacune de ces ressources. Vous pouvez télécharger l'exemple sur la page [SQS, SNS et CloudWatch](#).

### Note

Cet exemple crée des ressources AWS, qui peuvent éventuellement vous être facturées. Pour de plus amples informations sur la tarification AWS, veuillez consulter <http://aws.amazon.com/pricing/>. Certains services font partie du niveau d'offre gratuite d'AWS. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour de plus amples informations, veuillez consulter <http://aws.amazon.com/free/>.

Pour utiliser cet exemple, procédez comme suit :

1. Créez un répertoire `.ebextensions` (p. 737) dans le répertoire de niveau supérieur de votre bundle de fichiers source.
2. Créez deux fichiers de configuration avec l'extension `.config`, puis placez-les dans votre répertoire `.ebextensions`. Un fichier de configuration définit les ressources et l'autre fichier de configuration définit les options.
3. Déployez votre application sur Elastic Beanstalk.

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Créez un fichier de configuration (par exemple, `sqs.config`) qui définit les ressources. Dans cet exemple, nous créons une file d'attente SQS et définissons la propriété `VisibilityTimeout` dans la ressource `MySQSQueue`. Ensuite, nous créons une rubrique (Topic) SNS et nous indiquons que l'e-mail doit être envoyé à `someone@example.com` lorsque l'alarme se déclenche. Enfin, nous créons une alarme CloudWatch qui se déclenche si la file d'attente dépasse 10 messages. Dans la propriété `Dimensions`, nous spécifions le nom de la dimension et la valeur représentant la mesure de la dimension. Nous utilisons `Fn::GetAtt` pour renvoyer la valeur de `QueueName` à partir de `MySQSQueue`.

```
#This sample requires you to create a separate configuration file to define the custom
options for the SNS topic and SQS queue.
Resources:
  MySQSQueue:
    Type: AWS::SQS::Queue
    Properties:
      VisibilityTimeout:
```

```

Fn::GetOptionSetting:
  OptionName: VisibilityTimeout
  DefaultValue: 30
AlarmTopic:
  Type: AWS::SNS::Topic
  Properties:
    Subscription:
      - Endpoint:
          Fn::GetOptionSetting:
            OptionName: AlarmEmail
            DefaultValue: "nobody@amazon.com"
        Protocol: email
QueueDepthAlarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: "Alarm if queue depth grows beyond 10 messages"
    Namespace: "AWS/SQS"
    MetricName: ApproximateNumberOfMessagesVisible
    Dimensions:
      - Name: QueueName
        Value : { "Fn::GetAtt" : [ "MySQSQueue", "QueueName" ] }
    Statistic: Sum
    Period: 300
    EvaluationPeriods: 1
    Threshold: 10
    ComparisonOperator: GreaterThanThreshold
    AlarmActions:
      - Ref: AlarmTopic
    InsufficientDataActions:
      - Ref: AlarmTopic

Outputs :
  QueueURL:
    Description : "URL of newly created SQS Queue"
    Value : { Ref : "MySQSQueue" }
  QueueARN :
    Description : "ARN of newly created SQS Queue"
    Value : { "Fn::GetAtt" : [ "MySQSQueue", "Arn" ] }
  QueueName :
    Description : "Name newly created SQS Queue"
    Value : { "Fn::GetAtt" : [ "MySQSQueue", "QueueName" ] }

```

Pour plus d'informations sur les ressources utilisées dans cet exemple de fichier de configuration, consultez les références suivantes :

- [AWS::SQS::Queue](#)
- [AWS::SNS::Topic](#)
- [AWS::CloudWatch::Alarm](#)

Créez un fichier de configuration distinct nommé `options.config` et définissez les paramètres d'option personnalisés.

```

option_settings:
  "aws:elasticbeanstalk:customoption":
    VisibilityTimeout : 30
    AlarmEmail : "nobody@example.com"

```

Avec ces lignes de code, Elastic Beanstalk récupère les valeurs des propriétés `VisibilityTimeout` et `Subscription Endpoint` à partir des valeurs `VisibilityTimeout` et `Subscription Endpoint` figurant dans un fichier de configuration (`options.config` dans notre exemple). Celui-ci contient une section `option_settings` dont la section `aws:elasticbeanstalk:customoption` inclut une paire nom-valeur qui contient la valeur réelle à utiliser.

Dans l'exemple ci-dessus, cela signifie que les valeurs 30 et « nobody@amazon.com » seraient utilisées. Pour plus d'informations sur Fn::GetOptionSetting, consultez [the section called “Fonctions” \(p. 764\)](#).

## Exemple : DynamoDB, CloudWatch et SNS

Ce fichier de configuration définit la table DynamoDB comme un gestionnaire de session pour une application basée sur PHP à l'aide du kit SDK AWS pour PHP 2. Pour utiliser cet exemple, vous devez disposer d'un profil d'instance IAM, qui est ajouté aux instances de votre environnement et utilisé pour accéder à la table DynamoDB.

Vous pouvez télécharger ici l'exemple que nous utiliserons dans cette étape : [Exemple de prise en charge de session DynamoDB](#). L'exemple contient les fichiers suivants :

- Exemple d'application, `index.php`
- Un fichier de configuration (`dynamodb.config`) qui permet de créer et de configurer une table DynamoDB et d'autres ressources AWS, et d'installer un logiciel sur les instances EC2 qui hébergent l'application dans un environnement Elastic Beanstalk.
- Un fichier de configuration `options.config` qui remplace les valeurs par défaut dans `dynamodb.config` par des paramètres propres à cette installation spécifique

### `index.php`

```
<?php

// Include the SDK using the Composer autoloader
require '../vendor/autoload.php';

use Aws\DynamoDb\DynamoDbClient;

// Grab the session table name and region from the configuration file
list($tableName, $region) = file(__DIR__ . '/../sessiontable');
$tableName = rtrim($tableName);
$region = rtrim($region);

// Create a DynamoDB client and register the table as the session handler
$dynamodb = DynamoDbClient::factory(array('region' => $region));
$handler = $dynamodb->registerSessionHandler(array('table_name' => $tableName, 'hash_key' => 'username'));

// Grab the instance ID so we can display the EC2 instance that services the request
$instanceId = file_get_contents("http://169.254.169.254/latest/meta-data/instance-id");
?>
<h1>Elastic Beanstalk PHP Sessions Sample</h1>
<p>This sample application shows the integration of the Elastic Beanstalk PHP container and the session support for DynamoDB from the AWS SDK for PHP 2. Using DynamoDB session support, the application can be scaled out across multiple web servers. For more details, see the <a href="https://aws.amazon.com/php/">PHP Developer Center</a>. </p>

<form id="SimpleForm" name="SimpleForm" method="post" action="index.php">
<?php
echo 'Request serviced from instance ' . $instanceId . '<br/>';
echo '<br/>';

if (isset($_POST['continue'])) {
    session_start();
    $_SESSION['visits'] = $_SESSION['visits'] + 1;
    echo 'Welcome back ' . $_SESSION['username'] . '<br/>';
    echo 'This is visit number ' . $_SESSION['visits'] . '<br/>';
    session_write_close();
    echo '<br/>';
```

```

echo '<input type="Submit" value="Refresh" name="continue" id="continue"/>';
echo '<input type="Submit" value="Delete Session" name="killsession" id="killsession"/>';
} elseif (isset($_POST['killsession'])) {
    session_start();
    echo 'Goodbye ' . $_SESSION['username'] . '<br/>';
    session_destroy();
    echo 'Username: <input type="text" name="username" id="username" size="30"/><br/>';
    echo '<br/>';
    echo '<input type="Submit" value="New Session" name="newsession" id="newsession"/>';
} elseif (isset($_POST['newsession'])) {
    session_start();
    $_SESSION['username'] = $_POST['username'];
    $_SESSION['visits'] = 1;
    echo 'Welcome to a new session ' . $_SESSION['username'] . '<br/>';
    session_write_close();
    echo '<br/>';
    echo '<input type="Submit" value="Refresh" name="continue" id="continue"/>';
    echo '<input type="Submit" value="Delete Session" name="killsession" id="killsession"/>';
} else {
    echo 'To get started, enter a username.<br/>';
    echo '<br/>';
    echo 'Username: <input type="text" name="username" id="username" size="30"/><br/>';
    echo '<input type="Submit" value="New Session" name="newsession" id="newsession"/>';
}
?>
</form>
```

#### **.ebextensions/dynamodb.config**

```

Resources:
SessionTable:
    Type: AWS::DynamoDB::Table
    Properties:
        KeySchema:
            HashKeyElement:
                AttributeName:
                    Fn::GetOptionSetting:
                        OptionName : SessionHashKeyName
                        DefaultValue: "username"
                AttributeType:
                    Fn::GetOptionSetting:
                        OptionName : SessionHashKeyType
                        DefaultValue: "S"
        ProvisionedThroughput:
            ReadCapacityUnits:
                Fn::GetOptionSetting:
                    OptionName : SessionReadCapacityUnits
                    DefaultValue: 1
            WriteCapacityUnits:
                Fn::GetOptionSetting:
                    OptionName : SessionWriteCapacityUnits
                    DefaultValue: 1

SessionWriteCapacityUnitsLimit:
    Type: AWS::CloudWatch::Alarm
    Properties:
        AlarmDescription: { "Fn::Join" : [ "", [ { "Ref" : "AWSEBEnvironmentName" }, " write capacity limit on the session table." ] ] }
        Namespace: "AWS/DynamoDB"
        MetricName: ConsumedWriteCapacityUnits
        Dimensions:
            - Name: TableName
              Value: { "Ref" : "SessionTable" }
        Statistic: Sum
        Period: 300
```

```
EvaluationPeriods: 12
Threshold:
  Fn::GetOptionSetting:
    OptionName : SessionWriteCapacityUnitsAlarmThreshold
    DefaultValue: 240
  ComparisonOperator: GreaterThanThreshold
  AlarmActions:
    - Ref: SessionAlarmTopic
  InsufficientDataActions:
    - Ref: SessionAlarmTopic

SessionReadCapacityUnitsLimit:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: { "Fn::Join" : [ "", [ { "Ref" : "AWSEBEnvironmentName" }, " read
capacity limit on the session table." ]]}
    Namespace: "AWS/DynamoDB"
    MetricName: ConsumedReadCapacityUnits
    Dimensions:
      - Name: TableName
        Value: { "Ref" : "SessionTable" }
    Statistic: Sum
    Period: 300
  EvaluationPeriods: 12
  Threshold:
    Fn::GetOptionSetting:
      OptionName : SessionReadCapacityUnitsAlarmThreshold
      DefaultValue: 240
    ComparisonOperator: GreaterThanThreshold
    AlarmActions:
      - Ref: SessionAlarmTopic
    InsufficientDataActions:
      - Ref: SessionAlarmTopic

SessionThrottledRequestsAlarm:
  Type: AWS::CloudWatch::Alarm
  Properties:
    AlarmDescription: { "Fn::Join" : [ "", [ { "Ref" : "AWSEBEnvironmentName" }, ":
requests are being throttled." ]]}
    Namespace: AWS/DynamoDB
    MetricName: ThrottledRequests
    Dimensions:
      - Name: TableName
        Value: { "Ref" : "SessionTable" }
    Statistic: Sum
    Period: 300
  EvaluationPeriods: 1
  Threshold:
    Fn::GetOptionSetting:
      OptionName: SessionThrottledRequestsThreshold
      DefaultValue: 1
    ComparisonOperator: GreaterThanThreshold
    AlarmActions:
      - Ref: SessionAlarmTopic
    InsufficientDataActions:
      - Ref: SessionAlarmTopic

SessionAlarmTopic:
  Type: AWS::SNS::Topic
  Properties:
    Subscription:
      - Endpoint:
          Fn::GetOptionSetting:
            OptionName: SessionAlarmEmail
            DefaultValue: "nobody@amazon.com"
    Protocol: email
```

```

files:
  "/var/app/sessiontable":
    mode: "000444"
    content: |
      `{"Ref" : "SessionTable"}`
      `{"Ref" : "AWS::Region"}` 

  "/var/app/composer.json":
    mode: "000744"
    content:
      {
        "require": {
          "aws/aws-sdk-php": "*"
        }
      }

container_commands:
  "1-install-composer":
    command: "cd /var/app; curl -s http://getcomposer.org/installer | php"
  "2-install-dependencies":
    command: "cd /var/app; php composer.phar install"
  "3-cleanup-composer":
    command: "rm -Rf /var/app/composer.*"

```

Dans le modèle de fichier de configuration, nous commençons par créer la table DynamoDB et configurer la structure de clé primaire pour la table et les unités de capacités afin d'allouer des ressources suffisantes pour offrir le débit demandé. Ensuite, nous créons des alarmes CloudWatch pour WriteCapacity et ReadCapacity. Nous créons une rubrique SNS qui envoie un e-mail à « nobody@amazon.com » si le seuil d'alarme est dépassé.

Une fois que nous avons créé et configuré nos ressources AWS pour notre environnement, nous devons personnaliser les instances EC2. Nous utilisons la clé files pour transmettre les détails de la table DynamoDB aux instances EC2 de notre environnement et nous ajoutons une clé « require » dans le fichier composer.json pour le kit SDK AWS pour PHP 2. Enfin, nous exécutons des commandes de conteneur pour installer Composer, les dépendances requises, puis nous supprimons le programme d'installation.

#### **.ebextensions/options.config**

```

option_settings:
  "aws:elasticbeanstalk:customoption":
    SessionHashKeyName : username
    SessionHashKeyType : S
    SessionReadCapacityUnits : 1
    SessionReadCapacityUnitsAlarmThreshold : 240
    SessionWriteCapacityUnits : 1
    SessionWriteCapacityUnitsAlarmThreshold : 240
    SessionThrottledRequestsThreshold : 1
    SessionAlarmEmail : me@example.com

```

Remplacez la valeur SessionAlarmEmail par l'adresse e-mail à laquelle vous souhaitez recevoir les notifications de l'alarme. Le fichier options.config contient les valeurs utilisées pour certaines des variables définies dans dynamodb.config. Par exemple, dynamodb.config contient les lignes suivantes :

```

Subscription:
  - Endpoint:
    Fn::GetOptionSetting:
      OptionName: SessionAlarmEmail
      DefaultValue: "nobody@amazon.com"

```

Avec ces lignes de code, Elastic Beanstalk récupère la valeur de la propriété Endpoint à partir de la valeur SessionAlarmEmail figurant dans un fichier de configuration (`options.config` dans notre exemple d'application). Celui-ci contient une section `option_settings` dont la section `aws:elasticbeanstalk:customoption` inclut une paire nom-valeur qui contient la valeur réelle à utiliser. Dans l'exemple ci-dessus, cela signifie que la valeur `nobody@amazon.com` serait attribuée à `SessionAlarmEmail`.

Pour plus d'informations sur les ressources CloudFormation utilisées dans cet exemple, consultez les références suivantes :

- [AWS::DynamoDB::Table](#)
- [AWS::CloudWatch::Alarm](#)
- [AWS::SNS::Topic](#)

## Utilisation des configurations enregistrées par Elastic Beanstalk

Vous pouvez enregistrer la configuration de votre environnement en tant qu'objet dans Amazon S3 pouvant être appliqué dans d'autres environnements au cours de la création d'environnement, ou appliqués à un environnement en cours d'exécution. Les configurations enregistrées sont des modèles au format YAML qui définissent la [version de plateforme \(p. 31\)](#), le [niveau \(p. 14\)](#), les paramètres d'[option de configuration \(p. 658\)](#) et les balises.

Vous pouvez appliquer les balises à une configuration enregistrée lorsque vous la créez, et modifier des balises existantes de configurations enregistrées. Pour plus d'informations, consultez [Balisage de configurations enregistrées \(p. 783\)](#).

### Note

Les balises appliquées à une configuration enregistrée ne sont pas liées à des balises spécifiées dans une configuration enregistrée à l'aide de la clé `Tags` :. Ces dernières sont appliquées à un environnement lorsque vous appliquez la configuration enregistrée à l'environnement.

Créez une configuration enregistrée à partir de l'état actuel de votre environnement dans la console de gestion Elastic Beanstalk.

Pour enregistrer la configuration d'un environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment action (Actions d'environnement), puis Save configuration (Enregistrer la configuration).
4. Utilisez le formulaire à l'écran pour nommer la configuration enregistrée. Le cas échéant, fournissez une brève description, et ajoutez les clés et valeurs de balise.
5. Choisissez Enregistrer.
6. Choisissez Enregistrer.

## Save Configuration

Save this environment's current configuration.

Environment:

GettingStartedApp-env

Configuration name:

base

Description:

Base configuration

### Tags

Apply up to 50 tags. You can use tags to group and filter your resources. A tag is a key-value pair. The key must be unique within the resource and is case-sensitive. [Learn more](#)

Key

mytag1

Value

value1

[Remove tag](#)

[Add tag](#)

49 remaining

[Cancel](#)

[Save](#)

La configuration enregistrée inclut tous les paramètres que vous avez appliqués à l'environnement avec la console ou tout autre client qui utilise l'API Elastic Beanstalk. Vous pouvez ensuite appliquer la configuration enregistrée à votre environnement à une date ultérieure pour la restaurer à son état antérieur, ou l'appliquer à un nouvel environnement pendant la [création de l'environnement \(p. 442\)](#).

Vous pouvez télécharger une configuration à l'aide de la commande [the section called “eb config” \(p. 1071\)](#) de l'interface de ligne de commande EB, comme illustré dans l'exemple suivant. **NAME** correspond au nom de votre configuration enregistrée.

```
eb config get NAME
```

Pour appliquer une configuration enregistrée lors de la création de l'environnement (console Elastic Beanstalk)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

#### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Saved configurations (Configurations enregistrées).
4. Sélectionnez la configuration enregistrée que vous souhaitez appliquer, puis choisissez Launch environment (Lancer l'environnement).
5. Suivez les étapes de l'assistant pour créer votre environnement.

Les configurations enregistrées ne comprennent pas les paramètres appliqués avec les [fichiers de configuration \(p. 737\)](#) dans le code source de votre application. Si le même paramètre est appliqué à la fois dans un fichier de configuration et dans une configuration enregistrée, le paramètre dans la configuration enregistrée est prioritaire. De même, les options spécifiées dans la console Elastic Beanstalk remplacent les options dans les configurations enregistrées. Pour de plus amples informations, veuillez consulter [Préférence \(p. 659\)](#).

Les configurations enregistrées sont stockées dans le compartiment S3 Elastic Beanstalk, dans un dossier nommé en fonction de votre application. Par exemple, les configurations pour une application nommée `my-app` dans la région `us-west-2` pour le compte n° `123456789012` sont disponibles à l'emplacement `s3://elasticbeanstalk-us-west-2-123456789012/resources/templates/my-app/`.

Affichez le contenu d'une configuration enregistrée en l'ouvrant dans un éditeur de texte. L'exemple de configuration suivant montre la configuration d'un environnement de serveur web lancé avec la console de gestion Elastic Beanstalk.

```
EnvironmentConfigurationMetadata:  
  Description: Saved configuration from a multicontainer Docker environment created with  
    the Elastic Beanstalk Management Console  
  DateCreated: '1520633151000'  
  DateModified: '1520633151000'  
Platform:  
  PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit Amazon  
    Linux/2.5.0  
OptionSettings:  
  aws:elasticbeanstalk:command:  
    BatchSize: '30'  
    BatchSizeType: Percentage  
  aws:elasticbeanstalk:sns:topics:  
    Notification Endpoint: me@example.com  
aws:elb:policies:  
  ConnectionDrainingEnabled: true  
  ConnectionDrainingTimeout: '20'
```

```
aws:elb:loadbalancer:  
  CrossZone: true  
aws:elasticbeanstalk:environment:  
  ServiceRole: aws-elasticbeanstalk-service-role  
aws:elasticbeanstalk:application:  
  Application Healthcheck URL: /  
aws:elasticbeanstalk:healthreporting:system:  
  SystemType: enhanced  
aws:autoscaling:launchconfiguration:  
  IamInstanceProfile: aws-elasticbeanstalk-ec2-role  
  InstanceType: t2.micro  
  EC2KeyName: workstation-uswest2  
aws:autoscaling:updatepolicy:rollingupdate:  
  RollingUpdateType: Health  
  RollingUpdateEnabled: true  
EnvironmentTier:  
  Type: Standard  
  Name: WebServer  
AWSConfigurationTemplateVersion: 1.1.0.0  
Tags:  
  Cost Center: WebApp Dev
```

Vous pouvez modifier le contenu d'une configuration enregistrée et l'enregistrer au même emplacement dans Amazon S3. Toute configuration enregistrée avec le bon format stockée à l'emplacement approprié peut être appliquée à un environnement avec la console de gestion Elastic Beanstalk.

Les clés suivantes sont prises en charge.

- AWSConfigurationTemplateVersion (obligatoire) – La version du modèle de configuration (1.1.0.0).

```
AWSConfigurationTemplateVersion: 1.1.0.0
```

- Plateforme – Amazon Resource Name (ARN) de la version de plateforme de l'environnement. Vous pouvez spécifier la plateforme par ARN ou par nom de pile de solutions.

```
Platform:  
  PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit  
    Amazon Linux/2.5.0
```

- SolutionStack – Le nom complet de la [pile de solution \(p. 31\)](#) utilisée pour créer l'environnement.

```
SolutionStack: 64bit Amazon Linux 2017.03 v2.5.0 running Java 8
```

- OptionSettings – Les paramètres [Configuration option \(Option de configuration\) \(p. 658\)](#) à appliquer à l'environnement. Par exemple, l'entrée suivante définit le type d'instance sur t2.micro.

```
OptionSettings:  
  aws:autoscaling:launchconfiguration:  
    InstanceType: t2.micro
```

- Balises – Jusqu'à 47 balises à appliquer aux ressources créées dans l'environnement.

```
Tags:  
  Cost Center: WebApp Dev
```

- EnvironmentTier – Le type d'environnement à créer. Pour un environnement de serveur web, vous pouvez exclure cette section (le serveur web est la valeur par défaut). Pour un environnement de travail, utilisez les informations suivantes.

```
EnvironmentTier:
```

```
Name: Worker  
Type: SQS/HTTP
```

Consultez les rubriques suivantes pour d'autres méthodes de création et d'application de configurations enregistrées :

- Définition d'options de configuration avant la création de l'environnement (p. 661)
- Définition des options de configuration lors de la création de l'environnement (p. 665)
- Définition des options de configuration après la création de l'environnement (p. 671)

## Balisage de configurations enregistrées

Vous pouvez appliquer des balises à vos configurations AWS Elastic Beanstalk enregistrées. Les balises sont des paires clé-valeur associées aux ressources AWS. Pour de plus amples informations sur le balisage des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur de balise, et les types de ressources pris en charge, veuillez consulter [Balisage des ressources d'application Elastic Beanstalk \(p. 423\)](#).

Vous pouvez spécifier des balises lorsque vous créez une configuration enregistrée. Dans une configuration enregistrée, vous pouvez ajouter ou supprimer des balises, ainsi que mettre à jour les valeurs des balises existantes. Vous pouvez ajouter jusqu'à 50 balises à chaque configuration enregistrée.

### Ajout de balises lors de la création d'une configuration enregistrée

Lorsque vous utilisez la console Elastic Beanstalk pour [enregistrer une configuration \(p. 779\)](#), vous pouvez spécifier des clés et des valeurs de balise sur la page Save Configuration (Enregistrer la configuration).

Si vous utilisez l'interface de ligne de commande EB pour enregistrer une configuration, utilisez l'option `--tags` avec [eb config \(p. 1071\)](#) pour ajouter des balises.

```
~/workspace/my-app$ eb config --tags mytag1=value1,mytag2=value2
```

Avec l'AWS CLI ou les autres clients basés sur l'API, ajoutez des balises en utilisant le paramètre `--tags` sur la commande [create-configuration-template](#).

```
$ aws elasticbeanstalk create-configuration-template \  
  --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \  
  --application-name my-app --template-name my-template --solution-stack-name solution-stack
```

### Gestion des balises d'une configuration enregistrée existante

Vous pouvez ajouter, mettre à jour et supprimer des balises dans une configuration Elastic Beanstalk enregistrée existante.

Pour gérer les balises d'une configuration enregistrée à l'aide de la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Applications, puis sélectionnez le nom de votre application dans la liste.

### Note

Si vous avez plusieurs applications, utilisez la barre de recherche pour filtrer la liste des applications.

3. Dans le volet de navigation, recherchez le nom de votre application et choisissez Saved configurations (Configurations enregistrées).
4. Sélectionnez la configuration enregistrée que vous souhaitez gérer.
5. Choisissez Actions, puis Manage Tags (Gérer les balises).
6. Utilisez le formulaire à l'écran pour ajouter, mettre à jour ou supprimer des balises.
7. Choisissez Apply (Appliquer).

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour votre configuration enregistrée, utilisez [eb tags \(p. 1118\)](#) pour ajouter, mettre à jour, supprimer ou répertorier des balises.

Par exemple, la commande suivante répertorie les balises dans une configuration enregistrée.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-template"
```

La commande suivante met à jour la balise mytag1 et supprime la balise mytag2.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \  
--resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-template"
```

Pour obtenir une liste complète des options et d'autres exemples, consultez [eb tags \(p. 1118\)](#).

Avec l'AWS CLI ou d'autres clients basés sur l'API, utilisez la commande [list-tags-for-resource](#) pour afficher les balises d'une configuration enregistrée.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-template"
```

Utilisez la commande [update-tags-for-resource](#) pour ajouter, mettre à jour ou supprimer des balises dans une configuration enregistrée.

```
$ aws elasticbeanstalk update-tags-for-resource \  
--tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \  
--resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-template"
```

Spécifiez les balises à ajouter et les balises à mettre à jour dans le paramètre --tags-to-add de update-tags-for-resource. Une balise inexistante est ajoutée et la valeur d'une balise existante est mise à jour.

### Note

Pour utiliser certaines commandes de l'interface de ligne de commande EB et de l'AWS CLI avec une configuration Elastic Beanstalk enregistrée, vous avez besoin de l'ARN de la configuration enregistrée. Pour construire les ARN, vous devez d'abord utiliser la commande suivante pour récupérer la configuration enregistrée.

```
$ aws elasticbeanstalk describe-applications --application-names my-app
```

Recherchez la clé ConfigurationTemplates dans la sortie de la commande. Cet élément affiche le nom de la configuration enregistrée. Utilisez ce nom où **my-template** est spécifié dans les commandes mentionnées sur cette page.

## Manifeste d'environnement (env.yaml)

Vous pouvez inclure un manifeste d'environnement au format YAML à la racine du groupe source de votre application pour configurer le nom de l'environnement, la pile de solutions et les [liens d'environnement](#) (p. 530) à utiliser lors de la création de votre environnement.

Ce format de fichier inclut la prise en charge pour les groupes de l'environnement. Pour utiliser des groupes, précisez le nom de l'environnement dans le manifeste avec un symbole + à la fin. Lorsque vous créez ou mettez à jour l'environnement, spécifiez le nom du groupe avec --group-name (AWS CLI) ou --env-group-suffix (interface de ligne de commande EB). Pour de plus amples informations sur les groupes, veuillez consulter [Création et mise à jour de groupes d'environnements Elastic Beanstalk](#) (p. 474).

L'exemple de manifeste suivant définit un environnement de serveur web avec un lien vers un composant d'environnement de travail dont il dépend. Le manifeste utilise des groupes pour autoriser la création de plusieurs environnements avec le même groupe source :

**~/myapp/frontend/env.yaml**

```
AWSConfigurationTemplateVersion: 1.1.0.0
SolutionStack: 64bit Amazon Linux 2015.09 v2.0.6 running Multi-container Docker 1.7.1
(Generic)
OptionSettings:
  aws:elasticbeanstalk:command:
    BatchSize: '30'
    BatchSizeType: Percentage
  aws:elasticbeanstalk:sns:topics:
    Notification Endpoint: me@example.com
  aws:elb:policies:
    ConnectionDrainingEnabled: true
    ConnectionDrainingTimeout: '20'
  aws:elb:loadbalancer:
    CrossZone: true
  aws:elasticbeanstalk:environment:
    ServiceRole: aws-elasticbeanstalk-service-role
  aws:elasticbeanstalk:application:
    Application Healthcheck URL: /
  aws:elasticbeanstalk:healthreporting:system:
    SystemType: enhanced
  aws:autoscaling:launchconfiguration:
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role
    InstanceType: t2.micro
    EC2KeyName: workstation-uswest2
  aws:autoscaling:updatepolicy:rollingupdate:
    RollingUpdateType: Health
    RollingUpdateEnabled: true
Tags:
  Cost Center: WebApp Dev
  CName: front-A08G28LG+
  EnvironmentName: front+
  EnvironmentLinks:
    "WORKERQUEUE" : "worker+"

```

Les clés suivantes sont prises en charge.

- AWSConfigurationTemplateVersion (obligatoire) – La version du modèle de configuration (1.1.0.0).

```
AWSConfigurationTemplateVersion: 1.1.0.0
```

- Plateforme – Amazon Resource Name (ARN) de la version de plateforme de l'environnement. Vous pouvez spécifier la plateforme par ARN ou par nom de pile de solutions.

```
Platform:  
  PlatformArn: arn:aws:elasticbeanstalk:us-east-2::platform/Java 8 running on 64bit  
    Amazon Linux/2.5.0
```

- SolutionStack – Le nom complet de la [pile de solution \(p. 31\)](#) utilisée pour créer l'environnement.

```
SolutionStack: 64bit Amazon Linux 2017.03 v2.5.0 running Java 8
```

- OptionSettings – Les paramètres [Configuration option \(Option de configuration\) \(p. 658\)](#) à appliquer à l'environnement. Par exemple, l'entrée suivante définit le type d'instance sur t2.micro.

```
OptionSettings:  
  aws:autoscaling:launchconfiguration:  
    InstanceType: t2.micro
```

- Balises – Jusqu'à 47 balises à appliquer aux ressources créées dans l'environnement.

```
Tags:  
  Cost Center: WebApp Dev
```

- EnvironmentTier – Le type d'environnement à créer. Pour un environnement de serveur web, vous pouvez exclure cette section (le serveur web est la valeur par défaut). Pour un environnement de travail, utilisez les informations suivantes.

```
EnvironmentTier:  
  Name: Worker  
  Type: SQS/HTTP
```

- CName – CNAME pour l'environnement. Incluez un caractère + à la fin du nom pour activer des groupes.

```
CName: front-A08G28LG+
```

- EnvironmentName – Le nom de l'environnement à créer. Incluez un caractère + à la fin du nom pour activer des groupes.

```
EnvironmentName: front+
```

Avec les groupes activés, vous devez spécifier un nom de groupe lorsque vous créez les environnements. Elastic Beanstalk ajoute le nom du groupe au nom de l'environnement avec un trait d'union. Par exemple, avec le nom d'environnement `front+` et le nom de groupe `dev`, Elastic Beanstalk crée l'environnement nommé `front-dev`.

- EnvironmentLinks – Une carte de noms de variables et de noms d'environnements des dépendances. L'exemple suivant fait de l'environnement `worker+` une dépendance et indique à Elastic Beanstalk d'enregistrer les informations de lien sur une variable nommée `WORKERQUEUE`.

```
EnvironmentLinks:  
  "WORKERQUEUE" : "worker+"
```

La valeur de la variable de lien varie en fonction du type de l'environnement lié. Pour un environnement de serveur web, le lien est le CNAME de l'environnement. Pour un environnement de travail, le lien est le nom de la file d'attente Amazon Simple Queue Service (Amazon SQS) de l'environnement.

Les clés CName, EnvironmentName et EnvironmentLinks peuvent être utilisées pour créer des [groupes d'environnement \(p. 474\)](#) et des [liens vers d'autres environnements \(p. 530\)](#). Ces fonctionnalités sont

actuellement prises en charge lors de l'utilisation de l'interface de ligne de commande EB, de l'AWS CLI ou d'un SDK.

## Utilisation d'une Amazon Machine Image (AMI) personnalisée

Lorsque vous créez un environnement AWS Elastic Beanstalk, vous pouvez spécifier une Amazon Machine Image (AMI) à utiliser à la place de l'AMI Elastic Beanstalk standard incluse dans votre version de plateforme. Une AMI personnalisée peut améliorer le temps de mise en service lorsque les instances sont lancées dans votre environnement, si vous avez besoin d'installer de nombreux logiciels qui ne sont pas inclus dans les AMI standards.

Nous vous recommandons d'utiliser des [fichiers de configuration \(p. 737\)](#) pour configurer et personnaliser votre environnement rapidement et assurer la cohérence. Toutefois, l'application de configurations peut prendre beaucoup de temps lors de la création et des mises à jour de l'environnement. Si vous passez beaucoup de temps sur la configuration serveur dans les fichiers de configuration, vous pouvez résoudre le problème en créant une AMI personnalisée qui dispose déjà du logiciel et de la configuration dont vous avez besoin.

En outre, une AMI personnalisée vous permet d'apporter des modifications aux composants de bas niveau, tels que le noyau Linux, qui sont difficiles à mettre en œuvre et dont l'application dans les fichiers de configuration prend beaucoup de temps. Pour créer une AMI personnalisée, lancez une AMI de plateforme Elastic Beanstalk dans Amazon EC2, personnalisez le logiciel et la configuration selon vos besoins, puis arrêtez l'instance et enregistrez une AMI à partir de cette dernière.

## Création d'une AMI personnalisée

Pour identifier l'AMI Elastic Beanstalk de base

1. Dans une fenêtre de commande, exécutez une commande similaire à celle-ci. Pour plus d'informations, consultez la section [describe-platform-version](#) du Guide de référence des commandes de l'AWS CLI.

Spécifiez la région AWS dans laquelle vous souhaitez utiliser votre AMI personnalisée et remplacez le numéro de version de plateforme et l'ARN par la plateforme Elastic Beanstalk sur laquelle est basée votre application.

Example - Système d'exploitation Mac / Linux

```
$ aws elasticbeanstalk describe-platform-version --region us-east-2 \
    --platform-arn "arn:aws:elasticbeanstalk:us-east-2::platform/Tomcat 8.5 with Java
8 running on 64bit Amazon Linux/3.1.6" \
    --query PlatformDescription.CustomAmiList
[
    {
        "VirtualizationType": "pv",
        "ImageId": ""
    },
    {
        "VirtualizationType": "hvm",
        "ImageId": "ami-020ae06fdda6a0f66"
    }
]
```

### Example - Système d'exploitation Windows

```
C:\> aws elasticbeanstalk describe-platform-version --region us-east-2 --platform-arn="arn:aws:elasticbeanstalk:us-east-2::platform/IIS 10.0 running on 64bit Windows Server 2019/2.6.4" --query PlatformDescription.CustomAmiList
[
    {
        "VirtualizationType": "pv",
        "ImageId": ""
    },
    {
        "VirtualizationType": "hvm",
        "ImageId": "ami-020ae06fdda6a0f66"
    }
]
```

2. Notez la valeur `ImageId` qui ressemble à `ami-020ae06fdda6a0f66` dans le résultat.

La valeur est l'AMI Elastic Beanstalk standard pour la version de plateforme, l'architecture de l'instance EC2 et la région AWS qui s'avèrent pertinentes pour votre application. Si vous souhaitez créer des AMI pour plusieurs plateformes, architectures ou régions AWS, répétez la procédure afin d'identifier l'AMI de base appropriée pour chaque combinaison.

#### Remarques

- Ne créez pas d'AMI à partir d'une instance qui a été lancée dans un environnement Elastic Beanstalk. Elastic Beanstalk apporte des modifications aux instances au cours de la mise en service, ce qui peut générer des problèmes dans l'AMI enregistrée. En outre, si vous enregistrez une image à partir d'une instance d'un environnement Elastic Beanstalk, la version de votre application ayant été déployée dans l'instance sera transformée en partie fixe de l'image.
- Nous vous recommandons de toujours utiliser la dernière version de la plateforme. Lorsque vous effectuez une mise à jour vers une nouvelle version de plateforme, nous vous recommandons également de rebaser votre AMI personnalisée vers l'AMI de la nouvelle version de plateforme. Cela réduit les échecs de déploiement dus à des versions de package ou de bibliothèque incompatibles.

Pour Linux, vous pouvez également créer une AMI personnalisée à partir d'une AMI de la communauté qui n'a pas été publiée par Elastic Beanstalk. Vous pouvez utiliser la dernière AMI [Linux Amazon](#) comme point de départ. Lorsque vous lancez un environnement avec une AMI Linux non gérée par Elastic Beanstalk, Elastic Beanstalk tente d'installer des logiciels de plateforme (langage, cadre, serveur proxy, etc.) et des composants supplémentaires pour prendre en charge des fonctionnalités telles que les [rapports sur l'état amélioré \(p. 837\)](#).

#### Note

Les AMI personnalisées basées sur Windows Server nécessitent l'AMI Elastic Beanstalk standard renvoyée par `describe-platform-version`, comme indiqué précédemment à l'étape 1.

Bien qu'Elastic Beanstalk puisse utiliser une AMI qui n'est pas gérée par Elastic Beanstalk, l'augmentation du temps de mise en service résultant de l'installation de composants manquants par Elastic Beanstalk peut réduire ou éliminer les avantages de la création d'une AMI personnalisée. D'autres distributions Linux peuvent fonctionner, avec des solutions de dépannage, mais elles ne sont pas officiellement prises en charge. Si votre application nécessite une distribution Linux spécifique, une alternative consiste à créer une image Docker et à l'exécuter sur la [plateforme Docker \(p. 48\)](#) ou la [plateforme Docker multi-conteneurs \(p. 64\)](#) d'Elastic Beanstalk.

#### Pour créer une AMI personnalisée

1. Ouvrez la console Amazon EC2 à l'adresse <https://console.aws.amazon.com/ec2/>.

2. Choisissez Launch Instances.
3. Choisissez AMI de la communauté.
4. Si vous avez identifié une AMI Elastic Beanstalk de base (en utilisant `describe-platform-version`) ou une AMI Amazon Linux, entrez son ID dans la zone de recherche. Appuyez ensuite sur Entrée.

Vous pouvez également explorer la liste d'une autre AMI de communauté qui répond à vos besoins.

Note

Nous vous recommandons de choisir une AMI qui utilise la virtualisation HVM. Ces AMI indiquent Type de virtualisation : hvm dans leur description.



Pour plus d'informations sur les types de virtualisation d'instance, consultez la section [Types de virtualisation d'AMI Linux](#) du Guide de l'utilisateur Amazon EC2 pour les instances Linux ou la section [Types de virtualisation d'AMI Windows](#) du Guide de l'utilisateur Amazon EC2 pour les instances Windows.

5. Choisissez Sélectionner pour sélectionner l'AMI.
6. Sélectionnez un type d'instance, puis choisissez Suivant : Configurer les détails de l'instance.
7. (Pour les plateformes Linux) Développez la section Advanced Details (Détails avancés) et saisissez le texte suivant dans le champ User Data (Données utilisateur).

```
#cloud-config
repo_releasever: repository version number
repo_upgrade: none
```

Le numéro de version du référentiel correspond à l'année et au mois de la version dans le nom de l'AMI. Par exemple, les AMI basées sur la version de mars 2015 d'Amazon Linux comportent le numéro de version de référentiel suivant : 2015 . 03. Pour une image Elastic Beanstalk, cela correspond à la date indiquée dans le nom de la pile de solution pour votre [version de plateforme](#) (p. 31) basée sur l'AMI Amazon Linux (antérieure à Amazon Linux 2).

Note

Ce paramètre `repo_releasever` configure la fonction de verrouillage au lancement d'une AMI Amazon Linux. L'AMI utilise alors une version de référentiel fixe et spécifique lors de son lancement. Cette fonctionnalité n'est pas prise en charge sur Amazon Linux 2. Ne la spécifiez pas si votre environnement utilise une branche de plateforme Amazon Linux 2 actuelle.

Ce paramètre est requis si vous utilisez une AMI personnalisée avec Elastic Beanstalk uniquement sur les branches de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2).

Ce paramètre `repo_upgrade` désactive l'installation automatique des mises à jour de sécurité. Il est nécessaire d'utiliser une AMI personnalisée avec Elastic Beanstalk.

8. Suivez les étapes de l'assistant pour [lancer l'instance EC2](#). Lorsque vous y êtes invité, sélectionnez une paire de clés à laquelle vous avez accès afin de pouvoir vous connecter à l'instance pour suivre les prochaines étapes.
9. [Connectez-vous à l'instance](#) avec SSH ou RDP.
10. Effectuez toutes les personnalisations de votre choix.

11. (Plateformes Windows) Exécutez Sysprep avec le service EC2Config. Pour de plus amples informations sur EC2Config, veuillez consulter [Configuration d'une instance Windows à l'aide du service EC2Config](#). Assurez-vous que Sysprep est configuré pour générer un mot de passe aléatoire pouvant être extrait de AWS Management Console.
12. Dans la console Amazon EC2, arrêtez l'instance EC2. Ensuite, dans le menu Instance Actions (Actions d'instance), choisissez Crée l'image (EBS AMI).
13. Pour éviter de payer des frais AWS supplémentaires, [arrêtez l'instance EC2](#).

Pour utiliser votre AMI personnalisée dans un environnement Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

**Note**

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Capacity (Capacité), choisissez Edit (Modifier).
5. Pour l'ID d'AMI, entrez votre ID d'AMI personnalisé.
6. Choisissez Apply.

Lorsque vous créez un nouvel environnement avec l'AMI personnalisée, vous devez utiliser la même version de plateforme que celle utilisée comme base pour créer l'AMI. Si vous appliquez ultérieurement une [mise à jour de plateforme \(p. 496\)](#) à un environnement à l'aide d'une AMI personnalisée, Elastic Beanstalk tente d'appliquer les mises à jour de configuration et des bibliothèques pendant le processus d'action d'amorçage.

## Nettoyage d'une AMI personnalisée

Lorsque vous avez fini d'utiliser une AMI personnalisée et que vous n'en avez plus besoin pour lancer des environnements Elastic Beanstalk, nettoyez-la afin de réduire les coûts de stockage. Le nettoyage d'une AMI personnalisée implique l'annulation de son enregistrement dans Amazon EC2 et la suppression des autres ressources associées. Pour de plus amples informations, veuillez consulter [Annulation de l'inscription de votre AMI Linux](#) ou [Annuler l'inscription de votre AMI Windows](#).

## Service de fichiers statiques

Pour améliorer les performances, vous pouvez configurer le serveur proxy pour proposer des fichiers statiques (HTML ou images, par exemple) à partir d'un ensemble de répertoires dans votre application web. Lorsque le serveur proxy reçoit une demande pour un fichier dans le chemin spécifié, il fournit le fichier directement au lieu d'acheminer la demande vers votre application.

Elastic Beanstalk prend en charge la configuration du proxy pour servir des fichiers statiques sur la plupart des branches de plate-forme basées sur Amazon Linux 2. La seule exception est Docker.

**Note**

Sur les plates-formes Python et Ruby, Elastic Beanstalk configure certains dossiers de fichiers statiques par défaut. Pour plus de détails, consultez les sections de configuration de

fichiers statiques pour [Python \(p. 361\)](#) et [Ruby \(p. 388\)](#). Vous pouvez configurer des dossiers supplémentaires comme expliqué sur cette page.

## Configurer les fichiers statiques à l'aide de la console

Pour configurer le serveur proxy afin de servir des fichiers statiques

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Dans la section Fichiers statiques, entrez un chemin d'accès pour le service des fichiers statiques et le répertoire des fichiers statiques à servir dans la ligne vide en bas de la liste.

### Note

Si la section Static Files (fichiers statiques) n'apparaît pas, vous devez ajouter au moins un mappage à l'aide d'un [fichier de configuration \(p. 737\)](#). Pour plus de détails, consultez [the section called “Configurer des fichiers statiques à l'aide des options de configuration” \(p. 792\)](#) sur cette page.

Commencez le chemin d'accès par une barre oblique (/). Spécifiez un nom de répertoire à la racine du code source de votre application ; ne le commencez pas par une barre oblique.

Lorsque vous ajoutez un mappage, une ligne supplémentaire s'affiche au cas où vous souhaiteriez en ajouter une autre. Pour supprimer un mappage, cliquez sur l'icône Supprimer.

### Static files

Configure the proxy server to serve static files to reduce the request load on your application. [Learn more](#)

Path (Example: /assets)	Directory (Example: /static/assets)
/html	statichtml 
/images	staticimages 

6. Choisissez Apply.

## Configurer des fichiers statiques à l'aide des options de configuration

Vous pouvez utiliser un [fichier de configuration \(p. 737\)](#) pour configurer des chemins de fichiers statiques et des emplacements de répertoires à l'aide des options de configuration. Vous pouvez ajouter un fichier de configuration au bundle source de votre application et le déployer lors de la création de l'environnement ou d'un déploiement ultérieur.

Si votre environnement utilise une branche de plateforme basée sur Amazon Linux 2, utilisez l'espace de noms `aws:elasticbeanstalk:environment:proxy:staticfiles` ([p. 706](#)).

L'exemple de fichier de configuration suivant indique au serveur proxy de servir les fichiers du dossier `statichtml` sur le chemin `/html` et les fichiers du dossier `staticimages` sur le chemin `/images`.

Example `.ebextensions/static-files.config`

```
option_settings:  
  aws:elasticbeanstalk:environment:proxy:staticfiles:  
    /html: statichtml  
    /images: staticimages
```

Si votre environnement Elastic Beanstalk utilise une version de plateforme AMI Amazon Linux (antérieure à Amazon Linux 2), lisez les informations supplémentaires suivantes :

### Espaces de noms spécifiques à la plateforme AMI Amazon Linux

Sur les branches de la plateforme AMI Amazon Linux, les espaces de noms de configuration de fichiers statiques varient selon la plateforme. Pour plus d'informations, consultez l'une des pages suivantes :

- [Espaces de noms de la configuration Go \(p. 103\)](#)
- [Espaces de noms de la configuration Java SE \(p. 131\)](#)
- [Espaces de noms de la configuration Tomcat \(p. 121\)](#)
- [Espaces de noms de la configuration Node.js \(p. 257\)](#)
- [Espaces de noms de la configuration Python \(p. 361\)](#)

## Configuration de HTTPS pour votre environnement Elastic Beanstalk

Si vous avez acheté et configuré un [nom de domaine personnalisé \(p. 656\)](#) pour votre environnement Elastic Beanstalk, vous pouvez utiliser HTTPS pour permettre aux utilisateurs de se connecter à votre site web en toute sécurité. Si vous ne possédez pas un nom de domaine, vous pouvez toujours utiliser HTTPS avec un certificat auto-signé à des fins de développement et de test. HTTPS est une nécessité pour n'importe quelle application qui transmet des informations de connexion ou des données utilisateur.

La façon la plus simple d'utiliser HTTPS avec un environnement Elastic Beanstalk consiste à [attribuer un certificat serveur à l'équilibrEUR de charge de votre environnement \(p. 797\)](#). Lorsque vous configurez votre équilibrEUR de charge pour résilier HTTPS, la connexion entre le client et l'équilibrEUR de charge est sécurisée. Les connexions principales entre l'équilibrEUR de charge et les instances EC2 utilisent HTTP. Ainsi, aucune configuration supplémentaire des instances n'est requise.

### Note

Avec [AWS Certificate Manager \(ACM\)](#), vous pouvez créer gratuitement un certificat de confiance pour vos noms de domaine. Les certificats ACM ne peuvent être utilisés qu'avec des équilibrEURS

de charge AWS et des distributions Amazon CloudFront, et ACM est [disponible uniquement dans certaines régions AWS](#).

Pour utiliser un certificat ACM avec Elastic Beanstalk, veuillez consulter [Configuration de l'équilibrer de charge de votre environnement Elastic Beanstalk pour résilier les connexions HTTPS \(p. 797\)](#).

Si vous exécutez votre application dans un environnement d'instance unique, ou que vous avez besoin de sécuriser la connexion intégralement vers les instances EC2 derrière l'équilibrer de charge, vous pouvez [configurer le serveur proxy qui s'exécute sur l'instance pour résilier HTTPS \(p. 799\)](#). La configuration de vos instances pour résilier des connexions HTTPS nécessite l'utilisation de [fichiers de configuration \(p. 737\)](#) pour modifier le logiciel s'exécutant sur les instances et pour modifier les groupes de sécurité pour autoriser des connexions sécurisées.

Pour un HTTPS intégral dans un environnement à charge équilibrée, vous pouvez [combiner la résiliation instance et équilibrer de charge \(p. 824\)](#) pour chiffrer les deux connexions. Par défaut, si vous configurez l'équilibrer de charge pour acheminer le trafic à l'aide de HTTPS, il va faire confiance à n'importe quel certificat qui lui est présenté par les instances principales. Pour une sécurité optimale, vous pouvez attacher des stratégies à l'équilibrer de charge qui l'empêchent de se connecter aux instances qui ne présentent pas un certificat public auquel il fait confiance.

#### Note

Vous pouvez également configurer l'équilibrer de charge pour [relayer le trafic HTTPS sans le déchiffrer \(p. 827\)](#). L'inconvénient de cette méthode est que l'équilibrer de charge ne peut pas voir les demandes et ne peut donc pas optimiser le routage ou signaler les métriques de réponse.

Si ACM n'est pas disponible dans votre région, vous pouvez acheter un certificat de confiance auprès d'un tiers. Un certificat tiers peut être utilisé pour déchiffrer le trafic HTTPS au niveau de votre équilibrer de charge, sur les instances backend, ou les deux.

Pour le développement et les tests, vous pouvez [créer et signer un certificat \(p. 794\)](#) vous-même avec des outils open source. Les certificats auto-signés sont gratuits et faciles à créer, mais ne peuvent pas être utilisés pour le déchiffrement frontal sur les sites publics. Si vous essayez d'utiliser un certificat auto-signé pour une connexion HTTPS à un client, le navigateur de l'utilisateur affiche une erreur indiquant que votre site web n'est pas sécurisé. Vous pouvez, toutefois, utiliser un certificat auto-signé pour sécuriser les connexions principales sans problème.

ACM est l'outil le plus adéquat pour allouer, gérer et déployer vos certificats de serveur par programmation ou à l'aide de l'AWS CLI. Si ACM n'est pas [disponible dans votre région AWS](#), vous pouvez [charger un certificat tiers ou auto-signé et une clé privée \(p. 796\)](#) dans AWS Identity and Access Management (IAM) à l'aide de l'AWS CLI. Les certificats stockés dans IAM peuvent être utilisés avec les équilibreurs de charge et les distributions CloudFront.

#### Note

L'exemple d'application [Does it have Snakes?](#) sur GitHub inclut des instructions et des fichiers de configuration pour chaque méthode de configuration de HTTPS avec une application web Tomcat. Consultez le [fichier readme](#) et les [instructions HTTPS](#) pour plus de détails.

#### Rubriques

- [Création et signature d'un certificat X509 \(p. 794\)](#)
- [Chargement d'un certificat dans IAM \(p. 796\)](#)
- [Configuration de l'équilibrer de charge de votre environnement Elastic Beanstalk pour résilier les connexions HTTPS \(p. 797\)](#)
- [Configuration de votre application pour suspendre des connexions HTTPS sur l'instance \(p. 799\)](#)
- [Configuration du chiffrement de bout en bout dans un environnement Elastic Beanstalk à charge équilibrée \(p. 824\)](#)

- Configuration de l'équilibrage de charge de votre environnement pour TCP Passthrough (p. 827)
- Stockage sécurisé des clés privées dans Amazon S3 (p. 828)
- Configuration de la redirection HTTP vers HTTPS (p. 829)

## Création et signature d'un certificat X509

Vous pouvez créer un certificat X509 pour votre application avec OpenSSL. OpenSSL est une bibliothèque open source standard qui prend en charge un large éventail de fonctions de chiffrement, dont la création et la signature de certificats X509. Pour plus d'informations sur OpenSSL, consultez le site <http://www.openssl.org>.

### Note

Vous ne devez créer un certificat localement que si vous souhaitez utiliser HTTPS dans un environnement à instance unique (p. 799) ou effectuer un nouveau chiffrement sur le backend (p. 824) avec un certificat auto-signé. Si vous possédez un nom de domaine, vous pouvez créer un certificat dans AWS et l'utiliser gratuitement dans l'environnement à charge équilibrée à l'aide d'AWS Certificate Manager (ACM). Pour obtenir les instructions, veuillez consulter Demande de certificat dans le Guide de l'utilisateur AWS Certificate Manager.

Exécutez `openssl version` dans la ligne de commande pour voir si OpenSSL est déjà installé. Si ce n'est pas le cas, vous pouvez créer et installer le code source en suivant les instructions du [référentiel GitHub public](#), ou utiliser votre gestionnaire de package préféré. OpenSSL est également installé sur les images Linux d'Elastic Beanstalk. À titre de solution alternative rapide, vous pouvez vous connecter à une instance EC2 dans un environnement en cours d'exécution via la commande `eb ssh` de l'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) :

```
~/eb$ eb ssh
[ec2-user@ip-255-55-55-255 ~]$ openssl version
OpenSSL 1.0.1k-fips 8 Jan 2015
```

Vous devez créer une clé privée RSA pour générer votre demande de signature de certificat (CSR). Pour créer votre clé privée, utilisez la commande `openssl genrsa` :

```
[ec2-user@ip-255-55-55-255 ~]$ openssl genrsa 2048 > privatekey.pem
Generating RSA private key, 2048 bit long modulus
.
.
.
e is 65537 (0x10001)
```

**privatekey.pem**

Nom du fichier dans lequel vous souhaitez enregistrer la clé privée. Généralement, la commande `openssl genrsa` affiche le contenu de la clé privée à l'écran, mais cette commande redirige le résultat vers un fichier. Choisissez un nom de fichier et stockez le fichier dans un emplacement sécurisé de façon à pouvoir le récupérer ultérieurement. Si vous perdez votre clé privée, vous ne pourrez pas utiliser votre certificat.

Un fichier CSR est un fichier que vous pouvez envoyer à une autorité de certification (CA) pour demander un certificat de serveur numérique. Pour créer un fichier CSR, utilisez la commande `openssl req` :

```
$ openssl req -new -key privatekey.pem -out csr.pem
You are about to be asked to enter information that will be incorporated
```

into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
For some fields there will be a default value,  
If you enter '.', the field will be left blank.

Saisissez les informations demandées et appuyez sur Entrée. Le tableau suivant décrit les champs et fournit des exemples pour chacun d'entre eux.

Nom	Description	Exemple
Nom du pays	Abréviation ISO de deux lettres de votre pays.	US = États-Unis
État ou Province	Nom de l'état ou de la province où votre organisation se situe. Vous ne pouvez pas abréger ce nom.	Washington
Nom de la localité	Nom de la ville où votre organisation se situe.	Seattle
Nom de l'organisation	Nom légal complet de votre organisation. N'abrégez pas le nom de votre organisation.	Exemple d'entreprise
Unité d'organisation	(Facultatif) Informations supplémentaires sur l'organisation.	Marketing
Nom commun	Nom de domaine complet de votre site web. Il doit correspondre au nom de domaine que les utilisateurs voient lorsqu'ils visitent votre site (sinon, des erreurs de certificat s'affichent).	www.exemple.com
Adresse e-mail	Adresse e-mail de l'administrateur du site.	quelquun@example.com

Vous pouvez envoyer la demande de signature à un tiers pour la signature, ou la signer vous-même à des fins de test et de développement. Les certificats auto-signés peuvent aussi être utilisés pour une connexion backend HTTPS entre un équilibrEUR de charge et des instances EC2.

Pour signer le certificat, utilisez la commande openssl x509. L'exemple suivant utilise la clé privée générée lors de l'étape précédente (**privatekey.pem**) et la demande de signature (**csr.pem**) pour créer un certificat public nommé **public.crt**, dont la durée de validité est de **365** jours.

```
$ openssl x509 -req -days 365 -in csr.pem -signkey privatekey.pem -out public.crt
Signature ok
subject=/C=us/ST=washington/L=seattle/O=example corporation/OU=marketing/
CN=www.example.com/emailAddress=someone@example.com
Getting Private key
```

Conservez la clé privée et le certificat public pour une utilisation ultérieure. Vous pouvez supprimer la demande de signature. Veillez à toujours stocker la clé privée dans un emplacement sécurisé (p. 828) et à ne pas l'ajouter à votre code source.

Pour utiliser le certificat avec la plateforme Windows Server, vous devez le convertir au format PFX. Utilisez la commande suivante pour créer un certificat PFX à partir de la clé privée et des fichiers de certificat public :

```
$ openssl pkcs12 -export -out example.com.pfx -inkey privatekey.pem -in public.crt
Enter Export Password: password
Verifying - Enter Export Password: password
```

Maintenant que vous disposez d'un certificat, vous pouvez [le charger dans IAM \(p. 796\)](#) pour l'utiliser avec un équilibrEUR de charge, ou [configurer les instances de votre environnement pour suspendre HTTPS \(p. 799\)](#).

## Chargement d'un certificat dans IAM

Pour utiliser votre certificat avec l'équilibrEUR de charge de votre environnement Elastic Beanstalk, chargez le certificat et la clé privée dans AWS Identity and Access Management (IAM). Vous pouvez utiliser un certificat stocké dans IAM avec les équilibrEURS de charge Elastic Load Balancing et les distributions Amazon CloudFront.

### Note

AWS Certificate Manager (ACM) est l'outil préféré pour mettre en service, gérer et déployer vos certificats de serveur. Pour plus d'informations sur une demande de certificat ACM, consultez [Demande de certificat](#) dans le Guide de l'utilisateur AWS Certificate Manager. Pour plus d'informations sur l'importation de certificats tiers dans ACM, consultez [Importation de certificats](#) dans le Guide de l'utilisateur AWS Certificate Manager. Utilisez IAM pour charger un certificat seulement si ACM n'est pas [disponible dans votre région AWS](#).

Vous pouvez utiliser le [AWS Command Line Interface \(p. 1026\)](#) (AWS CLI) pour charger votre certificat. La commande suivante charge un certificat auto-signé nommé `https-cert.crt` avec une clé privée nommée `private-key.pem`:

```
$ aws iam upload-server-certificate --server-certificate-name elastic-beanstalk-x509 --  
certificate-body file://https-cert.crt --private-key file://private-key.pem  
{  
    "ServerCertificateMetadata": {  
        "ServerCertificateId": "AS5YEEIONO2Q7CAIHKNGC",  
        "ServerCertificateName": "elastic-beanstalk-x509",  
        "Expiration": "2017-01-31T23:06:22Z",  
        "Path": "/",  
        "Arn": "arn:aws:iam::123456789012:server-certificate/elastic-beanstalk-x509",  
        "UploadDate": "2016-02-01T23:10:34.167Z"  
    }  
}
```

Le préfixe `file://` indique à l'AWS CLI qu'elle doit charger le contenu d'un fichier dans le répertoire actuel. `elastic-beanstalk-x509` spécifie le nom affecté au certificat dans IAM.

Si vous avez acheté un certificat auprès d'une autorité de certification et reçu un fichier de chaîne de certificats, chargez-le également en incluant l'option `--certificate-chain`:

```
$ aws iam upload-server-certificate --server-certificate-name elastic-beanstalk-x509 --  
certificate-chain file://certificate-chain.pem --certificate-body file://https-cert.crt --  
private-key file://private-key.pem
```

Notez l'Amazon Resource Name (ARN) de votre certificat. Vous l'utiliserez lors de la mise à jour des paramètres de configuration de votre équilibrEUR de charge pour utiliser HTTPS.

### Note

Un certificat chargé dans IAM reste stocké même s'il n'est plus utilisé dans aucun équilibrEUR de charge de l'environnement. Il contient des données sensibles. Lorsque vous n'avez plus besoin du certificat pour aucun environnement, veillez à le supprimer. Pour de plus amples informations sur la suppression d'un certificat dans IAM, veuillez consulter [https://docs.aws.amazon.com/IAM/latest/UserGuide/id\\_credentials\\_server-certs.html#delete-server-certificate](https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_server-certs.html#delete-server-certificate).

Pour de plus amples informations sur les certificats de serveur dans IAM, veuillez consulter [Utilisation des certificats de serveur](#) dans le Guide de l'utilisateur IAM.

## Configuration de l'équilibrer de charge de votre environnement Elastic Beanstalk pour résilier les connexions HTTPS

Pour mettre à jour votre environnement AWS Elastic Beanstalk afin d'utiliser le protocole HTTPS, vous devez configurer un écouteur HTTPS pour l'équilibrer de charge dans votre environnement. Deux types d'équilibrer de charge prennent en charge un écouteur HTTPS : l'équilibrer Classic Load Balancer et l'équilibrer Application Load Balancer.

Vous pouvez utiliser la console Elastic Beanstalk ou un fichier de configuration pour configurer un écouteur sécurisé et attribuer le certificat.

### Note

Les environnements d'instance unique n'ont pas d'équilibrer de charge et ne prennent pas en charge la terminaison HTTPS au niveau de l'équilibrer de charge.

## Configuration d'un écouteur sécurisé à l'aide de la console Elastic Beanstalk

Pour attribuer un certificat à l'équilibrer de charge de votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Load balancer (Équilibrer de charge), choisissez Edit (Modifier).

### Note

Si la catégorie de configuration Load balancer (Équilibrer de charge) ne dispose pas du bouton Edit (Modifier), cela signifie que votre environnement ne dispose pas d'un [équilibrer de charge](#) (p. 520).

5. Sur la page Modifier l'équilibrer de charge, la procédure varie selon le type d'équilibrer de charge associé à votre environnement.
  - Equilibreur de charge classique
    - a. Choisissez Ajouter un écouteur.
    - b. Dans la boîte de dialogue Classic Load Balancer listener (Écouteur Classic Load Balancer), configurez les paramètres suivants :
      - Pour Port d'écoute, tapez le port du trafic entrant, généralement 443.
      - Pour Protocole d'écoute, choisissez HTTPS.
      - Pour Port de l'instance, tapez 80.
      - Pour Protocole de l'instance, choisissez HTTP.
      - Pour Certificat SSL, choisissez votre certificat.
    - c. Choisissez Add (Ajouter).

- Application Load Balancer
  - a. Choisissez Ajouter un écouteur.
  - b. Dans la boîte de dialogue Application Load Balancer listener (Écouteur de l'équilibrer Application Load Balancer), configurez les paramètres suivants :
    - Pour Port, tapez le port du trafic entrant, généralement 443.
    - Pour Protocole, choisissez HTTPS.
    - Pour Certificat SSL, choisissez votre certificat.
  - c. Choisissez Ajouter.

#### Note

Pour l'équilibrer Classic Load Balancer et l'équilibrer Application Load Balancer, si le menu déroulant n'affiche aucun certificat, vous devez créer ou charger un certificat pour votre [nom de domaine personnalisé \(p. 656\)](#) dans [AWS Certificate Manager \(ACM\)](#) (de préférence). Vous pouvez également charger un certificat dans IAM à l'aide de l'AWS CLI.

- Équilibrer de charge du réseau
    - a. Choisissez Ajouter un écouteur.
    - b. Dans la boîte de dialogue Network Load Balancer listener (Écouteur de l'équilibrer Network Load Balancer) pour Port, tapez le port de trafic entrant, généralement 443.
    - c. Choisissez Add (Ajouter).
6. Choisissez Apply.

## Configuration d'un écouteur sécurisé avec un fichier de configuration

Vous pouvez configurer un écouteur sécurisé sur votre équilibrer de charge avec un des [fichiers de configuration \(p. 737\)](#) suivants.

#### Example .ebextensions/securelistener-clb.config

Utilisez cet exemple lorsque votre environnement dispose d'un équilibrer Classic Load Balancer. L'exemple utilise des options dans l'espace de noms `aws:elb:listener` pour configurer un écouteur HTTPS sur le port 443 avec le certificat spécifié et pour transférer le trafic déchiffré vers les instances de votre environnement sur le port 80.

```
option_settings:  
  aws:elb:listener:443:  
    SSLCertificateId: arn:aws:acm:us-east-2:1234567890123:certificate/  
#####  
  ListenerProtocol: HTTPS  
  InstancePort: 80
```

Remplacez le texte en surbrillance par l'ARN de votre certificat. Il peut s'agir d'un certificat que vous avez créé ou chargé dans AWS Certificate Manager (ACM) (de préférence), ou un certificat que vous avez chargé dans IAM avec l'AWS CLI.

Pour de plus amples informations concernant les options de configuration d'un équilibrer Classic Load Balancer, veuillez consulter [Espaces de noms pour la configuration d'un équilibrer de charge Classic Load Balancer \(p. 573\)](#).

#### Example .ebextensions/securelistener-alb.config

Utilisez cet exemple lorsque votre environnement dispose d'un équilibrEUR Application Load Balancer. L'exemple utilise les options dans l'espace de noms `aws:elbv2:listener` pour configurer un écouteur HTTPS sur le port 443 avec le certificat spécifié. L'écouteur achemine le trafic vers le processus par défaut.

```
option_settings:  
  aws:elbv2:listener:443:  
    ListenerEnabled: 'true'  
    Protocol: HTTPS  
    SSLCertificateArns: arn:aws:acm:us-east-2:1234567890123:certificate/  
    #####
```

#### Example .ebextensions/securelistener-nlb.config

Utilisez cet exemple lorsque votre environnement dispose d'un équilibrEUR Network Load Balancer. L'exemple utilise les options dans l'espace de noms `aws:elbv2:listener` pour configurer un écouteur sur le port 443. L'écouteur achemine le trafic vers le processus par défaut.

```
option_settings:  
  aws:elbv2:listener:443:  
    ListenerEnabled: 'true'
```

## Configuration d'un groupe de sécurité

Si vous configurez votre équilibrEUR de charge pour transférer le trafic vers un port de l'instance autre que le port 80, vous devez ajouter une règle à votre groupe de sécurité autorisant le trafic entrant via le port de l'instance à partir de votre équilibrEUR de charge. Si vous créez votre environnement dans un VPC personnalisé, Elastic Beanstalk ajoute automatiquement cette règle.

Pour ajouter cette règle, ajoutez une clé `Resources` à un [fichier de configuration \(p. 737\)](#) dans le répertoire `.ebextensions` correspondant à votre application.

L'exemple de fichier de configuration suivante ajoute une règle de trafic entrant au groupe de sécurité `AWSEBSecurityGroup`. Cela permet le trafic sur le port 1000 à partir du groupe de sécurité de l'équilibrEUR de charge.

#### Example .ebextensions/sg-ingressfromlb.config

```
Resources:  
  sslSecurityGroupIngress:  
    Type: AWS::EC2::SecurityGroupIngress  
    Properties:  
      GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}  
      IpProtocol: tcp  
      ToPort: 1000  
      FromPort: 1000  
      SourceSecurityGroupName: {"Fn::GetAtt": ["AWSEBLoadBalancer",  
      "SourceSecurityGroup.GroupName"]}
```

## Configuration de votre application pour suspendre des connexions HTTPS sur l'instance

Vous pouvez utiliser des [fichiers de configuration \(p. 737\)](#) pour configurer le serveur proxy qui transmet le trafic à votre application afin de mettre des connexions HTTPS hors service. Cette opération est utile si vous souhaitez utiliser HTTPS avec un environnement d'instance unique ou si vous configurez votre équilibrEUR de charge pour transmettre le trafic sans le déchiffrer.

Pour activer HTTPS, vous devez autoriser le trafic entrant sur le port 443 vers l'instance EC2 sur laquelle votre application Elastic Beanstalk est exécutée. Pour ce faire, utilisez la clé `Resources` du fichier de configuration afin d'ajouter une règle pour le port 443 aux règles de trafic entrant pour le groupe de sécurité `AWSEBSecurityGroup`.

L'extrait suivant ajoute une règle de trafic entrant au groupe de sécurité `AWSEBSecurityGroup` qui ouvre le port 443 pour tout le trafic, pour un environnement d'instance unique :

**.ebextensions/https-instance-securitygroup.config**

```
Resources:  
  sslSecurityGroupIngress:  
    Type: AWS::EC2::SecurityGroupIngress  
    Properties:  
      GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}  
      IpProtocol: tcp  
      ToPort: 443  
      FromPort: 443  
      CidrIp: 0.0.0.0/0
```

Dans un environnement à charge équilibrée Amazon Virtual Private Cloud (Amazon VPC) par défaut, vous pouvez modifier cette stratégie pour accepter uniquement le trafic provenant de l'équilibrEUR de charge. Veuillez consulter [Configuration du chiffrement de bout en bout dans un environnement Elastic Beanstalk à charge équilibrée \(p. 824\)](#) pour obtenir un exemple.

Plates-formes

- [Résiliation de connexions HTTPS sur des instances EC2 exécutant Docker \(p. 800\)](#)
- [Suspension des connexions HTTPS sur des instances EC2 exécutant Go \(p. 802\)](#)
- [Résiliation des connexions HTTPS sur des instances EC2 exécutant Java SE \(p. 804\)](#)
- [Suspension des connexions HTTPS sur des instances EC2 exécutant Node.js \(p. 807\)](#)
- [Résiliation des connexions HTTPS sur des instances EC2 exécutant PHP \(p. 809\)](#)
- [Suspension des connexions HTTPS sur des instances EC2 exécutant Python \(p. 811\)](#)
- [Suspension des connexions HTTPS sur des instances EC2 exécutant Ruby \(p. 814\)](#)
- [Suspension des connexions HTTPS sur des instances EC2 exécutant Tomcat \(p. 818\)](#)
- [Résiliation des connexions HTTPS sur les instances Amazon EC2 exécutant .NET Core sous Linux \(p. 820\)](#)
- [Résiliation des connexions HTTPS sur les instances Amazon EC2 exécutant .NET \(p. 822\)](#)

## Résiliation de connexions HTTPS sur des instances EC2 exécutant Docker

Pour les conteneurs Docker, vous utilisez un [fichier de configuration \(p. 737\)](#) pour activer HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le document certificat et clé privée comme demandé, puis enregistrez-le dans le répertoire `.ebextensions` de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :  
`/etc/nginx/conf.d/https.conf`

Configure le serveur nginx. Ce fichier est chargé lorsque le service nginx démarre.

`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. Remplacez le `contenu du fichier de certificat` par le contenu de votre certificat.

### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
-----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. Remplacez le `contenu de clé privée` par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

Example `.ebextensions/https-instance.config`

```
files:
  /etc/nginx/conf.d/https.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      # HTTPS Server

      server {
        listen 443;
        server_name localhost;

        ssl on;
        ssl_certificate /etc/pki/tls/certs/server.crt;
        ssl_certificate_key /etc/pki/tls/certs/server.key;

        ssl_session_timeout 5m;

        ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
        ssl_prefer_server_ciphers on;

        location / {
          proxy_pass http://docker;
          proxy_http_version 1.1;

          proxy_set_header Connection "";
          proxy_set_header Host $host;
          proxy_set_header X-Real-IP $remote_addr;
          proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
          proxy_set_header X-Forwarded-Proto https;
        }
      }

  /etc/pki/tls/certs/server.crt:
```

```
mode: "000400"
owner: root
group: root
content: |
    -----BEGIN CERTIFICATE-----
    certificate file contents
    -----END CERTIFICATE-----

/etc/pki/tls/certs/server.key:
mode: "000400"
owner: root
group: root
content: |
    -----BEGIN RSA PRIVATE KEY-----
    private key contents # See note below.
    -----END RSA PRIVATE KEY-----
```

#### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

#### Example .ebextensions/https-instance-single.config

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrEUR de charge de façon à [transférer le trafic sécurisé sans le toucher \(p. 827\)](#) ou à [déchiffrer et rechiffrer \(p. 824\)](#) le trafic pour un chiffrement de bout en bout.

## Suspension des connexions HTTPS sur des instances EC2 exécutant Go

Pour les types de conteneurs Go, vous activez HTTPS avec un [fichier de configuration \(p. 737\)](#) et un fichier de configuration nginx qui configure le serveur nginx pour utiliser HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant les espaces réservés pour le certificat et la clé privée comme demandé, puis enregistrez-le dans le répertoire .ebextensions de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé Resources active le port 443 sur le groupe de sécurité utilisé par l'instance de votre environnement.
- La clé files crée les fichiers suivants sur l'instance :

/etc/pki/tls/certs/server.crt

Crée le fichier de certificat sur l'instance. Remplacez le *contenu du fichier de certificat* par le contenu de votre certificat.

#### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
-----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

/etc/pki/tls/certs/server.key

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

- La clé `container_commands` redémarre le serveur nginx une fois que tout est configuré de telle sorte que le serveur charge le fichier de configuration nginx.

#### Example .ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01restart_nginx:
    command: "service nginx restart"
```

#### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Placez les éléments suivants dans un fichier avec l'extension .conf dans le répertoire .ebextensions/nginx/conf.d/ de votre bundle de fichiers source (par exemple, .ebextensions/nginx/conf.d/https.conf). Remplacez **app\_port** par le numéro de port sur lequel votre application écoute. Cet exemple configure le serveur nginx pour écouter sur le port 443 à l'aide de SSL. Pour plus d'informations sur ces fichiers de configuration sur la plateforme Go, consultez [Configuration du proxy inverse \(p. 105\)](#).

Example .ebextensions/nginx/conf.d/https.conf

```
# HTTPS server

server {
    listen      443;
    server_name localhost;

    ssl          on;
    ssl_certificate /etc/pki/tls/certs/server.crt;
    ssl_certificate_key /etc/pki/tls/certs/server.key;

    ssl_session_timeout 5m;

    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers on;

    location / {
        proxy_pass http://localhost:app_port;
        proxy_set_header Connection "";
        proxy_http_version 1.1;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
    }
}
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

Example .ebextensions/https-instance-single.config

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrEUR de charge de façon à [transférer le trafic sécurisé sans le toucher \(p. 827\)](#) ou à [déchiffrer et rechiffrer \(p. 824\)](#) le trafic pour un chiffrement de bout en bout.

## Résiliation des connexions HTTPS sur des instances EC2 exécutant Java SE

Pour les types de conteneurs Java SE, vous activez HTTPS avec un [fichier de configuration \(p. 737\)](#) .ebextensions, ainsi qu'un fichier de configuration nginx qui configure le serveur nginx pour utiliser HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant les espaces réservés pour le certificat et la clé privée comme indiqué, puis enregistrez-le dans le répertoire `.ebextensions`. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :

```
/etc/pki/tls/certs/server.crt
```

Crée le fichier de certificat sur l'instance. Remplacez le *contenu du fichier de certificat* par le contenu de votre certificat.

#### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
-----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

```
/etc/pki/tls/certs/server.key
```

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

- La clé `container_commands` redémarre le serveur nginx une fois que tout est configuré de telle sorte que le serveur charge le fichier de configuration nginx.

#### Example `.ebextensions/https-instance.config`

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01restart_nginx:
    command: "service nginx restart"
```

#### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez

votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Placez les éléments suivants dans un fichier avec l'extension .conf dans le répertoire .ebextensions/nginx/conf.d/ de votre bundle de fichiers source (par exemple, .ebextensions/nginx/conf.d/https.conf). Remplacez `app_port` par le numéro de port sur lequel votre application écoute. Cet exemple configure le serveur nginx pour écouter sur le port 443 à l'aide de SSL. Pour plus d'informations sur ces fichiers de configuration sur la plateforme Java SE, consultez [Configuration du proxy inverse \(p. 134\)](#).

Example .ebextensions/nginx/conf.d/https.conf

```
# HTTPS server

server {
    listen      443;
    server_name localhost;

    ssl          on;
    ssl_certificate /etc/pki/tls/certs/server.crt;
    ssl_certificate_key /etc/pki/tls/certs/server.key;

    ssl_session_timeout 5m;

    ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
    ssl_prefer_server_ciphers on;

    location / {
        proxy_pass http://localhost:app_port;
        proxy_set_header Connection "";
        proxy_http_version 1.1;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto https;
    }
}
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

Example .ebextensions/https-instance-single.config

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrEUR de charge de façon à transférer le trafic sécurisé sans le toucher (p. 827) ou à déchiffrer et rechiffrer (p. 824) le trafic pour un chiffrement de bout en bout.

## Suspension des connexions HTTPS sur des instances EC2 exécutant Node.js

L'exemple de fichier de configuration suivant étend la configuration nginx par défaut (p. 262) pour écouter sur le port 443 et mettre fin aux connexions SSL/TLS avec un certificat public et une clé privée.

Si vous avez configuré votre environnement pour les rapports améliorés sur l'état (p. 837), vous devez configurer nginx pour générer les journaux d'accès. Pour ce faire, supprimez la mise en commentaire du bloc de lignes sous le commentaire où il est écrit `# For enhanced health...` en supprimant les caractères `#` de tête.

Example .ebextensions/https-instance.config

```
files:
  /etc/nginx/conf.d/https.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      # HTTPS server

      server {
        listen          443;
        server_name    localhost;

        ssl              on;
        ssl_certificate /etc/pki/tls/certs/server.crt;
        ssl_certificate_key /etc/pki/tls/certs/server.key;

        ssl_session_timeout 5m;

        ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
        ssl_prefer_server_ciphers on;

        # For enhanced health reporting support, uncomment this block:

        #if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
        #  set $year $1;
        #  set $month $2;
        #  set $day $3;
        #  set $hour $4;
        #}
        #access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour
healthd;
        #access_log /var/log/nginx/access.log main;

        location / {
          proxy_pass  http://nodejs;
          proxy_set_header Connection "";
          proxy_http_version 1.1;
          proxy_set_header Host           $host;
          proxy_set_header X-Real-IP     $remote_addr;
          proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
          proxy_set_header X-Forwarded-Proto https;
        }
      }

  /etc/pki/tls/certs/server.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN CERTIFICATE-----
```

```
certificate file contents
-----END CERTIFICATE-----

/etc/pki/tls/certs/server.key:
mode: "000400"
owner: root
group: root
content: |
-----BEGIN RSA PRIVATE KEY-----
private key contents # See note below.
-----END RSA PRIVATE KEY-----
```

La clé files crée les fichiers suivants sur l'instance :

/etc/nginx/conf.d/https.conf

Configure le serveur nginx. Ce fichier est chargé lorsque le service nginx démarre.

/etc/pki/tls/certs/server.crt

Crée le fichier de certificat sur l'instance. Remplacez le *contenu du fichier de certificat* par le contenu de votre certificat.

#### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans server.crt après votre certificat de site.

```
-----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

/etc/pki/tls/certs/server.key

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

#### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

Example .ebextensions/https-instance-single.config

Resources:

```
sslSecurityGroupIngress:  
  Type: AWS::EC2::SecurityGroupIngress  
  Properties:  
    GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}  
    IpProtocol: tcp  
    ToPort: 443  
    FromPort: 443  
    CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrer de charge de façon à transférer le trafic sécurisé sans le toucher ([p. 827](#)) ou à déchiffrer et rechiffrer ([p. 824](#)) le trafic pour un chiffrement de bout en bout.

## Résiliation des connexions HTTPS sur des instances EC2 exécutant PHP

Pour les types de conteneurs PHP, vous utilisez un [fichier de configuration \(p. 737\)](#) pour permettre au serveur HTTP Apache d'utiliser HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le document certificat et clé privée comme demandé, puis enregistrez-le dans le répertoire `.ebextensions` de votre bundle de fichiers source.

Le fichier de configuration effectue les tâches suivantes :

- La clé `packages` utilise yum pour installer `mod24_ssl`.
- La clé `files` crée les fichiers suivants sur l'instance :  
`/etc/httpd/conf.d/ssl.conf`

Configure le serveur Apache. Ce fichier se charge lorsque le service Apache démarre.

`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. Remplacez le *contenu du fichier de certificat* par le contenu de votre certificat.

### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
-----BEGIN CERTIFICATE-----  
certificate file contents  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----  
first intermediate certificate  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----  
second intermediate certificate  
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

### Example .ebextensions/https-instance.config

```
packages:
  yum:
    mod24_ssl : []

files:
  /etc/httpd/conf.d/ssl.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      LoadModule ssl_module modules/mod_ssl.so
      Listen 443
      <VirtualHost *:443>
        <Proxy *>
          Order deny,allow
          Allow from all
        </Proxy>

        SSLEngine          on
        SSLCertificateFile "/etc/pki/tls/certs/server.crt"
        SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"
        SSLCipherSuite     EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH
        SSLProtocol        All -SSLv2 -SSLv3
        SSLHonorCipherOrder On
        SSLSessionTickets Off

        Header always set Strict-Transport-Security "max-age=63072000; includeSubdomains;
preload"
        Header always set X-Frame-Options DENY
        Header always set X-Content-Type-Options nosniff

        ProxyPass / http://localhost:80/ retry=0
        ProxyPassReverse / http://localhost:80/
        ProxyPreserveHost on
        RequestHeader set X-Forwarded-Proto "https" early

      </VirtualHost>

  /etc/pki/tls/certs/server.crt:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    mode: "000400"
    owner: root
    group: root
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----
```

#### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le

déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

Example .ebextensions/https-instance-single.config

```
Resources:  
  sslSecurityGroupIngress:  
    Type: AWS::EC2::SecurityGroupIngress  
    Properties:  
      GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}  
      IpProtocol: tcp  
      ToPort: 443  
      FromPort: 443  
      CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrer de charge de façon à [transférer le trafic sécurisé sans le toucher \(p. 827\)](#) ou à [déchiffrer et rechiffrer \(p. 824\)](#) le trafic pour un chiffrement de bout en bout.

## Suspension des connexions HTTPS sur des instances EC2 exécutant Python

Pour les types de conteneurs Python utilisant Apache HTTP Server avec l'interface WSGI (Web Server Gateway Interface), vous utilisez un [fichier de configuration \(p. 737\)](#) pour permettre l'Apache HTTP Server d'utiliser HTTPS.

Ajoutez l'extrait suivant à votre [fichier de configuration \(p. 737\)](#), en remplaçant le document certificat et clé privée comme demandé, puis enregistrez-le dans le répertoire .ebextensions de votre bundle source. Le fichier de configuration effectue les tâches suivantes :

- La clé packages utilise yum pour installer mod24\_ssl.
- La clé files crée les fichiers suivants sur l'instance :  
`/etc/httpd/conf.d/ssl.conf`

Configure le serveur Apache. Si votre application n'est pas nommée `application.py`, remplacez le texte en surbrillance dans la valeur pour `WSGIScriptAlias` par le chemin d'accès local à votre application. Par exemple, une application Django peut être sur `django/wsgi.py`. L'emplacement doit correspondre à la valeur de l'option `WSGIPath` que vous avez définie pour votre environnement.

En fonction des besoins de votre application, vous aurez peut-être également besoin d'ajouter d'autres répertoires au paramètre `python-path`.

`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. Remplacez le [contenu du fichier de certificat](#) par le contenu de votre certificat.

### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
-----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

/etc/pki/tls/certs/server.key

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

- La clé `container_commands` arrête le service `httpd` une fois que tout a été configuré de telle sorte que le service utilise le nouveau certificat et le nouveau fichier `https.conf`.

#### Note

L'exemple fonctionne uniquement dans des environnements utilisant la plateforme [Python \(p. 359\)](#).

#### Example .ebextensions/https-instance.config

```
packages:
  yum:
    mod24_ssl : []

files:
  /etc/httpd/conf.d/ssl.conf:
    mode: "000644"
    owner: root
    group: root
    content: |
      LoadModule wsgi_module modules/mod_wsgi.so
      WSGIPythonHome /opt/python/run/baselinenv
      WSGISocketPrefix run/wsgi
      WSGIRestrictEmbedded On
      Listen 443
      <VirtualHost *:443>
        SSLEngine on
        SSLCertificateFile "/etc/pki/tls/certs/server.crt"
        SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"

        Alias /static/ /opt/python/current/app/static/
        <Directory /opt/python/current/app/static>
          Order allow,deny
          Allow from all
        </Directory>

        WSGIScriptAlias / /opt/python/current/app/application.py

        <Directory /opt/python/current/app>
          Require all granted
        </Directory>

        WSGIDaemonProcess wsgi-ssl processes=1 threads=15 display-name=%{GROUP} \
          python-path=/opt/python/current/app \
          python-home=/opt/python/run/venv \
          home=/opt/python/current/app \
```

```
        user=wsgi \
        group=wsgi
WSGIProcessGroup wsgi-ssl

</VirtualHost>

/etc/pki/tls/certs/server.crt:
mode: "000400"
owner: root
group: root
content: |
-----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----

/etc/pki/tls/certs/server.key:
mode: "000400"
owner: root
group: root
content: |
-----BEGIN RSA PRIVATE KEY-----
private key contents # See note below.
-----END RSA PRIVATE KEY-----


container_commands:
01killhttpd:
  command: "killall httpd"
02waitforhttpdeath:
  command: "sleep 3"
```

### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

### Example .ebextensions/https-instance-single.config

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrEUR de charge de façon à transférer le trafic sécurisé sans le toucher (p. 827) ou à déchiffrer et rechiffrer (p. 824) le trafic pour un chiffrement de bout en bout.

## Suspension des connexions HTTPS sur des instances EC2 exécutant Ruby

Pour les types de conteneur Ruby, la façon dont vous activez HTTPS dépend du type de serveur d'applications utilisé.

### Rubriques

- [Configuration de HTTPS pour Ruby avec Puma \(p. 814\)](#)
- [Configuration de HTTPS pour Ruby avec Passenger \(p. 816\)](#)

### Configuration de HTTPS pour Ruby avec Puma

Pour les types de conteneurs Ruby qui utilisent Puma en tant que serveur d'applications, vous utilisez un [fichier de configuration \(p. 737\)](#) pour activer HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le document certificat et clé privée comme demandé, puis enregistrez-le dans le répertoire `.ebextensions` de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :

`/etc/nginx/conf.d/https.conf`

Configure le serveur nginx. Ce fichier est chargé lorsque le service nginx démarre.

`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. Remplacez le *contenu du fichier de certificat* par le contenu de votre certificat.

#### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
-----BEGIN CERTIFICATE-----
certificate file contents
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

- La clé `container_commands` redémarre le serveur nginx une fois que tout est configuré de telle sorte que le serveur utilise le nouveau fichier `https.conf`.

### Example .ebextensions/https-instance.config

```
files:
  /etc/nginx/conf.d/https.conf:
    content: |
      # HTTPS server

      server {
        listen          443;
        server_name    localhost;

        ssl           on;
        ssl_certificate  /etc/pki/tls/certs/server.crt;
        ssl_certificate_key /etc/pki/tls/certs/server.key;

        ssl_session_timeout 5m;

        ssl_protocols  TLSv1 TLSv1.1 TLSv1.2;
        ssl_prefer_server_ciphers on;

        location / {
          proxy_pass  http://my_app;
          proxy_set_header Host $host;
          proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
          proxy_set_header X-Forwarded-Proto https;
        }

        location /assets {
          alias /var/app/current/public/assets;
          gzip_static on;
          gzip on;
          expires max;
          add_header Cache-Control public;
        }

        location /public {
          alias /var/app/current/public;
          gzip_static on;
          gzip on;
          expires max;
          add_header Cache-Control public;
        }
      }

  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----


container_commands:
  01restart_nginx:
    command: "service nginx restart"
```

#### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez

votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

Example .ebextensions/https-instance-single.config

```
Resources:  
  sslSecurityGroupIngress:  
    Type: AWS::EC2::SecurityGroupIngress  
    Properties:  
      GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}  
      IpProtocol: tcp  
      ToPort: 443  
      FromPort: 443  
      CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrEUR de charge de façon à [transférer le trafic sécurisé sans le toucher \(p. 827\)](#) ou à [déchiffrer et rechiffrer \(p. 824\)](#) le trafic pour un chiffrement de bout en bout.

## Configuration de HTTPS pour Ruby avec Passenger

Pour les types de conteneurs Ruby qui utilisent Passenger en tant que serveur d'applications, vous utilisez un fichier de configuration et un fichier JSON pour activer HTTPS.

Pour configurer HTTPS pour Ruby avec Passenger

1. Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le document certificat et clé privée comme demandé, puis enregistrez-le dans le répertoire .ebextensions de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé files crée les fichiers suivants sur l'instance :  
`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. Remplacez le *contenu du fichier de certificat* par le contenu de votre certificat.

### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans `server.crt` après votre certificat de site.

```
-----BEGIN CERTIFICATE-----  
certificate file contents  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----  
first intermediate certificate  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----  
second intermediate certificate
```

-----END CERTIFICATE-----

/etc/pki/tls/certs/server.key

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

#### Example Extrait .Ebextensions pour la configuration de HTTPS pour Ruby avec Passenger

```
files:  
  /etc/pki/tls/certs/server.crt:  
    content: |  
      -----BEGIN CERTIFICATE-----  
      certificate file contents  
      -----END CERTIFICATE-----  
  
  /etc/pki/tls/certs/server.key:  
    content: |  
      -----BEGIN RSA PRIVATE KEY-----  
      private key contents # See note below.  
      -----END RSA PRIVATE KEY-----
```

#### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

- Créez un fichier texte et ajoutez le JSON suivant au fichier. Enregistrez-le dans le répertoire racine de votre groupe source avec le nom `passenger-standalone.json`. Ce fichier JSON configure Passenger pour utiliser HTTPS.

#### Important

Ce fichier JSON ne doit pas contenir une marque d'ordre d'octet (BOM). Si tel est le cas, la bibliothèque Passenger JSON ne lira pas le fichier correctement et le service Passenger ne déarrera pas.

#### Example passenger-standalone.json

```
{  
  "ssl" : true,  
  "ssl_port" : 443,  
  "ssl_certificate" : "/etc/pki/tls/certs/server.crt",  
  "ssl_certificate_key" : "/etc/pki/tls/certs/server.key"  
}
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

#### Example .ebextensions/https-instance-single.config

```
Resources:  
  sslSecurityGroupIngress:  
    Type: AWS::EC2::SecurityGroupIngress
```

```
Properties:  
  GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}  
  IpProtocol: tcp  
  ToPort: 443  
  FromPort: 443  
  CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrEUR de charge de façon à transférer le trafic sécurisé sans le toucher (p. 827) ou à déchiffrer et rechiffrer (p. 824) le trafic pour un chiffrement de bout en bout.

## Suspension des connexions HTTPS sur des instances EC2 exécutant Tomcat

Pour les types de conteneurs Tomcat, vous devez utiliser un [fichier de configuration \(p. 737\)](#) afin d'autoriser Apache HTTP Server à utiliser HTTPS lorsqu'il agit en tant que proxy inverse pour Tomcat.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant le document certificat et clé privée comme demandé, puis enregistrez-le dans le répertoire `.ebextensions` de votre bundle de fichiers source. Le fichier de configuration effectue les tâches suivantes :

- La clé `files` crée les fichiers suivants sur l'instance :  
`/etc/pki/tls/certs/server.crt`

Crée le fichier de certificat sur l'instance. Remplacez le *contenu du fichier de certificat* par le contenu de votre certificat.

### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

`/etc/pki/tls/certs/server.key`

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

`/opt/elasticbeanstalk/hooks/appdeploy/post/99_start_httpd.sh`

Crée un script de hook post-déploiement pour redémarrer le service httpd.

### Example `.ebextensions/https-instance.config`

```
files:  
  /etc/pki/tls/certs/server.crt:  
    mode: "000400"  
    owner: root  
    group: root  
    content: |  
      -----BEGIN CERTIFICATE-----  
      certificate file contents  
      -----END CERTIFICATE-----  
  
  /etc/pki/tls/certs/server.key:  
    mode: "000400"  
    owner: root  
    group: root  
    content: |  
      -----BEGIN RSA PRIVATE KEY-----
```

```
private key contents # See note below.  
-----END RSA PRIVATE KEY-----  
  
/opt/elasticbeanstalk/hooks/appdeploy/post/99_start_httpd.sh:  
mode: "000755"  
owner: root  
group: root  
content: |  
#!/usr/bin/env bash  
sudo service httpd restart
```

Vous devez également configurer le serveur proxy de votre environnement pour écouter sur le port 443. La configuration Apache 2.4 suivante ajoute un écouteur sur le port 443. Pour en savoir plus, veuillez consulter [Configuration du serveur proxy de votre environnement Tomcat \(p. 126\)](#).

Example .ebextensions/httpd/conf.d/ssl.conf

```
Listen 443  
<VirtualHost *:443>  
    ServerName server-name  
    SSLEngine on  
    SSLCertificateFile "/etc/pki/tls/certs/server.crt"  
    SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"  
  
    <Proxy *>  
        Require all granted  
    </Proxy>  
    ProxyPass / http://localhost:8080/ retry=0  
    ProxyPassReverse / http://localhost:8080/  
    ProxyPreserveHost on  
  
    ErrorLog /var/log/httpd/elasticbeanstalk-ssl-error_log  
  
</VirtualHost>
```

Votre fournisseur de certificats peut inclure des certificats intermédiaires, que vous pouvez installer pour améliorer la compatibilité avec les clients mobiles. Configurez Apache avec un bundle de certificats intermédiaires en ajoutant ce qui suit à votre fichier de configuration SSL (consultez [Extension et remplacement de la configuration Apache par défaut \(p. 127\)](#) pour l'emplacement) :

- Dans le contenu du fichier `ssl.conf`, spécifiez le fichier de chaîne :

```
SSLCertificateKeyFile "/etc/pki/tls/certs/server.key"  
SSLCertificateChainFile "/etc/pki/tls/certs/gd_bundle.crt"  
SSLCipherSuite ECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH
```

- Ajoutez une nouvelle entrée à la clé `files` avec le contenu des certificats intermédiaires :

```
files:  
  /etc/pki/tls/certs/gd_bundle.crt:  
    mode: "000400"  
    owner: root  
    group: root  
    content: |  
      -----BEGIN CERTIFICATE-----  
      First intermediate certificate  
      -----END CERTIFICATE-----  
      -----BEGIN CERTIFICATE-----  
      Second intermediate certificate  
      -----END CERTIFICATE-----
```

### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

Example .ebextensions/https-instance-single.config

```
Resources:  
  sslSecurityGroupIngress:  
    Type: AWS::EC2::SecurityGroupIngress  
    Properties:  
      GroupId: {"Fn::GetAtt": ["AWSEBSecurityGroup", "GroupId"]}  
      IpProtocol: tcp  
      ToPort: 443  
      FromPort: 443  
      CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrEUR de charge de façon à transférer le trafic sécurisé sans le toucher (p. 827) ou à déchiffrer et rechiffrer (p. 824) le trafic pour un chiffrement de bout en bout.

## Résiliation des connexions HTTPS sur les instances Amazon EC2 exécutant .NET Core sous Linux

Pour les types de conteneur .NET Core sous Linux, vous activez HTTPS avec un [fichier de configuration \(p. 737\)](#) .ebextensions et un fichier de configuration nginx qui configure le serveur nginx pour utiliser HTTPS.

Ajoutez l'extrait suivant à votre fichier de configuration, en remplaçant les espaces réservés pour le certificat et la clé privée comme indiqué, puis enregistrez-le dans le répertoire .ebextensions. Le fichier de configuration effectue les tâches suivantes :

- La clé files crée les fichiers suivants sur l'instance :

/etc/pki/tls/certs/server.crt

Crée le fichier de certificat sur l'instance. Remplacez le *contenu du fichier de certificat* par le contenu de votre certificat.

### Note

YAML utilise une mise en retrait cohérente. Respectez le niveau de retrait lorsque vous remplacez du contenu dans un exemple de fichier de configuration et veillez à ce que votre éditeur de texte utilise des espaces, et non des caractères de tabulation, pour la mise en retrait.

Si vous avez des certificats intermédiaires, incluez-les dans server.crt après votre certificat de site.

```
-----BEGIN CERTIFICATE-----  
certificate file contents  
-----END CERTIFICATE-----  
-----BEGIN CERTIFICATE-----
```

```
first intermediate certificate
-----END CERTIFICATE-----
-----BEGIN CERTIFICATE-----
second intermediate certificate
-----END CERTIFICATE-----
```

/etc/pki/tls/certs/server.key

Crée le fichier de clé privée sur l'instance. Remplacez le *contenu de clé privée* par le contenu de la clé privée utilisée pour créer la demande de certificat ou le certificat auto-signé.

- La clé `container_commands` redémarre le serveur nginx une fois que tout est configuré de telle sorte que le serveur charge le fichier de configuration nginx.

#### Example .ebextensions/https-instance.config

```
files:
  /etc/pki/tls/certs/server.crt:
    content: |
      -----BEGIN CERTIFICATE-----
      certificate file contents
      -----END CERTIFICATE-----

  /etc/pki/tls/certs/server.key:
    content: |
      -----BEGIN RSA PRIVATE KEY-----
      private key contents # See note below.
      -----END RSA PRIVATE KEY-----

container_commands:
  01restart_nginx:
    command: "systemctl restart nginx"
```

#### Note

Évitez de valider un fichier de configuration qui contient votre clé privée de contrôle de code source. Une fois que vous avez testé la configuration et confirmé qu'elle fonctionne, stockez votre clé privée dans Amazon S3 et modifiez la configuration pour la télécharger durant le déploiement. Pour obtenir des instructions, consultez [Stockage sécurisé des clés privées dans Amazon S3 \(p. 828\)](#).

Placez les éléments suivants dans un fichier avec l'extension `.conf` dans le répertoire `.ebextensions/nginx/conf.d/` de votre bundle de fichiers source (par exemple, `.ebextensions/nginx/conf.d/https.conf`). Remplacez `app_port` par le numéro de port sur lequel votre application écoute. Cet exemple configure le serveur nginx pour écouter sur le port 443 à l'aide de SSL. Pour de plus amples informations sur les fichiers de configuration sur la plateforme .NET Core sous Linux, veuillez consulter [the section called “Serveur proxy” \(p. 165\)](#).

#### Example .ebextensions/nginx/conf.d/https.conf

```
# HTTPS server

server {
  listen        443;
  server_name  localhost;

  ssl           on;
  ssl_certificate /etc/pki/tls/certs/server.crt;
  ssl_certificate_key /etc/pki/tls/certs/server.key;

  ssl_session_timeout 5m;
```

```
ssl_protocols TLSv1 TLSv1.1 TLSv1.2;
ssl_prefer_server_ciphers on;

location / {
    proxy_pass http://localhost:app\_port;
    proxy_set_header Connection "";
    proxy_http_version 1.1;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto https;
}
}
```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

Example .ebextensions/https-instance-single.config

```
Resources:
  sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : [ "AWSEBSecurityGroup", "GroupId" ]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrEUR de charge de façon à [transférer le trafic sécurisé sans le toucher \(p. 827\)](#) ou à [déchiffrer et rechiffrer \(p. 824\)](#) le trafic pour un chiffrement de bout en bout.

## Résiliation des connexions HTTPS sur les instances Amazon EC2 exécutant .NET

Le [fichier de configuration \(p. 737\)](#) permet de créer et d'exécuter un script Windows PowerShell qui effectue les tâches suivantes :

- Il vérifie si un certificat HTTPS existant est lié au port 443.
- Il récupère le [certificat PFX \(p. 794\)](#) et le mot de passe auprès d'un compartiment Amazon S3.

### Note

Ajoutez une stratégie `AmazonS3ReadOnlyAccess` à `aws-elasticbeanstalk-ec2-role` pour accéder aux fichiers de certificat SSL et de mot de passe du compartiment Amazon S3.

- Il installe le certificat.
- Il lie le certificat au port 443.

### Note

Pour supprimer le point de terminaison HTTP (port 80), incluez la commande `Remove-WebBinding` sous la section Remove the HTTP binding de l'exemple.

Example .ebextensions/https-instance-dotnet.config

```
files:
```

```

"C:\certs\install-cert.ps1":
content: |
    import-module webadministration
    ## Settings - replace the following values with your own
    $bucket = "DOC-EXAMPLE-BUCKET"          ## S3 bucket name
    $certkey = "example.com.pfx" ## S3 object key for your PFX certificate
    $pwdkey = "password.txt"      ## S3 object key for a text file containing the
    certificate's password
    ##

    # Set variables
    $certfile = "C:\cert.pfx"
    $pwdfile = "C:\certs\pwdcontent"
    Read-S3Object -BucketName $bucket -Key $pwdkey -File $pwdfile
    $pwd = Get-Content $pwdfile -Raw

    # Clean up existing binding
    if ( Get-WebBinding "Default Web Site" -Port 443 ) {
        Echo "Removing WebBinding"
        Remove-WebBinding -Name "Default Web Site" -BindingInformation *:443:
    }
    if ( Get-Item -path IIS:\SslBindings\0.0.0.0!443 ) {
        Echo "Deregistering WebBinding from IIS"
        Remove-Item -path IIS:\SslBindings\0.0.0.0!443
    }

    # Download certificate from S3
    Read-S3Object -BucketName $bucket -Key $certkey -File $certfile

    # Install certificate
    Echo "Installing cert..."
    $securepwd = ConvertTo-SecureString -String $pwd -Force -AsPlainText
    $cert = Import-PfxCertificate -FilePath $certfile cert:\localMachine\my -Password
$securepwd

    # Create site binding
    Echo "Creating and registering WebBinding"
    New-WebBinding -Name "Default Web Site" -IP "*" -Port 443 -Protocol https
    New-Item -path IIS:\SslBindings\0.0.0.0!443 -value $cert -Force

    ## Remove the HTTP binding
    ## (optional) Uncomment the following line to unbind port 80
    # Remove-WebBinding -Name "Default Web Site" -BindingInformation *:80:
    ##

    # Update firewall
    netsh advfirewall firewall add rule name="Open port 443" protocol=TCP localport=443
action=allow dir=OUT

commands:
00_install_ssl:
    command: powershell -NoProfile -ExecutionPolicy Bypass -file C:\\certs\\install-
cert.ps1

```

Dans un environnement à instance unique, vous devez également modifier le groupe de sécurité de l'instance pour autoriser le trafic sur le port 443. Le fichier de configuration suivant récupère l'ID du groupe de sécurité à l'aide d'une [fonction \(p. 764\)](#) AWS CloudFormation et lui ajoute une règle.

Example .ebextensions/https-instance-single.config

```

Resources:
sslSecurityGroupIngress:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:

```

```
GroupId: {"Fn::GetAtt" : [ "AWSEBSecurityGroup", "GroupId" ]}  
IpProtocol: tcp  
ToPort: 443  
FromPort: 443  
CidrIp: 0.0.0.0/0
```

Pour un environnement équilibré en charge, vous configurez l'équilibrer de charge de façon à [transférer le trafic sécurisé sans le toucher \(p. 827\)](#) ou à [déchiffrer et rechiffrer \(p. 824\)](#) le trafic pour un chiffrement de bout en bout.

## Configuration du chiffrement de bout en bout dans un environnement Elastic Beanstalk à charge équilibrée

La suspension des connexions sécurisées au niveau de l'équilibrer de charge et l'utilisation de HTTP sur le backend peuvent s'avérer suffisantes pour votre application. Le trafic réseau entre les ressources AWS ne peut pas être écouté par les instances qui ne font pas partie de la connexion, même si elles sont exécutées sous le même compte.

Toutefois, si vous développez une application qui doit respecter des réglementations externes strictes, vous pouvez être contraint de sécuriser toutes les connexions réseau. Pour respecter ces obligations, vous pouvez utiliser la console Elastic Beanstalk ou les [fichiers de configuration \(p. 737\)](#) afin que l'équilibrer de charge de votre environnement Elastic Beanstalk se connecte aux instances backend en toute sécurité. La procédure suivante se concentre sur les fichiers de configuration.

Commencez par [ajouter un écouteur sécurisé à votre équilibrer de charge \(p. 797\)](#), si vous ne l'avez pas encore fait.

Vous devez également configurer les instances de votre environnement pour autoriser l'écoute sur le port sécurisé et la suspension des connexions HTTPS. La configuration varie selon la plateforme. Pour obtenir les instructions, veuillez consulter [Configuration de votre application pour suspendre des connexions HTTPS sur l'instance \(p. 799\)](#). Vous pouvez utiliser un [certificat auto-signé \(p. 794\)](#) pour les instances EC2 sans difficulté.

Ensuite, configurez l'écouteur pour transférer le trafic à l'aide du protocole HTTPS vers le port sécurisé utilisé par votre application. Utilisez l'un des fichiers de configuration suivants en fonction du type d'équilibrer de charge utilisé par votre environnement.

### **.ebextensions/https-reencrypt-clb.config**

Utilisez ce fichier de configuration avec un équilibrer Classic Load Balancer. En plus de configurer l'équilibrer de charge, le fichier de configuration modifie également par défaut la vérification de l'état pour utiliser le port 443 et HTTPS, afin de veiller à ce que l'équilibrer de charge puisse se connecter en toute sécurité.

```
option_settings:  
aws:elb:listener:443:  
  InstancePort: 443  
  InstanceProtocol: HTTPS  
aws:elasticbeanstalk:application:  
  Application Healthcheck URL: HTTPS:443 /
```

### **.ebextensions/https-reencrypt-alb.config**

Utilisez ce fichier de configuration avec un équilibrer Application Load Balancer.

```
option_settings:
```

```
aws:elbv2:listener:443:  
  DefaultProcess: https  
  ListenerEnabled: 'true'  
  Protocol: HTTPS  
aws:elasticbeanstalk:environment:process:https:  
  Port: '443'  
  Protocol: HTTPS
```

#### .ebextensions/https-reencrypt-nlb.config

Utilisez ce fichier de configuration avec un équilibrEUR Network Load Balancer.

```
option_settings:  
  aws:elbv2:listener:443:  
    DefaultProcess: https  
    ListenerEnabled: 'true'  
  aws:elasticbeanstalk:environment:process:https:  
    Port: '443'
```

L'option `DefaultProcess` est nommée ainsi en raison des équilibrEURS de charge Application Load Balancer, qui peuvent avoir des écouteurs autres que ceux par défaut sur le même port pour le trafic vers des chemins spécifiques (veuillez consulter [Application Load Balancer \(p. 574\)](#) pour plus de détails). Pour un équilibrEUR de charge Network Load Balancer, l'option spécifie le seul processus cible de cet écouteur.

Dans cet exemple, nous avons nommé le processus `https`, car il écoute le trafic sécurisé (HTTPS). L'écouteur envoie le trafic vers le processus sur le port désigné à l'aide du protocole TCP, car un équilibrEUR de charge Network Load Balancer fonctionne uniquement avec TCP. C'est tout à fait normal, car le trafic réseau pour HTTP et HTTPS est mis en œuvre par-dessus TCP.

#### Note

L'interface de ligne de commande EB et la console Elastic Beanstalk appliquent les valeurs recommandées pour les options précédentes. Vous devez supprimer ces paramètres si vous voulez utiliser des fichiers de configuration pour configurer la même chose. Pour de plus amples informations, veuillez consulter [Valeurs recommandées \(p. 660\)](#).

Dans la tâche suivante, vous devez modifier le groupe de sécurité de l'équilibrEUR de charge pour autoriser le trafic. Selon l'[Amazon Virtual Private Cloud](#) (Amazon VPC) dans lequel vous lancez votre environnement (le VPC par défaut ou un VPC personnalisé), le groupe de sécurité de l'équilibrEUR de charge varie. Dans un VPC par défaut, Elastic Load Balancing fournit un groupe de sécurité par défaut que tous les équilibrEURS de charge peuvent utiliser. Dans un VPC Amazon que vous créez, Elastic Beanstalk crée un groupe de sécurité que l'équilibrEUR de charge doit utiliser.

Pour prendre en charge les deux scénarios, vous pouvez créer un groupe de sécurité et demander à Elastic Beanstalk de l'utiliser. Le fichier de configuration suivant crée un groupe de sécurité et l'associe à l'équilibrEUR de charge.

#### .ebextensions/https-lbsecuritygroup.config

```
option_settings:  
  # Use the custom security group for the load balancer  
  aws:elb:loadbalancer:  
    SecurityGroups: `'{ "Ref" : "loadbalancerssg" }`'  
    ManagedSecurityGroup: `'{ "Ref" : "loadbalancerssg" }`'  
  
Resources:  
  loadbalancerssg:  
    Type: AWS::EC2::SecurityGroup  
    Properties:  
      GroupDescription: load balancer security group
```

```
VpcId: vpc-#####
SecurityGroupIngress:
  - IpProtocol: tcp
    FromPort: 443
    ToPort: 443
    CidrIp: 0.0.0.0/0
  - IpProtocol: tcp
    FromPort: 80
    ToPort: 80
    CidrIp: 0.0.0.0/0
SecurityGroupEgress:
  - IpProtocol: tcp
    FromPort: 80
    ToPort: 80
    CidrIp: 0.0.0.0/0
```

Remplacez le texte en surbrillance par l'ID de votre VPC par défaut ou personnalisé. L'exemple précédent inclut le trafic entrant et sortant via le port 80 pour autoriser les connexions HTTP. Vous pouvez supprimer ces propriétés si vous souhaitez autoriser uniquement les connexions sécurisées.

Enfin, ajoutez des règles de trafic entrant et sortant qui autorisent la communication sur le port 443 entre le groupe de sécurité de l'équilibreur de charge et le groupe de sécurité des instances.

#### **.ebextensions/https-backendsecurity.config**

```
Resources:
# Add 443-inbound to instance security group (AWSEBSecurityGroup)
httpsFromLoadBalancerSG:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
    IpProtocol: tcp
    ToPort: 443
    FromPort: 443
    SourceSecurityGroupId: {"Fn::GetAtt" : ["loadbalancerssg", "GroupId"]}
# Add 443-outbound to load balancer security group (loadbalancerssg)
httpsToBackendInstances:
  Type: AWS::EC2::SecurityGroupEgress
  Properties:
    GroupId: {"Fn::GetAtt" : ["loadbalancerssg", "GroupId"]}
    IpProtocol: tcp
    ToPort: 443
    FromPort: 443
    DestinationSecurityGroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
```

Si vous effectuez cette opération en dehors du processus de création du groupe de sécurité, vous pouvez limiter les groupes de sécurité source et de destination sans créer de dépendance circulaire.

Après avoir effectué toutes les tâches précédentes, l'équilibreur de charge se connecte à vos instances principales en toute sécurité à l'aide du protocole HTTPS. L'équilibreur de charge ne s'occupe pas de savoir si le certificat de votre instance est auto-signé ou a été émis par une autorité de certification compétente : il accepte tous les certificats qui lui sont présentés.

Pour modifier ce comportement, vous pouvez ajouter des stratégies à l'équilibreur de charge en lui demandant de ne faire confiance qu'à un certificat spécifique. Le fichier de configuration suivant crée deux stratégies. La première spécifie un certificat public, et la deuxième indique à l'équilibreur de charge qu'il ne doit faire confiance qu'à ce certificat pour les connexions au port 443 de l'instance.

#### **.ebextensions/https-backendauth.config**

```
option_settings:
```

Remplacez le texte en surbrillance par le contenu du certificat public de votre instance EC2.

## Configuration de l'équilibrer de charge de votre environnement pour TCP Passthrough

Si vous ne voulez pas que l'équilibrer de charge dans votre environnement AWS Elastic Beanstalk déchiffre le trafic HTTPS, vous pouvez configurer l'écouteur sécurisé pour relayer des requêtes vers les instances backend en l'état.

Commencez par [configurer les instances EC2 de votre environnement pour résilier HTTPS](#) (p. 799). Testez la configuration sur un environnement d'instance unique pour vous assurer que tout fonctionne avant d'ajouter un équilibrEUR de charge au mélange.

Ajoutez un [fichier de configuration \(p. 737\)](#) à votre projet pour configurer un auditeur sur le port 443 qui passe des paquets TCP en l'état vers le port 443 sur les instances principales :

`.ebextensions/https-lb-passthrough.config`

```
option_settings:  
    aws:elb:listener:443:  
        ListenerProtocol: TCP  
        InstancePort: 443  
        InstanceProtocol: TCP
```

Dans un VPC [Amazon Virtual Private Cloud](#) (Amazon VPC) par défaut, vous devez également ajouter une règle au groupe de sécurité des instances pour autoriser le trafic entrant sur 443 à partir de l'équilibrEUR de charge :

.ebextensions/https-instance-securitygroup.config

```
Resources:
  443inboundfromloadbalancer:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: {"Fn::GetAtt" : ["AWSEBSecurityGroup", "GroupId"]}
      IpProtocol: tcp
      ToPort: 443
      FromPort: 443
      SourceSecurityGroupName: { "Fn::GetAtt": [ "AWSLoadBalancer", "SourceSecurityGroup.GroupName" ] }
```

Dans un VPC personnalisé, Elastic Beanstalk met automatiquement à jour la configuration du groupe de sécurité.

## Stockage sécurisé des clés privées dans Amazon S3

La clé privée qui vous permet de vous connecter à votre certificat public est privée et ne doit pas être validée dans le code source. Vous pouvez éviter de stocker les clés privées dans les fichiers de configuration en les chargeant dans Amazon S3 et en configurant Elastic Beanstalk pour télécharger le fichier à partir d'Amazon S3 lors du déploiement de l'application.

L'exemple suivant décrit les sections [Ressources \(p. 759\)](#) et [fichiers \(p. 744\)](#) d'un [fichier de configuration \(p. 737\)](#), qui télécharge un fichier de clé privée à partir d'un compartiment Amazon S3.

Example .ebextensions/privatekey.config

```
Resources:  
  AWSEBAutoScalingGroup:  
    Metadata:  
      AWS::CloudFormation::Authentication:  
        S3Auth:  
          type: "s3"  
          buckets: ["elasticbeanstalk-us-west-2-123456789012"]  
          roleName:  
            Fn::GetOptionSetting:  
              Namespace: "aws:autoscaling:launchconfiguration"  
              OptionName: "IamInstanceProfile"  
              DefaultValue: "aws-elasticbeanstalk-ec2-role"  
    files:  
      # Private key  
      "/etc/pki/tls/certs/server.key":  
        mode: "000400"  
        owner: root  
        group: root  
        authentication: "S3Auth"  
        source: https://elasticbeanstalk-us-west-2-123456789012.s3.us-west-2.amazonaws.com/  
server.key
```

Remplacez l'URL et le nom de compartiment figurant dans l'exemple par vos propres valeurs. La première entrée du fichier ajoute une méthode d'authentification nommée S3Auth aux métadonnées du groupe Auto Scaling de l'environnement. Si vous avez configuré un [profil d'instance \(p. 22\)](#) personnalisé pour votre environnement, il est utilisé. Sinon, c'est la valeur par défaut du fichier aws-elasticbeanstalk-ec2-role qui s'applique. Le profil d'instance par défaut est autorisé à lire les données du compartiment de stockage Elastic Beanstalk. Si vous utilisez un autre compartiment, [ajoutez des autorisations au profil d'instance \(p. 925\)](#).

La deuxième entrée utilise la méthode d'authentification S3Auth pour télécharger la clé privée à partir de l'URL spécifiée et pour l'enregistrer dans /etc/pki/tls/certs/server.key. Le serveur proxy peut ensuite lire la clé privée à partir de cet emplacement afin de [mettre les connexions HTTPS hors service dans l'instance \(p. 799\)](#).

Le profil d'instance affecté aux instances EC2 de votre environnement doit être autorisé à lire l'objet clé à partir du compartiment spécifié. [Assurez-vous que le profil d'instance est autorisé \(p. 924\)](#) à lire l'objet dans IAM, et que les autorisations sur le compartiment et l'objet n'entraînent pas d'interdiction pour le profil d'instance.

Pour afficher les autorisations d'un compartiment

1. Ouvrez la [console de gestion Amazon S3](#).
2. Choisissez un compartiment.
3. Choisissez Properties, puis Autorisations.
4. Vérifiez que votre compte bénéficie d'une autorisation de lecture sur le compartiment.

5. Si une stratégie de compartiment est attachée, choisissez Stratégie de compartiment pour afficher les autorisations attribuées au compartiment.

## Configuration de la redirection HTTP vers HTTPS

Dans [Configuration de HTTPS pour votre environnement Elastic Beanstalk \(p. 792\)](#) et ses sous-rubriques, nous aborderons la configuration de votre environnement Elastic Beanstalk pour utiliser HTTPS afin de garantir le chiffrement du trafic dans votre application. Cette rubrique décrit comment faire en sorte que votre application gère élégamment le trafic HTTP si les utilisateurs finaux en sont à l'origine. Pour ce faire, vous devez configurer la redirection de HTTP vers HTTPS, parfois appelée forçage HTTPS.

Pour configurer la redirection, vous devez d'abord configurer votre environnement pour gérer le trafic HTTPS. Ensuite, vous redirigez le trafic HTTP vers HTTPS. Ces deux étapes sont abordées dans les sous-sections suivantes.

### Configurer votre environnement pour gérer le trafic HTTPS

Selon la configuration d'équilibrage de charge de votre environnement, effectuez l'une des opérations suivantes :

- Environnement à charge équilibrée : [configurez votre équilibrEUR de charge pour résilier les connexions HTTPS \(p. 797\)](#).
- Environnement à instance unique : [configurez votre application pour résilier les connexions HTTPS au niveau de l'instance \(p. 799\)](#). Cette configuration dépend de la plate-forme de votre environnement.

### Rediriger le trafic HTTP vers HTTPS

Vous pouvez configurer les serveurs web sur les instances de votre environnement ou l'équilibrEUR Application Load Balancer de l'environnement pour rediriger le trafic HTTP vers HTTPS. Effectuez l'une des actions suivantes :

- Configurer les serveurs web d'instance : cette méthode fonctionne sur n'importe quel environnement de serveur web. Configurez les serveurs web sur vos instances Amazon Elastic Compute Cloud (Amazon EC2) pour répondre au trafic HTTP avec un état de réponse de redirection HTTP. Cette configuration dépend de la plate-forme de votre environnement. Recherchez le dossier de votre plateforme dans la collection [https-redirect](#) sur GitHub et utilisez l'exemple de fichier de configuration dans ce dossier.

Si votre environnement utilise les [vérifications de l'état Elastic Load Balancing \(p. 835\)](#), l'équilibrEUR de charge attend qu'une instance saine réponde aux messages de vérification de l'état HTTP avec des réponses HTTP 200 (OK). Par conséquent, votre serveur web ne devrait pas rediriger ces messages vers HTTPS. Les exemples de fichiers de configuration dans [https-redirect](#) gèrent cette exigence correctement.

- Configurer l'équilibrEUR de charge : cette méthode fonctionne si vous disposez d'un environnement à charge équilibrée qui utilise un [équilibrEUR Application Load Balancer \(p. 574\)](#). L'équilibrEUR Application Load Balancer peut envoyer des réponses de redirection à mesure que le trafic HTTP arrive. Dans ce cas, vous n'avez pas besoin de configurer la redirection sur les instances de votre environnement. Nous avons deux exemples de fichiers de configuration sur GitHub qui montrent comment configurer l'équilibrEUR Application Load Balancer pour la redirection. Le fichier de configuration [alb-http-to-https-redirection-full.config](#) crée un écouteur HTTPS sur le port 443 et modifie l'écouteur de port 80 par défaut pour rediriger le trafic HTTP entrant vers HTTPS. Le fichier de configuration [alb-http-to-https-redirection.config](#) attend que l'écouteur 443 soit défini (vous pouvez utiliser des espaces de noms de configuration Elastic Beanstalk standard ou la console Elastic Beanstalk). Ensuite, il prend soin de modifier l'écouteur du port 80 pour la redirection.

# Surveillance d'un environnement

Lorsque vous exécutez un site web de production, il est important de savoir que votre application est disponible et répond aux demandes. Pour aider à la surveillance de la réactivité de votre application, Elastic Beanstalk offre des fonctionnalités qui surveillent les statistiques concernant votre application et créent des alertes qui se déclenchent quand des seuils sont dépassés.

## Rubriques

- [Surveillance de l'état de l'environnement dans la console de gestionAWS \(p. 830\)](#)
- [Création de rapports d'intégrité de base \(p. 834\)](#)
- [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#)
- [Gestion des alarmes \(p. 874\)](#)
- [Affichage de l'historique des modifications d'un environnement Elastic Beanstalk \(p. 877\)](#)
- [Affichage du flux d'événements d'un environnement Elastic Beanstalk \(p. 879\)](#)
- [Affichage de la liste des instances de serveur et connexion à ces instances \(p. 881\)](#)
- [Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk \(p. 884\)](#)

## Surveillance de l'état de l'environnement dans la console de gestionAWS

Vous pouvez accéder aux informations opérationnelles concernant votre application depuis la console Elastic Beanstalk. La console affiche l'état de votre environnement et l'intégrité de l'application en un coup de œil. Dans la page Environments (Environnements) de la console et dans la page de chaque application, les environnements de la liste sont codés par couleur pour indiquer leur état.

### Pour surveiller un environnement dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Surveillance.

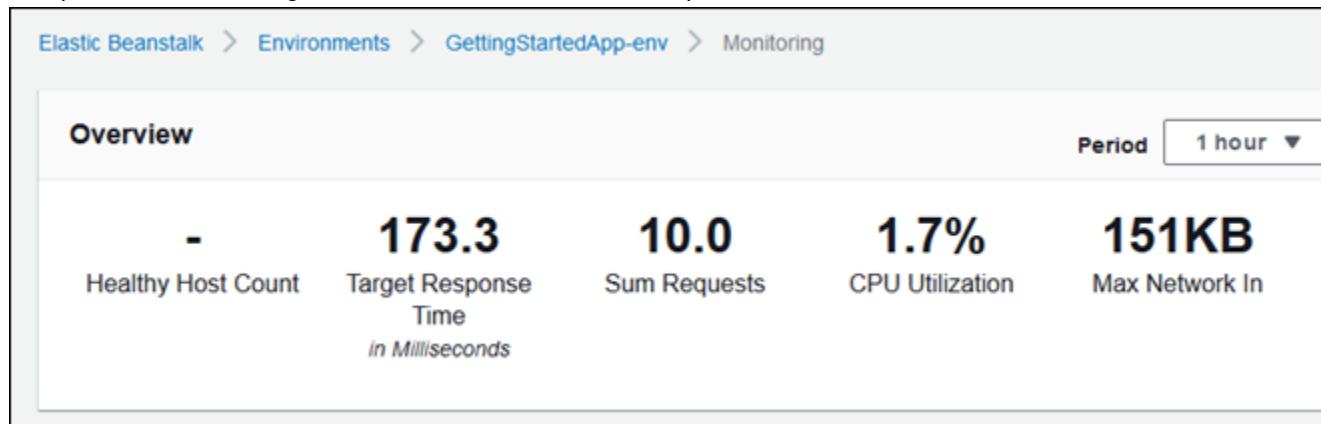
La page Monitoring vous montre les statistiques globales sur votre environnement, telles que l'utilisation de l'UC et la latence moyenne. Outre les statistiques globales, vous pouvez afficher les graphiques de surveillance qui affichent l'utilisation des ressources sur la durée. Vous pouvez cliquer sur n'importe lequel des graphiques pour afficher des informations plus détaillées.

#### Note

Par défaut, seules les métriques CloudWatch de base sont activées, renvoyant des données sur des périodes de cinq minutes. Vous pouvez activer plus de métriques CloudWatch détaillées en une minute en modifiant les paramètres de configuration de votre environnement.

## Présentation

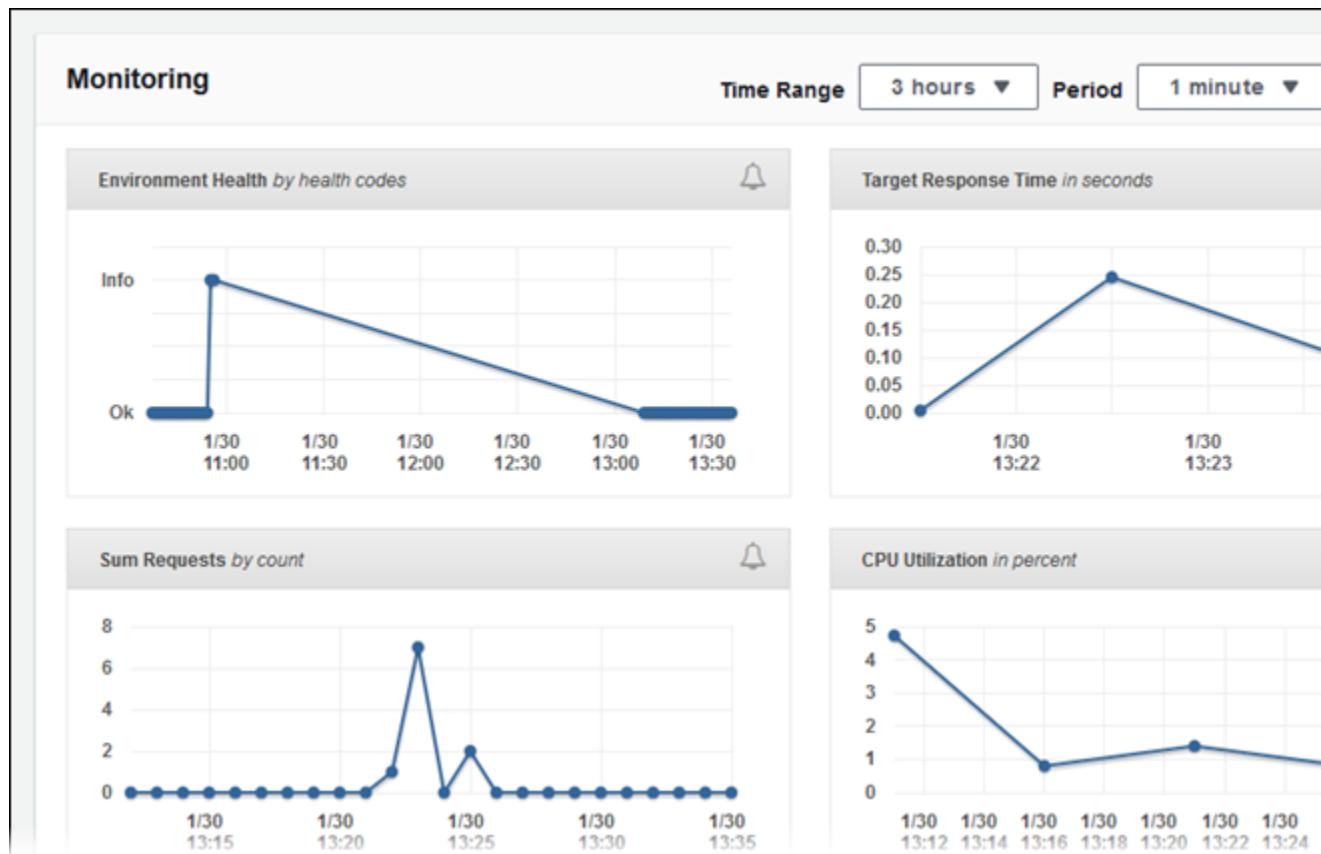
Une présentation de l'intégrité de l'environnement est affichée près du haut de l'écran.



La section de présentation montre un résumé personnalisable de l'activité dans votre environnement sur une période temporelle. Choisissez la liste déroulante Period (Période) et sélectionnez une durée pour afficher les informations sur une période comprise entre une minute et un jour.

## Surveillance des graphiques

Sous la présentation se trouvent des graphiques illustrant des données sur l'intégrité globale de l'environnement au cours d'une période. Choisissez la liste déroulante Period (Période) et sélectionnez une durée pour définir le temps entre les deux points de tracé sur une période comprise entre une minute et un jour. Choisissez la liste déroulante Time range (Plage de temps) et sélectionnez une durée pour définir l'axe temporel du graphique sur une période comprise entre trois heures et deux semaines.



## Personnalisation de la console de surveillance

Cliquez sur Edit (Modifier) à côté de chaque section de surveillance pour personnaliser les informations affichées.

The screenshot shows the AWS Elastic Beanstalk Overview page. At the top, there's a header with 'Overview', 'Period' set to '1 hour', and a 'Cancel' button. Below the header are four cards displaying metrics:

- Healthy Host Count: 173.3
- Target Response Time (in Milliseconds): 10.0
- Sum Requests: 0.8%
- Max Network In: 54KB
- Max Network Out: 91KB

Below these cards is a section titled 'Add Overview' with the following fields:

- Resource: AWSEBV2LoadBalancer
- CloudWatch metric: (empty field)
- Statistic: (empty dropdown)
- Description: (empty field)
- Dimensions: (empty field)

A message at the bottom right of this section says: "Added 'Avg ActiveConnectionCount.' Click Save to preserve your config".

Pour supprimer l'un des éléments existants, choisissez dans le coin supérieur droit.

Pour ajouter une présentation ou un graphique

1. Cliquez sur Edit (Modifier) dans la section Overview (Présentation) ou Monitoring (Surveillance).
2. Sélectionnez une ressource. Les ressources prises en charge sont le groupe Auto Scaling de votre environnement, l'équilibré de charge Elastic Load Balancing et l'environnement lui-même.
3. Sélectionnez une métrique CloudWatch pour la ressource. veuillez consulter [Publication de métriques personnalisées Amazon CloudWatch pour un environnement \(p. 862\)](#) pour obtenir une liste complète des métriques prises en charge.
4. Sélectionnez une statistique. Les statistiques par défaut sont la valeur moyenne de la métrique cloudwatch sélectionnée au cours de la plage de temps (présentation) ou entre les points de traçage (graphique).
5. Saisissez une Description. La description est l'étiquette pour l'élément affiché dans la console de surveillance.
6. Choisissez Add (Ajouter).
7. Répétez les étapes précédentes pour ajouter des éléments ou choisissez Save (Enregistrer) pour finaliser la modification.

Pour de plus amples informations sur les métriques et les dimensions pour chaque ressource, veuillez consulter [Métriques, espaces de noms et référence de dimensions Amazon CloudWatch](#) dans le Guide de l'utilisateur Amazon CloudWatch.

Les métriques [Elastic Load Balancing](#) et [Amazon EC2](#) sont activées pour tous les environnements.

Avec [l'état amélioré \(p. 837\)](#), la métrique `EnvironmentHealth` est activée et un graphique est automatiquement ajouté à la console de surveillance. Des métriques supplémentaires deviennent disponibles pour utilisation dans la console de surveillance lorsque vous les avez activées dans la configuration de l'environnement. L'état amélioré ajoute également la [page Health \(p. 850\)](#) à la console de gestion.

#### Note

Lorsque vous activez des métriques CloudWatch supplémentaires pour votre environnement, il leur faut quelques minutes pour qu'elles commencent à être signalées et à s'afficher dans la liste des métriques que vous utilisez pour ajouter des graphiques et des statistiques de présentation.

Veuillez consulter [Publication de métriques personnalisées Amazon CloudWatch pour un environnement \(p. 862\)](#) pour obtenir une liste des métriques d'état amélioré disponibles.

## Création de rapports d'intégrité de base

AWS Elastic Beanstalk utilise les informations de plusieurs sources afin de déterminer si votre environnement est disponible et pour traiter des demandes provenant d'Internet. L'état de santé d'un environnement est représenté par l'une des quatre couleurs et s'affiche sur la [page de présentation de l'environnement \(p. 429\)](#) de la console Elastic Beanstalk. Il est également disponible à partir de l'API `DescribeEnvironments` et en appelant `eb status` avec l'[interface de ligne de commande \(CLI\) EB \(p. 1027\)](#).

Avant les versions de plateforme Linux version 2, le seul système de rapports sur l'état de santé était le système de rapports de base sur l'état. Le système de création de rapports de base sur l'état fournit des informations sur l'état des instances dans un environnement Elastic Beanstalk en fonction des vérifications de l'état effectuées par Elastic Load Balancing pour les environnements à charge équilibrée ou Amazon Elastic Compute Cloud pour des environnements instance unique.

En plus de vérifier l'état de vos instances EC2, Elastic Beanstalk surveille les autres ressources de votre environnement et signale les ressources manquantes ou mal configurées qui peuvent conduire à une indisponibilité de l'environnement pour les utilisateurs.

Les métriques recueillies par les ressources dans votre environnement sont publiées sur Amazon CloudWatch à intervalles de cinq minutes. Cela inclut des métriques du système d'exploitation d'EC2 ainsi que des métriques de demande d'Elastic Load Balancing. Vous pouvez afficher des graphiques basés sur ces métriques CloudWatch sur la [page Monitoring \(Surveillance\) \(p. 830\)](#) de la console de l'environnement. Pour l'intégrité de base, ces métriques ne sont pas utilisées pour déterminer une intégrité de l'environnement.

#### Rubriques

- [Couleurs de l'intégrité \(p. 835\)](#)
- [Vérifications de l'état Elastic Load Balancing \(p. 835\)](#)
- [Vérifications de l'état d'un environnement à instance unique et d'un environnement de travail \(p. 836\)](#)
- [Contrôles supplémentaires \(p. 836\)](#)
- [Métriques Amazon CloudWatch \(p. 836\)](#)

## Couleurs de l'intégrité

Elastic Beanstalk signale l'état d'un environnement de serveur web en fonction de la façon dont l'application qui s'y exécute répond à la vérification de l'état. Elastic Beanstalk utilise l'une des quatre couleurs pour décrire l'état, comme illustré dans le tableau suivant :

Couleur	Description
Gris	Votre environnement est en cours de mise à jour.
Vert	Votre environnement a passé avec succès la vérification de l'état la plus récente. Au moins une instance dans votre environnement est disponible et accepte des demandes.
Jaune	Votre environnement a échoué à une ou plusieurs vérifications de l'état. Certaines demandes à votre environnement sont en cours d'échec.
Rouge	Votre environnement a échoué à au moins trois vérifications de l'état, ou une ressource d'environnement est devenue indisponible. Les demandes échouent systématiquement.

Ces descriptions s'appliquent uniquement aux environnements utilisant la création de rapports d'intégrité de base. Veuillez consulter [Couleurs et états utilisés dans les rapports d'intégrité \(p. 854\)](#) pour obtenir des détails sur l'état amélioré.

## Vérifications de l'état Elastic Load Balancing

Dans un environnement à charge équilibrée, Elastic Load Balancing envoie une demande à chaque instance dans un environnement toutes les 10 secondes afin de confirmer que les instances sont saines. Par défaut, l'équilibrEUR de charge est configuré pour ouvrir une connexion TCP sur le port 80. Si l'instance reconnaît la connexion, elle est considérée comme saine.

Vous pouvez choisir remplacer ce paramètre en spécifiant une ressource existante dans votre application. Si vous spécifiez un chemin d'accès, tel que /health, l'URL de vérification de l'état est définie sur `HTTP:80/health`. L'URL de vérification de l'état doit être définie sur un chemin d'accès qui est toujours desservi par votre application. Si elle est définie sur une page statique qui est desservie ou mise en cache par le serveur web devant votre application, les vérifications de l'état ne révéleront pas de problèmes avec le serveur d'applications ou le conteneur web. Pour obtenir des instructions sur la modification de votre URL de vérification de l'état, consultez [Vérification de l'état \(p. 571\)](#).

Si une URL de vérification de l'état est configurée, Elastic Load Balancing attend une demande GET qu'il soumet pour renvoyer une réponse 200 OK. L'application échoue à la vérification de l'état en cas de défaut de réponse dans les 5 secondes, ou si la réponse est un code d'état autre que HTTP. Après 5 échecs consécutifs de vérification de l'état, Elastic Load Balancing suspend l'instance.

Pour de plus amples informations sur les vérifications de l'état Elastic Load Balancing, veuillez consulter [Vérification de l'état](#) dans le Guide de l'utilisateur Elastic Load Balancing.

### Note

La configuration d'une URL de vérification de l'état ne modifie pas le comportement de vérification de l'état du groupe Auto Scaling d'un environnement. Une instance non saine est supprimée de l'équilibrEUR de charge, mais n'est pas automatiquement remplacée par Amazon EC2 Auto Scaling, sauf si vous configurez Amazon EC2 Auto Scaling pour utiliser la vérification de l'état Elastic Load Balancing comme base pour le remplacement des instances. Pour configurer Amazon EC2 Auto Scaling pour remplacer les instances qui échouent à une vérification de l'état Elastic Load Balancing, veuillez consulter [Paramètre de vérification de l'état Auto Scaling \(p. 561\)](#).

## Vérifications de l'état d'un environnement à instance unique et d'un environnement de travail

Dans une instance ou un environnement de travail unique, Elastic Beanstalk détermine l'état de l'instance en surveillant son état d'instance Amazon EC2. Les paramètres d'état Elastic Load Balancing, y compris les URL de vérification de l'état HTTP, ne peuvent pas être utilisés dans ces types d'environnement.

Pour de plus amples informations sur les vérifications de l'état des instances Amazon EC2, veuillez consulter [Surveillance des instances avec les vérifications de l'état](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

## Contrôles supplémentaires

Outre les vérifications de l'état Elastic Load Balancing, Elastic Beanstalk surveille les ressources de votre environnement et l'état devient rouge si les ressources ne parviennent pas à se déployer, ne sont pas correctement configurées ou deviennent indisponibles. Ces contrôles confirment ce qui suit :

- Le groupe Auto Scaling de l'environnement est disponible et possède au minimum une instance.
- Le groupe de sécurité de l'environnement est disponible et est configuré pour autoriser le trafic entrant sur le port 80.
- L'environnement CNAME existe et pointe vers l'équilibrEUR de charge approprié.
- Dans un environnement de travail, la file d'attente Amazon Simple Queue Service (Amazon SQS) est interrogée au moins une fois toutes les trois minutes.

## Métriques Amazon CloudWatch

Avec la création de rapports d'état de base, le service Elastic Beanstalk ne publie aucune métriques sur Amazon CloudWatch. Les métriques CloudWatch utilisées pour produire des graphiques sur la [page Monitoring \(Surveillance\) \(p. 830\)](#) de la console d'environnement sont publiées par les ressources de votre environnement.

Par exemple, EC2 publie les métriques suivantes pour les instances dans le groupe Auto Scaling de votre environnement :

### CPUUtilization

Pourcentage d'unités de calcul actuellement en cours d'utilisation.

`DiskReadBytes, DiskReadOps, DiskWriteBytes, DiskWriteOps`

Nombre d'octets lus et écrits et nombre d'opérations de lecture et d'écriture.

`NetworkIn, NetworkOut`

Nombre d'octets envoyés et reçus.

Elastic Load Balancing publie les métriques suivantes pour l'équilibrEUR de charge de votre environnement :

### BackendConnectionErrors

Nombre d'échecs de connexion entre l'équilibrEUR de charge et les instances d'environnement.

`HTTPCode_Backend_2XX, HTTPCode_Backend_4XX`

Nombre de codes de réponses aboutis (2XX) et d'erreur client (4XX) générés par des instances dans votre environnement.

#### Latency

Nombre de secondes entre le moment où l'équilibrEUR de charge relaie une demande à une instance et celui de la réception de la réponse.

#### RequestCount

Nombre de demandes terminées.

Ces listes ne sont pas complÈtes. Pour obtenir une liste complÈte des mÈtriques pouvant Être rapportÈes pour ces ressources, veuillez consulter les rubriques suivantes dans le Manuel du dÈveloppeur Amazon CloudWatch :

#### Metrics

Namespace	Sujet
AWS::ElasticLoadBalancing::LoadBalancer	<a href="#">MÈtriques et ressources Elastic Load Balancing</a>
AWS::AutoScaling::AutoScalingGroup	<a href="#">MÈtriques et ressources Amazon Elastic Compute Cloud</a>
AWS::SQS::Queue	<a href="#">MÈtriques et ressources Amazon SQS</a>
AWS::RDS::DBInstance	<a href="#">Dimensions et mÈtriques Amazon RDS</a>

## MÈtrique d'intÈgritÈ d'environnement de travail

Pour les environnements de travail uniquement, le dÈmon SQS publie une mÈtrique personnalisÈe pour l'Etat de l'environnement sur CloudWatch, oÙ une valeur de 1 est de couleur verte. Vous pouvez passer en revue les donnÈes des mÈtriques d'Etat CloudWatch dans votre compte a l'aide de l'espace de noms `ElasticBeanstalk/SQSD`. La dimension de mÈtrique est `EnvironmentName`, et le nom de mÈtrique est `Health`. Toutes les instances publient leurs mÈtriques sur le mÈme espace de noms.

Pour permettre au dÈmon de publier des mÈtriques, le profil d'instance de l'environnement doit avoir l'autorisation d'appeler `cloudwatch:PutMetricData`. Cette autorisation est incluse dans le profil d'instance par dÈfaut. Pour plus d'informations, consultez [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#).

## Surveillance et crÈation de rapports d'intÈgritÈ amÈliorÈe

La crÈation de rapports d'Etat de santÈ amÈliorÈ est une fonction que vous pouvez activer sur votre environnement pour autoriser AWS Elastic Beanstalk a collecter des informations complÈmentaires sur les ressources de votre environnement. Elastic Beanstalk analyse les informations recueillies pour fournir un meilleur aperÈu de l'Etat global de l'environnement et permettre l'identification des problÈmes pouvant entraîner une indisponibilitÈ de votre application.

En plus des modifications dans le mode de fonctionnement des couleurs d'Etat, l'Etat amÈliorÈ ajoute un descripteur statut qui fournit un indicateur de la gravitÈ des problÈmes observÈs lorsqu'un environnement est jaune ou rouge. Lorsque davantage d'informations sont disponibles sur l'Etat actuel, vous pouvez choisir le bouton Causes pour afficher des informations d'Etat detaillÈes sur la [page d'Etat \(p. 849\)](#).

Elastic Beanstalk > Environments > GettingStartedApp-env

**Elastic Beanstalk is updating your environment.**  
To cancel this operation select **Abort Current Operation** from the **Actions** dropdown.  
[View Events](#)

**GettingStartedApp-env**  Refresh

[GettingStartedApp-env.bx7dx222kv.us-east-2.elasticbeanstalk.com](#)

**Health** Info Causes

**Running version**  
Sample Application Upload and deploy

Tomcat 8  
64bit

**Recent events**

Time	Type	Details
2020-01-28 15:16:51 UTC-0800	INFO	Deploying new version to instance(s).
2020-01-28 15:16:47 UTC-0800	INFO	Environment update is starting.
2020-01-28 12:11:17 UTC-0800	INFO	Environment health has transitioned from Pending to Ok. Initialization completed.

Pour fournir des informations détaillées sur l'état des instances Amazon EC2 s'exécutant dans votre environnement, Elastic Beanstalk inclut un [agent de vérification de l'état \(p. 840\)](#) dans l'AMI (Amazon Machine Image) pour chaque version de plateforme qui prend en charge les rapports améliorés sur l'état. L'agent de vérification de l'état surveille les journaux de serveur web et les métriques système et les transmet au service Elastic Beanstalk. Elastic Beanstalk analyse ces métriques et ces données issues d'Elastic Load Balancing et d'Amazon EC2 Auto Scaling pour fournir un aperçu global de l'état d'un environnement.

Outre la collecte et la présentation des informations relatives aux ressources de votre environnement, Elastic Beanstalk surveille les ressources de votre environnement pour plusieurs conditions d'erreur et fournit des notifications pour vous aider à éviter les défaillances et à résoudre les problèmes de configuration. [Les facteurs susceptibles d'influer sur l'état de votre environnement \(p. 840\)](#) incluent les résultats de chaque demande de votre application, les métriques du système d'exploitation de vos instances et l'état du déploiement le plus récent.

Vous pouvez afficher l'état de santé en temps réel en utilisant la page de [présentation de l'environnement \(p. 849\)](#) de la console Elastic Beanstalk ou la commande `eb health (p. 1053)` de l'interface de ligne de commande Elastic Beanstalk ([p. 1027](#)). Pour enregistrer et suivre l'état de

L'environnement et des instances sur la durée, vous pouvez configurer votre environnement de sorte à publier dans Amazon CloudWatch les informations recueillies par Elastic Beanstalk pour les rapports améliorés sur l'état en tant que métriques personnalisées. Les [frais](#) CloudWatch relatifs aux métriques personnalisées s'appliquent à toutes les métriques autres que `EnvironmentHealth` : ils sont donc gratuits.

Les rapports améliorés sur l'état de santé nécessitent une version 2 ou ultérieure pour la [version de plateforme \(p. 31\)](#). Pour surveiller les ressources et publier des métriques, votre environnement doit avoir à la fois un rôle de [profil d'instance et de service \(p. 837\)](#). La plateforme Docker multi-conteneurs n'inclut pas de serveur web par défaut, mais elle peut être utilisée avec les rapports améliorés sur l'état de santé si vous configurez votre serveur web pour [fournir des journaux au format approprié \(p. 870\)](#).

#### Remarques sur la plateforme Windows

- Cette fonction n'est pas disponible dans les [versions de la plateforme Windows Server](#) antérieures à la version 2 (v2).
- Lorsque vous activez les rapports améliorés sur l'état de santé pour un environnement Windows Server, ne modifiez pas la [configuration de la journalisation IIS](#). Pour que la surveillance améliorée de l'état de santé fonctionne correctement, la journalisation IIS doit être configurée avec le format W3C et les destinations d'événements de journal ETW event only ou Both log file and ETW event.

Par ailleurs, ne désactivez pas ou n'arrêtez pas le service Windows de l'[agent de vérification de l'état Elastic Beanstalk \(p. 840\)](#) sur les instances de votre environnement. Pour collecter et signaler des informations d'état de santé améliorées sur une instance, ce service doit être activé et en cours d'exécution.

L'intégrité améliorée nécessite que l'environnement dispose d'un profil d'instance. Le profil d'instance doit avoir des rôles qui permettent à vos instances d'environnement de collecter et de signaler des informations d'intégrité améliorée. La première fois que vous créez un environnement avec une plateforme v2 dans la console Elastic Beanstalk, Elastic Beanstalk vous invite à créer les rôles requis et active par défaut les rapports améliorés sur l'état. Poursuivez votre lecture pour en savoir plus sur le fonctionnement des rapports améliorés sur l'état, ou consultez [Activation des rapports améliorés sur l'état Elastic Beanstalk \(p. 845\)](#) pour commencer à utiliser immédiatement ces rapports.

Pour pouvoir prendre en charge les rapports améliorés sur l'état sans condition, les plateformes Amazon Linux 2 exigent des profils d'instance. Lorsque vous créez un environnement à l'aide d'une plateforme Amazon Linux 2, Elastic Beanstalk active toujours les rapports améliorés sur l'état, quelle que soit la façon dont vous créez l'environnement (à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande (CLI) EB, de la AWS CLI ou de l'API).

#### Rubriques

- [Agent de vérification de l'état Elastic Beanstalk \(p. 840\)](#)
- [Facteurs de détermination de l'intégrité de l'environnement et de l'instance \(p. 840\)](#)
- [Personnalisation d'une règle de vérification de l'état \(p. 842\)](#)
- [Rôles d'intégrité améliorée \(p. 843\)](#)
- [Autorisation de santé améliorée \(p. 843\)](#)
- [Événements d'intégrité améliorée \(p. 844\)](#)
- [Comportement de la création de rapports d'intégrité améliorée au cours des mises à jour, des déploiements et de la mise à l'échelle \(p. 845\)](#)
- [Activation des rapports améliorés sur l'état Elastic Beanstalk \(p. 845\)](#)
- [Surveillance améliorée de l'état avec la console de gestion de l'environnement \(p. 849\)](#)
- [Couleurs et états utilisés dans les rapports d'intégrité \(p. 854\)](#)
- [Métriques des instances \(p. 856\)](#)

- Configuration de règles d'intégrité améliorée pour un environnement (p. 859)
- Publication de métriques personnalisées Amazon CloudWatch pour un environnement (p. 862)
- Utilisation des rapports améliorés sur l'état à l'aide de l'API Elastic Beanstalk (p. 869)
- Format de journal d'intégrité améliorée (p. 870)
- Notifications et dépannage (p. 873)

## Agent de vérification de l'état Elastic Beanstalk

L'agent de vérification de l'état Elastic Beanstalk est un processus démon (ou un service, dans les environnements Windows) qui s'exécute sur chaque instance Amazon EC2 de votre environnement, en surveillant les métriques d'état au niveau de l'application et du système d'exploitation et en signalant les problèmes à Elastic Beanstalk. L'agent d'état est inclus dans toutes les versions de plateforme Linux à partir de la version 2.0 de chaque plateforme.

L'agent de vérification de l'état rapporte des métriques similaires à celles [publiées sur CloudWatch \(p. 836\)](#) par Amazon EC2 Auto Scaling et Elastic Load Balancing dans le cadre des [rapports de base sur l'état \(p. 834\)](#), y compris la charge de l'UC, les codes HTTP et la latence. Toutefois, l'agent de vérification de l'état rapporte les métriques directement à Elastic Beanstalk, avec une granularité et une fréquence supérieures aux rapports de base sur l'état.

Pour l'intégrité de base, ces métriques sont publiées toutes les cinq minutes et peuvent être contrôlées avec des graphiques dans la console de gestion d'environnement. Avec les rapports améliorés sur l'état, l'agent de vérification de l'état Elastic Beanstalk rapporte des métriques à Elastic Beanstalk toutes les 10 secondes. Elastic Beanstalk utilise les métriques fournies par l'agent de vérification de l'état pour déterminer l'état de santé de chaque instance dans l'environnement et, combinées à d'autres facteurs (p. 840), pour déterminer l'état global de l'environnement.

L'état global de l'environnement peut être affiché en temps réel sur la page de présentation de l'environnement de la console Elastic Beanstalk. Il est également publié dans CloudWatch par Elastic Beanstalk toutes les 60 secondes. Vous pouvez consulter en temps réel les métriques détaillées indiquées par l'agent d'état en utilisant la commande [eb health \(p. 1053\)](#) dans l'[interface de ligne de commande EB \(p. 1027\)](#).

En payant un petit supplément, vous pouvez choisir de publier des métriques individuelles au niveau de l'instance et de l'environnement dans CloudWatch toutes les 60 secondes. Les métriques publiées dans CloudWatch peuvent ensuite être utilisées pour créer des [graphiques de surveillance \(p. 832\)](#) dans la [console de gestion de l'environnement \(p. 429\)](#).

Les rapports améliorés sur l'état n'impliquent un coût que si vous choisissez de publier des métriques améliorées sur l'état dans CloudWatch. Lorsque vous utilisez l'intégrité améliorée, vous obtenez encore les métriques d'intégrité de base publiées gratuitement, même si vous ne choisissez pas de publier des métriques d'intégrité améliorée.

Pour obtenir des détails sur les métriques rapportées par l'agent de vérification de l'état, veuillez consulter [Métriques des instances \(p. 856\)](#). Pour de plus amples informations sur la publication de métriques améliorées sur l'état dans CloudWatch, veuillez consulter [Publication de métriques personnalisées Amazon CloudWatch pour un environnement \(p. 862\)](#).

## Facteurs de détermination de l'intégrité de l'environnement et de l'instance

Outre les vérifications système des rapports de base sur l'état, notamment les [Vérifications de l'état Elastic Load Balancing \(p. 835\)](#) et la [surveillance des ressources \(p. 836\)](#), les rapports améliorés sur l'état

Elastic Beanstalk collectent des données supplémentaires sur l'état des instances de votre environnement. Sont incluses les métriques du système d'exploitation, les journaux de serveur et l'état des opérations d'environnement en cours, telles que les déploiements et les mises à jour. Le service de rapports sur l'état Elastic Beanstalk associe des informations issues de toutes les sources disponibles et les analyse pour évaluer l'état global de l'environnement.

## Opérations et commandes

Lorsque vous effectuez une opération dans votre environnement, telle que le déploiement d'une nouvelle version d'une application, Elastic Beanstalk apporte plusieurs modifications qui affectent l'état de santé de l'environnement.

Par exemple, lorsque vous déployez une nouvelle version d'une application dans un environnement qui exécute plusieurs instances, des messages similaires au suivant s'affichent lorsque vous surveillez l'état de santé de l'environnement [avec l'interface de ligne de commande EB \(p. 1053\)](#).

id	status	cause
Overall	Info	Command is executing on 3 out of 5 instances
i-bb65c145	Pending	91 % of CPU is in use. 24 % in I/O wait
i-ba65c144	Pending	Performing application deployment (running for 31 seconds)
i-f6a2d525	Ok	Performing initialization (running for 12 seconds)
seconds		Application deployment completed 23 seconds ago and took 26
i-e8a2d53b	Pending	94 % of CPU is in use. 52 % in I/O wait
i-e81cca40	Ok	Performing application deployment (running for 33 seconds)

Dans cet exemple, l'état général de l'environnement est Ok et la cause de cet état est que la commande s'exécute sur 3 des 5 instances. Trois des instances dans l'environnement ont le statut Pending, indiquant qu'une opération est en cours.

Lorsqu'une opération est terminée, Elastic Beanstalk rapporte des informations complémentaires sur l'opération. A titre d'exemple, Elastic Beanstalk affiche les informations suivantes sur une instance qui a déjà été mise à jour avec la nouvelle version de l'application :

i-f6a2d525	Ok	Application deployment completed 23 seconds ago and took 26
	seconds	

Les informations sur l'intégrité de l'instance incluent également des détails sur le déploiement le plus récent sur chaque instance dans votre environnement. Chaque instance indique un état et un ID de déploiement. L'ID de déploiement est un nombre entier qui augmente d'un niveau chaque fois que vous déployez une nouvelle version de votre application ou que vous modifiez les paramètres des options de configuration des instances, telles que les variables d'environnement. Vous pouvez utiliser les informations de déploiement pour identifier les instances qui exécutent la mauvaise version de votre application après un échec de [déploiement propagé \(p. 479\)](#).

Dans la colonne de cause, Elastic Beanstalk inclut des messages d'information sur la réussite des opérations et d'autres états sains pour différentes vérifications de l'état. Ces messages ne sont toutefois pas conservés indéfiniment. Les causes des statuts d'environnement défectueux sont conservées jusqu'à ce que l'environnement renvoie un état sain.

## Expiration de commande

Elastic Beanstalk applique un délai d'expiration de commande à partir du moment où une opération commence à autoriser une instance à effectuer la transition vers un état sain. Cette expiration de la

commande est définie dans la configuration de déploiement et de mise à jour de votre environnement (dans l'espace de noms [aws:elasticbeanstalk:command \(p. 699\)](#)) et est paramétrée par défaut sur 10 minutes.

Les mises à jour propagées sont l'occasion pour Elastic Beanstalk d'appliquer un délai d'expiration distinct à chaque lot dans l'opération. Cette expiration est définie dans le cadre de la configuration des mises à jour propagées de l'environnement (dans l'espace de noms [aws:autoscaling:updatepolicy:rollingupdate \(p. 690\)](#)). Si toutes les instances dans le lot sont saines dans le délai d'expiration de la commande, l'opération se poursuit et passe au lot suivant. Dans le cas contraire, l'opération échoue.

#### Note

Si votre application ne réussit pas les vérifications de l'état avec le statut OK, mais qu'elle est stable à un autre niveau, vous pouvez définir l'option `HealthCheckSuccessThreshold` dans l'espace de noms [aws:elasticbeanstalk:command namespace \(p. 699\)](#) afin de modifier le niveau auquel Elastic Beanstalk considère une instance comme étant saine.

Pour qu'un environnement de serveur web soit considéré comme sain, chaque instance dans l'environnement ou le lot chaque instance doit réussir 12 vérifications de l'état consécutives en deux minutes. Dans un environnement de travail, chaque instance doit réussir 18 vérifications de l'état. Avant l'expiration de la commande, Elastic Beanstalk n'abaisse pas l'état de santé d'un environnement lorsque les vérifications de l'état échouent. Si les instances de l'environnement deviennent saines avant l'expiration de la commande, l'opération réussit.

## Requêtes HTTP

Lorsqu'aucune opération n'est en cours sur un environnement, la source principale d'informations sur l'intégrité de l'instance et de l'environnement repose sur les journaux de serveur web pour chaque instance. Pour déterminer l'état d'une instance et l'état global de l'environnement, Elastic Beanstalk prend en compte le nombre de demandes, le résultat de chaque demande et la vitesse à laquelle chaque demande a été résolue.

Sur les plateformes Linux, Elastic Beanstalk lit et analyse les journaux des serveurs web pour obtenir des informations sur les demandes HTTP. Sur la plateforme Windows Server, Elastic Beanstalk reçoit [directement ces informations du serveur web IIS \(p. 858\)](#).

Votre environnement peut ne pas avoir de serveur web actif. Par exemple, la plateforme Docker multi-conteneurs n'inclut pas de serveur web. Les autres plateformes comprennent un serveur web, et votre application peut le désactiver. Dans ces cas-là, votre environnement exige une configuration supplémentaire pour fournir à l'[agent de vérification de l'état Elastic Beanstalk \(p. 840\)](#) les journaux au format dont il a besoin pour transmettre les informations d'état au service Elastic Beanstalk. Consultez [Format de journal d'intégrité améliorée \(p. 870\)](#) pour plus de détails.

## Métriques du système d'exploitation

Elastic Beanstalk surveille les métriques du système d'exploitation rapportées par l'agent de vérification de l'état pour identifier les instances qui sont constamment à court de ressources système.

Pour obtenir des détails sur les métriques rapportées par l'agent de vérification de l'état, veuillez consulter [Métriques des instances \(p. 856\)](#).

## Personnalisation d'une règle de vérification de l'état

Le rapport sur l'intégrité améliorée d'Elastic Beanstalk s'appuie sur un ensemble de règles qui déterminent l'intégrité de votre environnement. Certaines de ces règles peuvent ne pas être adaptées à votre application. Un cas courant est une application qui renvoie de fréquence erreurs HTTP 4xx en raison de

sa conception. Elastic Beanstalk utilise l'une de ses règles par défaut pour conclure que quelque chose ne fonctionne pas correctement, puis modifie l'état de santé de votre environnement de OK à Avertissement, Dégradé ou Grave, en fonction du taux d'erreur. Pour gérer ce cas correctement, Elastic Beanstalk vous permet de configurer cette règle et d'ignorer les erreurs HTTP 4xx de l'application. Pour plus d'informations, consultez [Configuration de règles d'intégrité améliorée pour un environnement \(p. 859\)](#).

## Rôles d'intégrité améliorée

Les rapports améliorés sur l'état exigent deux rôles : un rôle de service pour Elastic Beanstalk et un profil d'instance pour l'environnement. La fonction du service permet à Elastic Beanstalk d'interagir avec d'autres services AWS en votre nom afin de recueillir des informations sur les ressources de votre environnement. Le profil d'instance permet aux instances de votre environnement d'écrire des journaux dans Amazon S3 et de communiquer des informations améliorées sur l'état au service Elastic Beanstalk.

Lorsque vous créez un environnement Elastic Beanstalk à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande EB, Elastic Beanstalk crée un rôle de service par défaut et attache les stratégies gérées requises à un profil d'instance par défaut pour votre environnement.

Si vous utilisez l'API, un kit SDK ou la AWS CLI pour créer des environnements, vous devez créer ces rôles à l'avance et les spécifier lors de la création de l'environnement pour utiliser les rapports améliorés sur l'état de santé. Pour obtenir des instructions sur la création de rôles appropriés pour vos environnements, veuillez consulter [Rôles de service, profils d'instance et stratégies utilisateur \(p. 21\)](#).

Nous vous recommandons d'utiliser des stratégies gérées pour votre profil d'instance et votre rôle de service. Les stratégies gérées sont des stratégies AWS Identity and Access Management (IAM) gérées par Elastic Beanstalk. L'utilisation de stratégies gérées garantit que votre environnement dispose de toutes les autorisations nécessaires pour fonctionner correctement.

Pour le profil d'instance, vous pouvez utiliser la stratégie `AWSElasticBeanstalkWebTier` ou `AWSElasticBeanstalkWorkerTier` gérée, pour un environnement de [niveau serveur web \(p. 15\)](#) ou de [niveau de travail \(p. 16\)](#), respectivement. Pour de plus amples informations sur ces deux stratégies de profil d'instance gérées, veuillez consulter [the section called "Profils d'instance" \(p. 921\)](#).

## Autorisation de santé améliorée

Les stratégies gérées du profil d'instance Elastic Beanstalk contiennent des autorisations pour l'action `elasticbeanstalk:PutInstanceStatistics`. Cette action ne fait pas partie de l'API Elastic Beanstalk. Elle fait partie d'une autre API que les instances d'environnement utilisent en interne pour communiquer des informations améliorées sur l'état au service Elastic Beanstalk. Vous nappelez pas cette API directement.

Lorsque vous créez un environnement, l'autorisation de l'action `elasticbeanstalk:PutInstanceStatistics` est activée par défaut. Pour renforcer la sécurité de votre environnement et prévenir l'usurpation des données d'état de santé en votre nom, nous vous recommandons de garder l'autorisation de cette action activée. Si vous utilisez des stratégies gérées pour votre profil d'instance, cette fonction est disponible pour votre nouvel environnement sans aucune autre configuration. Si vous utilisez un profil d'instance personnalisé au lieu d'une stratégie gérée, votre environnement peut afficher un état de santé Aucune donnée. Cela se produit car les instances ne sont pas autorisées pour l'action qui communique des données d'intégrité améliorées au service.

Pour autoriser l'action, incluez l'instruction suivante dans votre profil d'instance.

```
{  
    "Sid": "ElasticBeanstalkHealthAccess",  
    "Action": [  
        "elasticbeanstalk:PutInstanceStatistics"
```

```
],
"Effect": "Allow",
"Resource": [
    "arn:aws:elasticbeanstalk:*:application/*",
    "arn:aws:elasticbeanstalk:*:environment/*"
]
}
```

Si vous ne souhaitez pas utiliser l'autorisation d'état de santé améliorée pour le moment, désactivez-la en définissant l'option `EnhancedHealthAuthEnabled` dans l'espace de noms [the section called “aws:elasticbeanstalk:healthreporting:system” \(p. 706\)](#) sur `false`. Si cette option est désactivée, les autorisations décrites précédemment ne sont pas requises. Vous pouvez les supprimer du profil d'instance pour accorder un [accès sur la base du moindre privilège \(p. 1138\)](#) à vos applications et environnements.

#### Note

Auparavant, le paramètre par défaut pour `EnhancedHealthAuthEnabled` était `false`, ce entraînait la désactivation par défaut de l'autorisation de l'action `elasticbeanstalk:PutInstanceStatistics`. Pour activer cette action pour un environnement existant, définissez l'option `EnhancedHealthAuthEnabled` dans l'espace de noms [the section called “aws:elasticbeanstalk:healthreporting:system” \(p. 706\)](#) sur `true`. Vous pouvez configurer cette option à l'aide d'un paramètre d'option (p. 738) dans un [fichier de configuration \(p. 737\)](#).

## Événements d'intégrité améliorée

Le système d'intégrité améliorée génère des événements lorsqu'un environnement effectue la transition entre différents états. L'exemple suivant illustre la sortie d'événements provenant d'un environnement effectuant une transition entre les états Infos, OK et Grave.

Recent events		
Time	Type	Details
2020-01-28 16:06:04 UTC-0800	INFO	Environment health has transitioned from Severe to Ok.
2020-01-28 16:05:04 UTC-0800	INFO	Added instance [i-03280193ba1ba4171] to your environment.
2020-01-28 16:05:04 UTC-0800	WARN	Removed instance [i-0a4a27bbbff9994ba5] from your environment due to a EC2 health check failure.
2020-01-28 16:03:04 UTC-0800	WARN	Environment health has transitioned from Ok to Severe. ELB processes are not healthy on all instances. Instances are sending data. ELB health is failing or not available for all instances.
2020-01-28 15:19:06 UTC-0800	INFO	Environment health has transitioned from Info to Ok. Application update completed 75 seconds ago.

Lors du passage à un état dégradé, l'événement de rapport amélioré sur l'état de santé inclut un message indiquant la cause de la transition.

Les changements de statut au niveau de l'instance ne conduisent pas tous Elastic Beanstalk à émettre un événement. Pour éviter les fausses alarmes, Elastic Beanstalk ne génère d'événement d'état que si un problème persiste dans plusieurs vérifications.

Des informations d'état au niveau de l'environnement en temps réel, y compris le statut, la couleur et la cause, sont disponibles sur la page de [présentation de l'environnement \(p. 430\)](#) de la console Elastic Beanstalk et de l'[interface de ligne de commande EB \(p. 1027\)](#). En associant l'interface de ligne de commande EB à votre environnement et en exécutant la commande `eb health (p. 1053)`, vous pouvez aussi afficher les statuts en temps réel de chacune des instances dans votre environnement.

## Comportement de la création de rapports d'intégrité améliorée au cours des mises à jour, des déploiements et de la mise à l'échelle

Activer la création de rapports d'intégrité améliorée peut affecter le comportement de votre environnement au cours des déploiements et des mises à jour de configuration. Elastic Beanstalk ne termine pas un lot de mises à jour tant que toutes les instances n'ont pas réussi les vérifications de l'état. Par ailleurs, étant donné que les rapports améliorés sur l'état de santé appliquent un standard supérieur et surveille plusieurs facteurs, les instances qui réussissent la [vérification de l'état ELB \(p. 835\)](#) des rapports basiques sur l'état de santé ne réussissent pas forcément dans les rapports améliorés sur l'état de santé. Consultez les rubriques sur [rolling configuration updates \(p. 489\)](#) et [rolling deployments \(p. 479\)](#) pour plus d'informations sur la façon dont les vérifications de l'état affectent le processus de mise à jour.

Les rapports améliorés sur l'état peuvent également mettre en évidence la nécessité de définir une [URL de vérification de l'état \(p. 571\)](#) correcte pour Elastic Load Balancing. Lorsque votre environnement s'adapte pour répondre à la demande, de nouvelles instances commencent à accepter des demandes dès qu'elles réussissent suffisamment de vérifications de l'état ELB. Si aucune URL de vérification de l'état n'est configurée, le délai après qu'une nouvelle instance soit capable d'accepter une connexion TCP peut se réduire à seulement 20 secondes.

Si votre application n'a pas fini de démarrer au moment où l'équilibrEUR de charge la déclare suffisamment saine pour recevoir du trafic, vous verrez un flot de demandes ayant échoué, et votre environnement commencera à échouer à des vérifications de l'état. Une URL de vérification de l'état qui atteint un chemin servi par votre application peut éviter ce problème. Les vérifications de l'état ELB ne réussissent pas tant qu'une demande GET adressée à l'URL de vérification de l'état n'a pas renvoyé un code de statut 200.

## Activation des rapports améliorés sur l'état Elastic Beanstalk

Les nouveaux environnements créés avec les dernières [versions de plateforme \(p. 31\)](#) incluent l'[agent de vérification de l'état \(p. 840\)](#) AWS Elastic Beanstalk, qui prend en charge la création des rapports améliorés sur l'état. Si vous créez votre environnement dans la console Elastic Beanstalk ou avec l'[interface de ligne de commande EB](#), les rapports améliorés sur l'état sont activés par défaut. Vous pouvez également définir vos préférences relatives aux rapports améliorés sur l'état dans le code source de votre application, à l'aide des [fichiers de configuration \(p. 737\)](#).

Les rapports améliorés sur l'état nécessitent un [profil d'instance \(p. 22\)](#) et un [rôle de service \(p. 21\)](#) incluant l'ensemble standard d'autorisations. Lorsque vous créez un environnement dans la console Elastic Beanstalk, Elastic Beanstalk crée automatiquement les rôles nécessaires. Pour obtenir des instructions sur la création de votre premier environnement, veuillez consulter [Mise en route avec Elastic Beanstalk \(p. 3\)](#).

### Rubriques

- [Activation des rapports améliorés sur l'état à l'aide de la console Elastic Beanstalk \(p. 846\)](#)
- [Activation des rapports d'intégrité améliorée via l'interface de ligne de commande EB \(p. 847\)](#)
- [Activation des rapports d'intégrité améliorées via un fichier de configuration \(p. 848\)](#)

## Activation des rapports améliorés sur l'état à l'aide de la console Elastic Beanstalk

Pour activer les rapports améliorés sur l'état dans un environnement en cours d'exécution à l'aide de la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

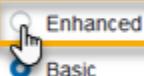
3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Dans la section Rapport sur l'état de santé, choisissez Amélioré dans le champ Présentation.

## Modify monitoring

### Health reporting

Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. A custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see [Amazon CloudWatch Metrics](#).

System



CloudWatch Custom Metrics - Instance

[Choose metrics](#)

CloudWatch Custom Metrics - Environment

[Choose metrics](#)

[Cancel](#)

### Note

Les options relatives aux rapports améliorés sur l'état de santé ne s'affichent pas si vous utilisez une [plateforme ou une version non prise en charge \(p. 837\)](#).

6. Choisissez Apply.

La console Elastic Beanstalk active par défaut les rapports améliorés sur l'état lorsque vous créez un environnement avec la version 2 (v2) de la plateforme. Vous pouvez désactiver les rapports améliorés sur l'état en modifiant l'option des rapports sur l'état lors de la création de l'environnement.

Pour désactiver les rapports améliorés sur l'état lors de la création d'un environnement à l'aide de la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Créez une application (p. 406) ou sélectionnez une application existante.
3. Créez un environnement (p. 439). Sur la page Créer un nouvel environnement, avant de choisir Créer un environnement, choisissez Configurer plus d'options.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Dans la section Rapport sur l'état de santé, choisissez Basique dans le champ Présentation.

## Modify monitoring

**Health reporting**  
Enhanced health reporting provides free real-time application and operating system monitoring of the instances and other resources in your environment. A custom metric is provided free with enhanced health reporting. Additional charges apply for each custom metric. For more information, see [Amazon CloudWatch Metrics](#).

System  
 Enhanced  
 Basic

CloudWatch Custom Metrics - Instance  
[Choose metrics](#)

CloudWatch Custom Metrics - Environment  
[Choose metrics](#)

**Health monitoring rule customization**  
Configure the HTTP application and load balancer static codes included in determining your environment's health. [Learn more](#)

6. Choisissez Enregistrer.

## Activation des rapports d'intégrité améliorée via l'interface de ligne de commande EB

Lorsque vous créez un environnement avec la commande eb create, l'interface de ligne de commande EB active les rapports améliorés sur l'état par défaut, et applique le rôle de service et le profil d'instance par défaut.

Vous pouvez spécifier un autre rôle de service par nom en utilisant l'option `--service-role`.

Si votre environnement est exécuté dans une version v2 de la plateforme avec des rapports basiques sur l'état de santé et que vous souhaitez passer aux rapports améliorés sur l'état de santé, procédez comme suit.

Pour activer les rapports améliorés sur l'état dans un environnement en cours d'exécution via l'[interface de ligne de commande EB \(p. 1027\)](#)

1. Utilisez la commande eb config pour ouvrir le fichier de configuration dans l'éditeur de texte par défaut.

```
~/project$ eb config
```

2. Recherchez l'espace de noms aws:elasticbeanstalk:environment dans la section des paramètres. Assurez-vous que la valeur de ServiceRole n'est pas nulle et qu'elle correspond au nom de votre [rôle de service \(p. 21\)](#).

```
aws:elasticbeanstalk:environment:  
  EnvironmentType: LoadBalanced  
  ServiceRole: aws-elasticbeanstalk-service-role
```

3. Sous l'espace de noms aws:elasticbeanstalk:healthreporting:system:, remplacez la valeur SystemType par **enhanced**.

```
aws:elasticbeanstalk:healthreporting:system:  
  SystemType: enhanced
```

4. Enregistrez le fichier de configuration et fermez l'éditeur de texte.
5. L'interface de ligne de commande EB lance une mise à jour de l'environnement pour appliquer les modifications apportées à la configuration. Attendez la fin de l'opération ou appuyez sur Ctrl+C pour quitter l'interface en toute sécurité.

```
~/project$ eb config  
Printing Status:  
INFO: Environment update is starting.  
INFO: Health reporting type changed to ENHANCED.  
INFO: Updating environment no-role-test's configuration settings.
```

## Activation des rapports d'intégrité améliorées via un fichier de configuration

Vous pouvez activer les rapports améliorés sur l'état en incluant un [fichier de configuration \(p. 737\)](#) dans votre bundle source. L'exemple suivant présente un fichier de configuration qui active les rapports améliorés sur l'état et affecte le rôle de service et le profil d'instance par défaut à l'environnement :

Example .ebextensions/enhanced-health.config

```
option_settings:  
  aws:elasticbeanstalk:healthreporting:system:  
    SystemType: enhanced  
  aws:autoscaling:launchconfiguration:  
    IamInstanceProfile: aws-elasticbeanstalk-ec2-role  
  aws:elasticbeanstalk:environment:  
    ServiceRole: aws-elasticbeanstalk-service-role
```

Si vous avez créé votre propre rôle de service ou profil d'instance, remplacez le texte en surbrillance par les noms de ces rôles.

## Surveillance améliorée de l'état avec la console de gestion de l'environnement

Une fois que vous avez activé les rapports améliorés sur l'état dans AWS Elastic Beanstalk, vous pouvez surveiller l'état de l'environnement dans la [console de gestion de l'environnement \(p. 429\)](#).

### Rubriques

- [Présentation de l'environnement \(p. 849\)](#)
- [Page « Health \(Santé\) » de l'environnement \(p. 850\)](#)
- [Page « Monitoring \(Surveillance\) » \(p. 854\)](#)

## Présentation de l'environnement

La [présentation de l'environnement \(p. 430\)](#) affiche les [statuts d'état \(p. 854\)](#) de l'environnement et répertorie les événements qui apportent des informations sur l'évolution récente des statuts d'état.

Pour afficher la présentation de l'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

Pour plus d'informations sur l'état de santé de l'environnement actuel, ouvrez la page Santé en choisissant Causes. Sinon, dans le panneau de navigation, sélectionnez Health (Santé).

Time	Type	Details
2020-01-28 16:03:04 UTC-0800	WARN	Environment health has transitioned from Ok to Severe. ELB processes are not healthy on all instances. Instances are sending data. ELB health is failing or not available for all instances.
2020-01-28 15:19:06 UTC-0800	INFO	Environment health has transitioned from Info to Ok. Application update completed 75 seconds ago.

## Page « Health (Santé) » de l'environnement

La page Health (État) affiche l'état de santé, les métriques et les causes concernant l'environnement et chaque instance Amazon EC2 de l'environnement.

### Note

Elastic Beanstalk affiche la page Health (État) uniquement si vous avez [activé la surveillance améliorée de l'état \(p. 845\)](#) pour l'environnement.

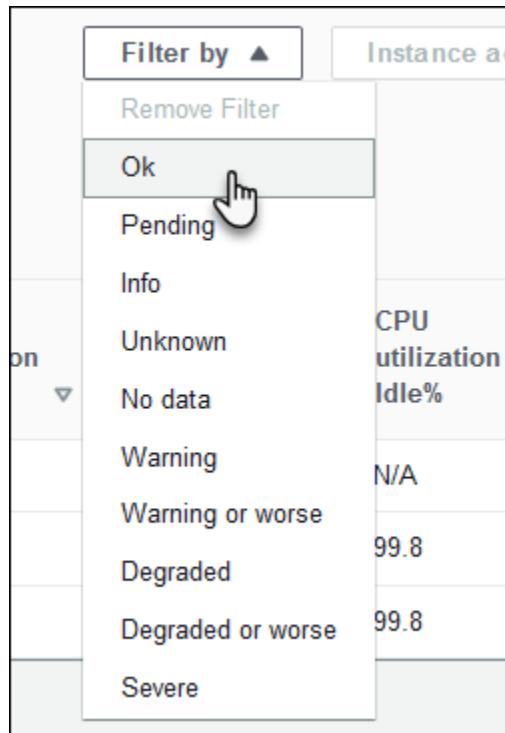
L'image suivante montre la page Health (Santé) pour un environnement Linux.

Instance ID	Status	Running	Deployment	Requests/sec	2xx Responses	3xx Responses	4xx Responses	5xx Responses	P99 Latency	P90 Latency
Overall	Ok	N/A	N/A	0.4	100%	0.0%	0.0%	0.0%	0.002	0.00
i-00227807c4c4a1334	Ok	2 hours	3	0.2	2	0	0	0	0.002	0.00
i-03280193ba1ba4171	Ok	19 days	3	0.2	2	0	0	0	0.001	0.00

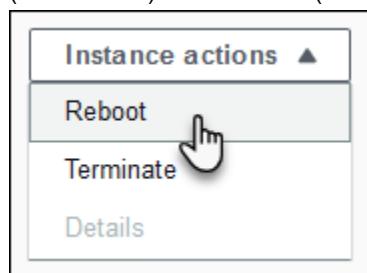
L'image suivante montre la page Health (Santé) pour un environnement Windows. Notez que les métriques d'UC sont différentes de celles d'un environnement Linux.

Instance ID	Status	Running	Deployment ID	Requests/sec	2xx Responses	3xx Responses	4xx Responses	5xx Responses	P99 Latency
Overall	Ok	N/A	N/A	0.2	100%	0.0%	0.0%	0.0%	0.015
i-04b3b4c983018af	Ok	20 days	1	0.2	2	0	0	0	0.015

En haut de la page, vous pouvez voir le nombre total d'instances d'environnement, ainsi que le nombre d'instances par état. Pour afficher uniquement les instances associées à un état particulier, choisissez Filter By (Filtrer par), puis choisissez un état (p. 854).



Pour redémarrer ou suspendre une instance défectueuse, choisissez Actions d'instance, puis Reboot (Redémarrer) ou Terminate (Résilier).



Elastic Beanstalk met à jour la page Health (État) toutes les 10 secondes. Il fournit des informations sur la santé de l'environnement et de l'instance.

Pour chaque instance Amazon EC2 de l'environnement, la page affiche l'ID et le [statut \(p. 854\)](#) de l'instance, le temps écoulé depuis le lancement de l'instance, l'ID du déploiement le plus récent exécuté sur l'instance, les réponses et la latence des demandes traitées par l'instance, ainsi que des informations sur la charge et l'utilisation de l'UC. La ligne Overall (Globale) affiche des informations sur la réponse moyenne et la latence pour l'ensemble de l'environnement.

La page affiche de nombreux détails dans un très grand tableau. Pour masquer certaines colonnes,

choisissez  (Preferences (Préférences)). Sélectionnez ou effacez les noms de colonne, puis choisissez Confirme (Confirmer).

**Preferences**

Options
Visible columns
Instance ID
Status
Running
Deployment ID
Requests/sec
2xx Responses
3xx Responses

Choisissez l'Instance ID (ID d'instance) de n'importe quelle instance pour afficher plus d'informations sur cette dernière, y compris sa zone de disponibilité et le type d'instance.

Instance ID	Status	Running	Deployment ID	Request
Overall	Ok	N/A	N/A	0.2
i-00227807c4c4a1334	Ok	1 day	3	0.1
i-03280193ba1ba4171	Ok	20 days	3	0.1



### i-00227807c4c4a1334 details



Instance ID: i-00227807c4c4a1334

Instance type: t2.micro

Availability zone: us-east-2b

Choisissez l'Deployment ID (ID de déploiement) d'une instance pour afficher des informations sur le dernier déploiement ([p. 476](#)) sur l'instance.

Instance ID	Status	Running	Deployment ID	Request
Overall	Ok	N/A	N/A	0.2
i-00227807c4c4a1334	Ok	1 day	3	0.1
i-03280193ba1ba4171	Ok	20 days	3	0.1



### Deployment details



Deployment ID 3

Version: Sample Application-3

Deployed 1 day ago

Les informations sur le déploiement incluent les éléments suivants :

- ID de déploiement — Identifiant unique du [déploiement \(p. 476\)](#). Les ID de déploiement commencent par 1 et augmentent d'une unité chaque fois que vous déployez une nouvelle version d'application ou modifiez des paramètres de configuration qui ont un impact sur le logiciel ou le système d'exploitation exécuté sur les instances de votre environnement.
- Version — Libellé de version du code source de l'application utilisé dans le déploiement.
- Statut — Statut du déploiement ([In Progress](#), [Deployed](#) ou [Failed](#)).
- Durée écoulée — Durée écoulée depuis le début du déploiement (pour les déploiements en cours) ou durée écoulée depuis la fin du déploiement (pour les déploiements terminés).

Si vous [activez l'intégration à X-Ray \(p. 638\)](#) dans votre environnement et que vous instrumentez votre application avec le kit SDK AWS X-Ray, la page Health (Santé) ajoute des liens à la console AWS X-Ray dans la ligne de présentation.

Tests/sec ▾	2xx Responses ▾	3xx Responses ▾	4xx Responses ▾	5xx Responses ▾	P99 Latency ▾	P90 Latency ▾	P75 Latency ▾	P50 Latency ▾
	100%	0.0%	0.0%	0.0%	0.002	0.002	0.002	0.002
1	0	0	0	0	0.002	0.002	0.002	0.002
1	0	0	0	0	0.001	0.001	0.001	0.001

Choisissez un lien afin d'afficher les traces relatives aux statistiques mises en surbrillance dans la console AWS X-Ray.

## Page « Monitoring (Surveillance) »

La page Monitoring (Surveillance) affiche un récapitulatif des statistiques et des graphiques relatifs aux métriques Amazon CloudWatch personnalisées qui sont générées par le système de génération de rapports améliorés sur l'état. Pour savoir comment ajouter des graphiques et des statistiques à cette page, consultez [Surveillance de l'état de l'environnement dans la console de gestionAWS \(p. 830\)](#).

## Couleurs et états utilisés dans les rapports d'intégrité

Les rapports améliorés sur l'état de santé indiquent l'état de santé des instances et de l'environnement global à l'aide de quatre couleurs, comme dans les [rapports basiques sur l'état de santé \(p. 834\)](#). Les rapports améliorés sur l'état de santé incluent également sept statuts d'état de santé. Ce sont des termes descriptifs, composés d'un seul mot, qui vous permettent de mieux comprendre l'état de santé de votre environnement.

## État de l'instance et état de l'environnement

Chaque fois qu'Elastic Beanstalk vérifie l'état de votre environnement, les rapports améliorés sur l'état vérifient l'état de chaque instance de votre environnement, en analysant l'ensemble des [données \(p. 840\)](#) disponibles. En cas d'échec de l'une des vérifications de niveau inférieur, Elastic Beanstalk rétrograde l'état de l'instance.

Elastic Beanstalk affiche les informations sur l'état pour l'environnement global (couleur, statut et cause) dans la [console de gestion de l'environnement \(p. 429\)](#). Ces informations sont également disponibles dans l'interface de ligne de commande EB. Le statut d'état de santé et les messages d'explication sont mis à jour toutes les 10 secondes pour chaque instance, et sont disponibles via l'[interface de ligne de commande EB \(p. 1027\)](#) lorsque vous affichez le statut d'état de santé avec [eb health \(p. 1053\)](#).

Elastic Beanstalk tire parti des changements d'état des instances pour évaluer l'état de l'environnement, mais ne change pas immédiatement l'état de santé de l'environnement. Lorsque la vérification de l'état d'une instance échoue au moins trois fois dans un laps de temps d'une minute, Elastic Beanstalk peut rétrograder l'état de l'environnement. En fonction du nombre d'instances de l'environnement et du problème identifié, une instance défaillante peut conduire Elastic Beanstalk à afficher un message d'information ou à changer l'état de santé de l'environnement en le faisant passer de vert (OK) à jaune (Avertissement) ou rouge (Dégradé ou Grave).

## OK (vert)

Ce statut s'affiche lorsque :

- L'instance réussit les vérifications de l'état et l'agent d'état ne signale aucun problème.
- La plupart des instances de l'environnement réussissent les vérifications de l'état et l'agent d'état ne signale aucun problème majeur.
- L'instance réussit les vérifications de l'état et traite les demandes normalement.

Exemple : votre environnement a été déployé récemment et accepte les demandes normalement. 5 % de demandes renvoient des erreurs de la série 400. Le déploiement s'est terminé normalement sur chaque instance.

Message (instance) : Application deployment completed 23 seconds ago and took 26 seconds.

## Avertissement (jaune)

Ce statut s'affiche lorsque :

- L'agent d'état signale un nombre modéré d'échecs de la demande ou d'autres problèmes pour une instance ou un environnement.
- Une opération est en cours sur une instance et prend beaucoup de temps.

Exemple : une instance de l'environnement a le statut Grave.

Message (environnement) : Impaired services on 1 out of 5 instances.

## Dégradé (rouge)

Ce statut s'affiche lorsque l'agent d'état signale un nombre élevé d'échecs de la demande ou d'autres problèmes pour une instance ou un environnement.

Exemple : l'environnement est en cours de mise à l'échelle ascendante vers 5 instances.

Message (environnement) : 4 active instances is below Auto Scaling group minimum size 5.

## Grave (rouge)

Ce statut s'affiche lorsque l'agent d'état signale un nombre très élevé d'échecs de la demande ou d'autres problèmes pour une instance ou un environnement.

Exemple : Elastic Beanstalk ne parvient pas à contacter l'équilibrer de charge pour obtenir l'état de l'instance.

Message (environnement) : ELB health is failing or not available for all instances. None of the instances are sending data. Unable to assume role "arn:aws:iam::123456789012:role/aws-elasticbeanstalk-service-role". Verify that the role exists and is configured correctly.

Message (Instances) : Instance ELB health has not been available for 37 minutes. Pas de données. Last seen 37 minutes ago.

## Info (vert)

Ce statut s'affiche lorsque :

- Une opération est en cours sur une instance.
- Une opération est en cours sur plusieurs instances d'un environnement.

Exemple : une nouvelle version d'application est en cours de déploiement dans les instances en cours d'exécution.

Message (environnement) : Command is executing on 3 out of 5 instances.

Message (instance) : Performing application deployment (running for 3 seconds).

## En attente (gris)

Ce statut s'affiche lorsqu'une opération est en cours sur une instance et que le [délai d'attente de la commande \(p. 841\)](#) n'est pas dépassé.

Exemple : vous avez créé l'environnement récemment et des instances sont en cours d'amorçage.

Message : Performing initialization (running for 12 seconds).

## Inconnu (gris)

Ce statut s'affiche lorsqu'Elastic Beanstalk et l'agent de vérification de l'état signalent une quantité de données insuffisante sur une instance.

Exemple : aucune donnée n'est reçue.

## Suspendu (gris)

Ce statut s'affiche lorsqu'Elastic Beanstalk cesse de surveiller l'état de l'environnement. L'environnement peut ne pas fonctionner correctement. Certaines conditions d'état graves, si elles persistent, conduisent Elastic Beanstalk à faire passer l'environnement au statut Suspendu.

Exemple : Elastic Beanstalk ne parvient pas à accéder au [rôle de service \(p. 926\)](#) de l'environnement.

Exemple : le [groupe Auto Scaling \(p. 547\)](#) créé par Elastic Beanstalk pour l'environnement a été supprimé.

Message : Environment health has transitioned from OK to Severe. Il n'y a aucune instance. La capacité souhaitée du groupe Auto Scaling est définie sur 1.

# Métriques des instances

Des métriques d'instance fournissent des informations sur l'intégrité d'instances dans votre environnement. L'[agent de vérification de l'état Elastic Beanstalk \(p. 840\)](#) s'exécute sur chaque instance. Il rassemble et transmet à Elastic Beanstalk des métriques relatives aux instances. Elastic Beanstalk analyse ensuite ces métriques pour déterminer l'état des instances dans vos environnements.

L'agent de vérification de l'état Elastic Beanstalk sur instance recueille des métriques sur les instances à partir de serveurs web et du système d'exploitation. Pour obtenir des informations sur les serveurs web

sur les plateformes Linux, Elastic Beanstalk lit et analyse les journaux des serveurs web. Sur la plateforme Windows Server, Elastic Beanstalk reçoit directement ces informations du serveur web IIS. Les serveurs web fournissent des informations sur les demandes HTTP entrantes : le nombre de requêtes qui sont entrées, combien ont généré des erreurs et le délai qui a été nécessaire à leur résolution. Le système d'exploitation fournit des informations instantanées sur l'état des ressources des instances : la charge de l'UC et la répartition du temps consacré à chaque type de processus.

L'agent de vérification de l'état recueille des métriques de système d'exploitation et de serveur web et les transmet à Elastic Beanstalk toutes les 10 secondes. Elastic Beanstalk analyse les données et utilise les résultats pour mettre à jour l'état de santé de chaque instance et de l'environnement.

#### Rubriques

- [Métriques de serveur web \(p. 857\)](#)
- [Métriques du système d'exploitation \(p. 858\)](#)
- [Capture des métriques du serveur web dans IIS sous Windows Server \(p. 858\)](#)

## Métriques de serveur web

Sur les plateformes Linux, l'agent de vérification de l'état Elastic Beanstalk lit les métriques de serveur web à partir des journaux générés par le conteneur web ou le serveur qui traite les demandes sur chaque instance de votre environnement. Les plateformes Elastic Beanstalk sont configurées pour générer deux journaux : un au format lisible par l'utilisateur et un au format lisible par la machine. L'agent de vérification de l'état transmet les journaux lisibles par la machine à Elastic Beanstalk toutes les 10 secondes.

Pour de plus amples informations sur le format de journal utilisé par Elastic Beanstalk, veuillez consulter [Format de journal d'intégrité améliorée \(p. 870\)](#).

Sur la plateforme Windows Server, Elastic Beanstalk ajoute un module au pipeline de demandes du serveur web IIS et capture les métriques relatives aux délais des demandes HTTP et aux codes de réponse. Le module envoie ces métriques à l'agent d'état de l'instance à l'aide d'un canal de communication inter-processus (IPC) hautes performances. Pour plus d'informations sur l'implémentation, consultez [Capture des métriques du serveur web dans IIS sous Windows Server \(p. 858\)](#).

#### Métriques de serveur web signalées

##### RequestCount

Nombre de requêtes gérées par le serveur web par seconde au cours des 10 dernières secondes. Affiché comme un  $x$  / sec (demandes par seconde) moyen dans l'interface de ligne de commande EB et sur la [Page « Health \(Santé\) » de l'environnement \(p. 850\)](#).

##### Status2xx, Status3xx, Status4xx, Status5xx

Nombre de requêtes ayant abouti sur chaque type de code de statut au cours des 10 dernières secondes. Par exemple, les demandes ayant abouti renvoient 200 OK, les redirections renvoient 301 et une erreur 404 est renvoyée si l'URL saisie ne correspond à aucune ressource de l'application.

L'interface de ligne de commande (CLI) EB et la [Page « Health \(Santé\) » de l'environnement \(p. 850\)](#) affichent ces métriques sous la forme d'un nombre brut de demandes pour les instances et sous la forme d'un pourcentage des demandes globales pour les environnements.

##### p99 . 9, p99, p95, p90, p85, p75, p50, p10

Moyenne de latence pour le pourcentage de requêtes  $x$  le plus lent au cours des 10 dernières secondes, où  $x$  est la différence entre le nombre et 100. Par exemple, p99 . 1 . 403 indique que les demandes faisant partie des 1 % les plus lentes au cours des 10 dernières secondes avaient une latence moyenne de 1 351 secondes.

## Métriques du système d'exploitation

L'agent de vérification de l'état Elastic Beanstalk rapporte les métriques de système d'exploitation suivantes. Elastic Beanstalk utilise ces métriques pour identifier les instances qui subissent une charge lourde constante. Les métriques diffèrent selon le système d'exploitation.

### Métriques de système d'exploitation signalées (Linux)

#### Running

Le temps qui s'est écoulé depuis le lancement de l'instance.

#### Load 1, Load 5

Charge moyenne au cours des dernières périodes de 1 minute et de 5 minutes. Affiché comme une valeur décimale indiquant le nombre moyen de processus qui s'exécutent pendant cette durée. Si le nombre affiché est plus élevé que le nombre de vCPU (threads) disponibles, alors le reste correspond au nombre moyen de processus qui ont été en attente.

Par exemple, si votre type d'instance a quatre vCPU et que la charge est 4,5, en moyenne 0,5 processus a été en attente au cours de cette période, ce qui équivaut à un processus en attente 50 % du temps.

#### User %, Nice %, System %, Idle %, I/O Wait %

Pourcentage de temps que l'UC a consacré à chaque état au cours des 10 dernières secondes.

### Métriques de système d'exploitation signalées (Windows)

#### Running

Le temps qui s'est écoulé depuis le lancement de l'instance.

#### % User Time, % Privileged Time, % Idle Time

Pourcentage de temps que l'UC a consacré à chaque état au cours des 10 dernières secondes.

## Capture des métriques du serveur web dans IIS sous Windows Server

Sur la plateforme Windows Server, Elastic Beanstalk ajoute un module au pipeline de demandes du serveur web IIS et capture les métriques relatives aux délais des demandes HTTP et aux codes de réponse. Le module envoie ces métriques à l'agent d'état de l'instance à l'aide d'un canal de communication inter-processus (IPC) hautes performances. L'agent de vérification de l'état regroupe ces métriques, les combine avec celles du système d'exploitation et les envoie au service Elastic Beanstalk.

### Détails de l'implémentation

Pour capturer les métriques provenant d'IIS, Elastic Beanstalk implémente une interface [IHttpModule](#) gérée et s'abonne aux événements [BeginRequest](#) et [EndRequest](#). Cela permet au module de signaler la latence des requêtes HTTP et les codes de réponse pour toutes les requêtes web gérées par IIS. Pour ajouter le module au pipeline de demandes d'IIS, Elastic Beanstalk enregistre le module dans la section [`<modules>`](#) du fichier de configuration IIS `%windir%\System32\inetsrv\config\applicationHost.config`.

Le module Elastic Beanstalk dans IIS envoie les métriques de demandes web capturées à l'agent de vérification de l'état sur instance, qui est un service Windows appelé `HealthD`. Pour envoyer ces données,

le module utilise [NetNamedPipeBinding](#), qui fournit une liaison sécurisée et fiable, optimisée pour les communications machine.

## Configuration de règles d'intégrité améliorée pour un environnement

AWS Elastic BeanstalkLe rapport sur l'intégrité améliorée s'appuie sur un ensemble de règles qui déterminent l'état de votre environnement. Certaines de ces règles peuvent ne pas être adaptées à votre application. Voici quelques exemples courants :

- Vous utilisez des outils de test côté client. Dans ce cas, des erreurs fréquentes de client HTTP (4xx) sont attendues.
- Vous utilisez [AWS WAF](#) avec l'équilibrEUR de charge Application Load Balancer de votre environnement pour bloquer le trafic entrant indésirable. Dans ce cas, l'équilibrEUR de charge Application Load Balancer renvoie l'erreur HTTP 403 pour chaque message entrant rejeté.

Par défaut, Elastic Beanstalk inclut toutes les erreurs HTTP 4xx de l'application lors de la détermination de l'état de l'environnement. L'état de santé de votre environnement passe de OK à Warning (Avertissement), Degraded (Dégradé) ou Severe (Grave), en fonction du taux d'erreur. Pour gérer correctement les cas mentionnés dans les exemples précédents, Elastic Beanstalk vous permet de configurer certaines règles d'état améliorées. Vous pouvez choisir d'ignorer les erreurs HTTP 4xx de l'application sur les instances de l'environnement ou d'ignorer les erreurs HTTP 4xx renvoyées par l'équilibrEUR de charge de l'environnement. Cette rubrique décrit comment effectuer ces modifications de configuration.

### Note

Actuellement, il s'agit de la seule personnalisation de règle d'intégrité améliorée disponible. Vous ne pouvez pas configurer l'intégrité améliorée pour ignorer d'autres erreurs HTTP en plus de 4xx.

## Configuration des règles d'état améliorées à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer les règles d'état améliorées dans votre environnement.

Pour configurer la vérification des codes d'état HTTP 4xx à l'aide de la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Sous Personnalisation de la règle de surveillance de l'intégrité, activez ou désactivez les options Ignorer souhaitées.

## Health monitoring rule customization

Configure the HTTP application and load balancer status codes included in determining your environment's health.

Ignore application 4xx

Enabled

Ignore load balancer 4xx

Enabled

6. Choisissez Apply.

## Configuration des règles d'intégrité améliorée à l'aide de l'interface de ligne de commande EB

Vous pouvez utiliser l'interface de ligne de commande EB pour configurer les règles d'état améliorées en enregistrant la configuration de votre environnement en local, en ajoutant une entrée qui configure les règles d'état améliorées, puis en chargeant la configuration dans Elastic Beanstalk. Vous pouvez appliquer la configuration enregistrée à un environnement pendant ou après la création.

Pour configurer la vérification du code d'état HTTP 4xx à l'aide de l'interface de ligne de commande EB et des configurations enregistrées

1. Initialisez votre dossier de projet avec [eb init \(p. 1036\)](#).
2. Créez un environnement en exécutant la commande [eb create \(p. 1040\)](#).
3. Enregistrez un modèle de configuration localement en exécutant la commande eb config save. L'exemple suivant utilise l'option --cfg pour spécifier le nom de la configuration.

```
$ eb config save --cfg 01-base-state
Configuration saved at: ~/project/.elasticbeanstalk/saved_configs/01-base-state.cfg.yml
```

4. Ouvrez le fichier de configuration enregistrée dans un éditeur de texte.
5. Sous OptionSettings > aws:elasticbeanstalk:healthreporting:system:, ajoutez une clé ConfigDocument pour lister chaque règle d'intégrité améliorée à configurer. Le ConfigDocument suivant désactive la vérification des codes d'état HTTP 4xx de l'application, tout en conservant la vérification du code HTTP 4xx de l'équilibreur de charge activé.

```
OptionSettings:
  ...
aws:elasticbeanstalk:healthreporting:system:
  ConfigDocument:
    Rules:
      Environment:
        Application:
          ApplicationRequests4xx:
            Enabled: false
      ELB:
        ELBRequests4xx:
```

```
    Enabled: true
    Version: 1
    SystemType: enhanced
    ...
```

#### Note

Vous pouvez combiner Rules et CloudWatchMetrics dans le même paramètre d'option ConfigDocument. Le paramètre CloudWatchMetrics est décrit dans [Publication de métriques personnalisées Amazon CloudWatch pour un environnement \(p. 862\)](#).

Si vous avez précédemment activé CloudWatchMetrics, le fichier de configuration que vous récupérez à l'aide de la commande eb config save possède déjà une clé ConfigDocument avec une section CloudWatchMetrics. Ne la supprimez pas : ajoutez une section Rules dans la même valeur d'option ConfigDocument.

6. Enregistrez le fichier de configuration et fermez l'éditeur de texte. Dans cet exemple, le fichier de configuration mis à jour est enregistré avec un nom qui est différent (02-cloudwatch-enabled.cfg.yml) de celui du fichier de configuration téléchargé. Cela crée une configuration enregistrée distincte lorsque le fichier est téléchargé. Vous pouvez utiliser le même nom que le fichier téléchargé pour remplacer la configuration existante sans en créer une.
7. Utilisez la commande eb config put pour charger le fichier de configuration mis à jour dans Elastic Beanstalk.

```
$ eb config put 02-cloudwatch-enabled
```

Lorsque vous utilisez les commandes eb config get et put avec des configurations enregistrés, n'incluez pas l'extension de nom de fichier.

8. Appliquez la configuration enregistrée à votre environnement en cours d'exécution.

```
$ eb config --cfg 02-cloudwatch-enabled
```

L'option --cfg spécifie un fichier de configuration nommé qui est appliqué à l'environnement. Vous pouvez enregistrer le fichier de configuration en local ou dans Elastic Beanstalk. Si un fichier de configuration avec le nom spécifié existe dans les deux emplacements, l'interface de ligne de commande EB utilise le fichier local.

## Configuration des règles d'intégrité améliorée à l'aide d'un document de configuration

Le document de configuration pour les règles d'intégrité améliorée est un document JSON qui répertorie les règles à configurer.

L'exemple suivant montre un document de configuration qui désactive la vérification des codes d'état HTTP 4xx de l'application et active la vérification des codes d'état HTTP 4xx de l'équilibreur de charge.

```
{
  "Rules": {
    "Environment": {
      "Application": {
        "ApplicationRequests4xx": {
          "Enabled": false
        }
      },
      "ELB": {
        "ELBRequests4xx": {
```

```
        "Enabled": true
    }
}
},
"Version": 1
}
```

Pour l'AWS CLI, vous transmettez le document sous forme de valeur pour la clé `Value` dans un argument de paramètres d'option, qui est lui-même un objet JSON. Dans ce cas, vous devez utiliser des guillemets d'échappement dans le document intégré. La commande suivante vérifie si les paramètres de configuration sont valides.

```
$ aws elasticbeanstalk validate-configuration-settings --application-name my-app --environment-name my-env --option-settings '[
{
    "Namespace": "aws:elasticbeanstalk:healthreporting:system",
    "OptionName": "ConfigDocument",
    "Value": "{\"Rules\": { \"Environment\": { \"Application\": {
        \"ApplicationRequests4xx\": { \"Enabled\": false } }, \"ELB\": { \"ELBRequests4xx\": {
            \"Enabled\": true } } }, \"Version\": 1 }}"
}
]'
```

Pour un fichier de configuration `.ebextensions` au format YAML, vous pouvez fournir le document JSON en l'état.

```
option_settings:
- namespace: aws:elasticbeanstalk:healthreporting:system
  option_name: ConfigDocument
  value: {
    "Rules": {
      "Environment": {
        "Application": {
          "ApplicationRequests4xx": {
            "Enabled": false
          }
        },
        "ELB": {
          "ELBRequests4xx": {
            "Enabled": true
          }
        }
      }
    },
    "Version": 1
  }
```

## Publication de métriques personnalisées Amazon CloudWatch pour un environnement

Vous pouvez publier les données recueillies par la création de rapports d'état de santé amélioré AWS Elastic Beanstalk sur Amazon CloudWatch en tant que métriques personnalisées. La publication de métriques dans CloudWatch vous permet de surveiller les changements de performances des applications sur la durée et d'identifier les éventuels problèmes en suivant l'évolution de la latence des demandes et de l'utilisation des ressources en fonction de la charge.

En publiant des métriques dans CloudWatch, vous les rendez également disponibles pour une utilisation avec les [graphiques de surveillance \(p. 831\)](#) et les [alarmes \(p. 874\)](#). Une métrique gratuite,

EnvironmentHealth, est activée automatiquement lorsque vous utilisez la création de rapports améliorés sur l'état. Les métriques personnalisées autres que EnvironmentHealth impliquent des [frais CloudWatch](#) standard.

Pour publier des métriques personnalisées CloudWatch pour un environnement, vous devez commencer par activer les rapports améliorés sur l'état dans l'environnement. Pour obtenir des instructions, consultez [Activation des rapports améliorés sur l'état Elastic Beanstalk \(p. 845\)](#).

#### Rubriques

- [Métriques de création de rapports d'intégrité améliorés \(p. 863\)](#)
- [Configuration des métriques CloudWatch à l'aide de la console Elastic Beanstalk \(p. 864\)](#)
- [Configuration des métriques personnalisées CloudWatch à l'aide de l'interface de ligne de commande EB \(p. 865\)](#)
- [Fourniture des documents de configuration des métriques personnalisées \(p. 866\)](#)

## Métriques de création de rapports d'intégrité améliorés

Lorsque vous activez les rapports améliorés sur l'état dans votre environnement, le système de génération de rapports améliorés sur l'état publie automatiquement une [métrique personnalisée CloudWatch](#), EnvironmentHealth. Pour publier des métriques supplémentaires dans CloudWatch, configurez votre environnement avec ces métriques à l'aide de la [console Elastic Beanstalk \(p. 864\)](#), de l'[interface de ligne de commande EB \(p. 865\)](#) ou du fichier [.ebextensions \(p. 658\)](#).

Vous pouvez publier les métriques d'état améliorées suivantes à partir de votre environnement dans CloudWatch.

### Métriques disponibles (toutes les plateformes)

#### EnvironmentHealth

Environnement uniquement. Il s'agit de la seule métrique CloudWatch publiée par le système de génération de rapports améliorés sur l'état, sauf si vous configurez des métriques supplémentaires. L'état de l'environnement est représentée par un des sept [statuts \(p. 854\)](#). Dans la console CloudWatch, ces statuts sont mappés aux valeurs suivantes :

- 0 – OK
- 1 – Info
- 5 – Inconnu
- 10 – Pas de données
- 15 – Avertissement
- 20 – Dégradé
- 25 – Grave

`InstancesSevere`, `InstancesDegraded`, `InstancesWarning`, `InstancesInfo`, `InstancesOk`, `InstancesPending`, `InstancesUnknown`, `InstancesNoData`

Environnement uniquement. Ces métriques indiquent le nombre d'instances dans les environnements avec chaque état de santé. `InstancesNoData` indique le nombre d'instances pour lesquelles aucune donnée ne sera reçue.

`ApplicationRequestsTotal`, `ApplicationRequests5xx`, `ApplicationRequests4xx`, `ApplicationRequests3xx`, `ApplicationRequests2xx`

Instance et environnement. Indique le nombre total de requêtes terminées par l'instance ou l'environnement et le nombre de requêtes ayant abouti avec chaque catégorie de code d'état.

`ApplicationLatencyP10, ApplicationLatencyP50, ApplicationLatencyP75,  
ApplicationLatencyP85, ApplicationLatencyP90, ApplicationLatencyP95,  
ApplicationLatencyP99, ApplicationLatencyP99.9`

Instance et environnement. Indique la quantité moyenne de temps, en secondes, nécessaire pour terminer le pourcentage x le plus rapide de requêtes.

#### `InstanceHealth`

Instance uniquement. Indique l'état d'intégrité actuel de l'instance. L'état d'instance est représentée par un statut (sur sept [status \(p. 854\)](#) au total). Dans la console CloudWatch, ces statuts sont mappés aux valeurs suivantes :

- 0 – OK
- 1 – Info
- 5 – Inconnu
- 10 – Pas de données
- 15 – Avertissement
- 20 – Dégradé
- 25 – Grave

#### Métriques disponibles (Linux)

`CPUIrq, CPUIdle, CPUUser, CPUSystem, CPUSoftirq, CPUIowait, CPUNice`

Instance uniquement. Indique le pourcentage de temps que l'UC a passé dans chaque état au cours de la dernière minute.

#### `LoadAverage1min`

Instance uniquement. La charge d'UC moyenne de l'instance au cours de la dernière minute.

#### `RootFilesystemUtil`

Instance uniquement. Indique le pourcentage d'espace disque en cours d'utilisation.

#### Métriques disponibles (Windows)

`CPUIdle, CPUUser, CPUPrileged`

Instance uniquement. Indique le pourcentage de temps que l'UC a passé dans chaque état au cours de la dernière minute.

## Configuration des métriques CloudWatch à l'aide de la console Elastic Beanstalk

Vous pouvez utiliser la console Elastic Beanstalk pour configurer votre environnement afin de publier des métriques d'état améliorées dans CloudWatch et les rendre disponibles pour une utilisation avec des graphiques de surveillance et des alarmes.

Pour configurer des métriques personnalisées CloudWatch dans la console Elastic Beanstalk

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Sous Health reporting (Rapport sur l'état de santé), sélectionnez les métriques d'instance et d'environnement que vous souhaitez publier dans CloudWatch. Pour sélectionner plusieurs métriques, appuyez sur la touche Ctrl tout en choisissant.
6. Choisissez Apply.

L'activation des métriques personnalisées CloudWatch les ajoute à la liste des métriques disponibles sur la page [Monitoring \(Surveillance\) \(p. 830\)](#).

## Configuration des métriques personnalisées CloudWatch à l'aide de l'interface de ligne de commande EB

Vous pouvez utiliser l'interface de ligne de commande EB pour configurer des métriques personnalisées en enregistrant la configuration de votre environnement en local, en ajoutant une entrée qui définit les métriques à publier, puis en chargeant la configuration dans Elastic Beanstalk. Vous pouvez appliquer la configuration enregistrée à un environnement pendant ou après la création.

Pour configurer des métriques personnalisées CloudWatch avec l'interface de ligne de commande EB et les configurations enregistrées

1. Initialisez votre dossier de projet avec [eb init \(p. 1036\)](#).
2. Créez un environnement en exécutant la commande [eb create \(p. 1040\)](#).
3. Enregistrez un modèle de configuration localement en exécutant la commande eb config save. L'exemple suivant utilise l'option --cfg pour spécifier le nom de la configuration.

```
$ eb config save --cfg 01-base-state
Configuration saved at: ~/project/.elasticbeanstalk/saved_configs/01-base-state.cfg.yml
```

4. Ouvrez le fichier de configuration enregistrée dans un éditeur de texte.
5. Sous OptionSettings > aws:elasticbeanstalk:healthreporting:system:, ajoutez une clé ConfigDocument pour activer chacune des métriques CloudWatch de votre choix. Par exemple, le ConfigDocument suivant publie des métriques ApplicationRequests5xx et ApplicationRequests4xx au niveau de l'environnement et des métriques ApplicationRequestsTotal au niveau de l'instance.

```
OptionSettings:
  ...
  aws:elasticbeanstalk:healthreporting:system:
    ConfigDocument:
      CloudWatchMetrics:
        Environment:
          ApplicationRequests5xx: 60
          ApplicationRequests4xx: 60
        Instance:
          ApplicationRequestsTotal: 60
        Version: 1
      SystemType: enhanced
  ...
```

Dans l'exemple, 60 indique le nombre de secondes entre les mesures. C'est la seule valeur actuellement prise en charge.

Note

Vous pouvez combiner CloudWatchMetrics et Rules dans le même paramètre d'option ConfigDocument. Le paramètre Rules est décrit dans [Configuration de règles d'intégrité améliorée pour un environnement \(p. 859\)](#).

Si vous avez précédemment utilisé Rules pour configurer les règles d'intégrité améliorée, le fichier de configuration que vous récupérez à l'aide de la commande eb config save possède déjà une clé ConfigDocument avec une section Rules. Ne la supprimez pas : ajoutez une section CloudWatchMetrics dans la même valeur d'option ConfigDocument.

6. Enregistrez le fichier de configuration et fermez l'éditeur de texte. Dans cet exemple, le fichier de configuration mis à jour est enregistré avec un nom qui est différent (02-cloudwatch-enabled.cfg.yml) de celui du fichier de configuration téléchargé. Cela crée une configuration enregistrée distincte lorsque le fichier est téléchargé. Vous pouvez utiliser le même nom que le fichier téléchargé pour remplacer la configuration existante sans en créer une.
7. Utilisez la commande eb config put pour charger le fichier de configuration mis à jour dans Elastic Beanstalk.

```
$ eb config put 02-cloudwatch-enabled
```

Lorsque vous utilisez les commandes eb config get et put avec des configurations enregistrées, n'incluez pas l'extension de fichier.

8. Appliquez la configuration enregistrée à votre environnement en cours d'exécution.

```
$ eb config --cfg 02-cloudwatch-enabled
```

L'option --cfg spécifie un fichier de configuration nommé qui est appliqué à l'environnement. Vous pouvez enregistrer le fichier de configuration en local ou dans Elastic Beanstalk. Si un fichier de configuration avec le nom spécifié existe dans les deux emplacements, l'interface de ligne de commande EB utilise le fichier local.

## Fourniture des documents de configuration des métriques personnalisées

Le document de configuration pour les métriques personnalisées Amazon CloudWatch est un document JSON qui répertorie les métriques à publier aux niveaux de l'instance et de l'environnement. L'exemple suivant illustre un document de configuration qui active toutes les métriques personnalisées disponibles.

```
{
  "CloudWatchMetrics": {
    "Environment": {
      "ApplicationLatencyP99.9": 60,
      "InstancesSevere": 60,
      "ApplicationLatencyP90": 60,
      "ApplicationLatencyP99": 60,
      "ApplicationLatencyP95": 60,
      "InstancesUnknown": 60,
      "ApplicationLatencyP85": 60,
      "InstancesInfo": 60,
      "ApplicationRequests2xx": 60,
      "InstancesDegrade": 60,
      "InstancesWarning": 60,
```

```

    "ApplicationLatencyP50": 60,
    "ApplicationRequestsTotal": 60,
    "InstancesNoData": 60,
    "InstancesPending": 60,
    "ApplicationLatencyP10": 60,
    "ApplicationRequests5xx": 60,
    "ApplicationLatencyP75": 60,
    "InstancesOk": 60,
    "ApplicationRequests3xx": 60,
    "ApplicationRequests4xx": 60
  },
  "Instance": {
    "ApplicationLatencyP99.9": 60,
    "ApplicationLatencyP90": 60,
    "ApplicationLatencyP99": 60,
    "ApplicationLatencyP95": 60,
    "ApplicationLatencyP85": 60,
    "CPUUser": 60,
    "ApplicationRequests2xx": 60,
    "CPUIdle": 60,
    "ApplicationLatencyP50": 60,
    "ApplicationRequestsTotal": 60,
    "RootFilesystemUtil": 60,
    "LoadAverage1min": 60,
    "CPUIRQ": 60,
    "CPUNice": 60,
    "CPUWait": 60,
    "ApplicationLatencyP10": 60,
    "LoadAverage5min": 60,
    "ApplicationRequests5xx": 60,
    "ApplicationLatencyP75": 60,
    "CPUSystem": 60,
    "ApplicationRequests3xx": 60,
    "ApplicationRequests4xx": 60,
    "InstanceHealth": 60,
    "CPUsoftirq": 60
  }
},
"Version": 1
}

```

Pour l'AWS CLI, vous transmettez le document sous forme de valeur pour la clé `Value` dans un argument de paramètres d'option, qui est lui-même un objet JSON. Dans ce cas, vous devez utiliser des guillemets d'échappement dans le document intégré.

```
$ aws elasticbeanstalk validate-configuration-settings --application-name my-app --environment-name my-env --option-settings '[{"Namespace": "aws:elasticbeanstalk:healthreporting:system", "OptionName": "ConfigDocument", "Value": "{\"CloudWatchMetrics\": {\"Environment\": {\"ApplicationLatencyP99.9\": 60, \"InstancesSevere\": 60, \"ApplicationLatencyP90\": 60, \"ApplicationLatencyP95\": 60, \"InstancesUnknown\": 60, \"ApplicationLatencyP85\": 60, \"InstancesInfo\": 60, \"ApplicationRequests2xx\": 60, \"InstancesDegraded\": 60, \"InstancesWarning\": 60, \"ApplicationLatencyP50\": 60, \"ApplicationRequestsTotal\": 60, \"InstancesNoData\": 60, \"InstancesPending\": 60, \"ApplicationLatencyP10\": 60, \"ApplicationRequests5xx\": 60, \"ApplicationLatencyP75\": 60, \"InstancesOk\": 60, \"ApplicationRequests3xx\": 60, \"ApplicationRequests4xx\": 60}, \"Instance\": {\"ApplicationLatencyP99.9\": 60, \"ApplicationLatencyP90\": 60, \"ApplicationLatencyP99\": 60, \"ApplicationLatencyP85\": 60, \"CPUUser\": 60, \"ApplicationRequests2xx\": 60, \"CPUIdle\": 60, \"ApplicationLatencyP50\": 60, \"ApplicationRequestsTotal\": 60, \"RootFilesystemUtil\": 60, \"LoadAverage1min\": 60, \"CPUIRQ\": 60, \"CPUNice\": 60, \"CPUWait\": 60, \"ApplicationLatencyP10\": 60, \"LoadAverage5min\": 60, \"ApplicationRequests5xx\": 60, \"ApplicationLatencyP75\": 60}}}}']"
```

```
    60, \"CPUSystem\": 60, \"ApplicationRequests3xx\": 60, \"ApplicationRequests4xx\": 60,  
    \"InstanceHealth\": 60, \"CPUSoftirq\": 60}}, \"Version\": 1}  
  }  
]
```

Pour un fichier de configuration `.ebextensions` au format YAML, vous pouvez fournir le document JSON en l'état.

```
option_settings:  
  - namespace: aws:elasticbeanstalk:healthreporting:system  
    option_name: ConfigDocument  
    value: {  
      "CloudWatchMetrics": {  
        "Environment": {  
          "ApplicationLatencyP99.9": 60,  
          "InstancesSevere": 60,  
          "ApplicationLatencyP90": 60,  
          "ApplicationLatencyP99": 60,  
          "ApplicationLatencyP95": 60,  
          "InstancesUnknown": 60,  
          "ApplicationLatencyP85": 60,  
          "InstancesInfo": 60,  
          "ApplicationRequests2xx": 60,  
          "InstancesDegraded": 60,  
          "InstancesWarning": 60,  
          "ApplicationLatencyP50": 60,  
          "ApplicationRequestsTotal": 60,  
          "InstancesNoData": 60,  
          "InstancesPending": 60,  
          "ApplicationLatencyP10": 60,  
          "ApplicationRequests5xx": 60,  
          "ApplicationLatencyP75": 60,  
          "InstancesOk": 60,  
          "ApplicationRequests3xx": 60,  
          "ApplicationRequests4xx": 60  
        },  
        "Instance": {  
          "ApplicationLatencyP99.9": 60,  
          "ApplicationLatencyP90": 60,  
          "ApplicationLatencyP99": 60,  
          "ApplicationLatencyP95": 60,  
          "ApplicationLatencyP85": 60,  
          "CPUUser": 60,  
          "ApplicationRequests2xx": 60,  
          "CPUIidle": 60,  
          "ApplicationLatencyP50": 60,  
          "ApplicationRequestsTotal": 60,  
          "RootFilesystemUtil": 60,  
          "LoadAverage1min": 60,  
          "CPUIRQ": 60,  
          "CPUNice": 60,  
          "CPUIowait": 60,  
          "ApplicationLatencyP10": 60,  
          "LoadAverage5min": 60,  
          "ApplicationRequests5xx": 60,  
          "ApplicationLatencyP75": 60,  
          "CPUSystem": 60,  
          "ApplicationRequests3xx": 60,  
          "ApplicationRequests4xx": 60,  
          "InstanceHealth": 60,  
          "CPUSoftirq": 60  
        }  
      },  
      "Version": 1  
    }  
  }
```

}

## Utilisation des rapports améliorés sur l'état à l'aide de l'API Elastic Beanstalk

Les rapports améliorés sur l'état d'AWS Elastic Beanstalk présentent des exigences en matière de rôles et de pile de solutions. Par conséquent, vous devez mettre à jour les scripts et le code que vous avez utilisés avant la publication des rapports améliorés sur l'état avant de pouvoir les utiliser. Pour assurer la rétrocompatibilité, les rapports améliorés sur l'état ne sont pas activés par défaut lorsque vous créez un environnement à l'aide de l'API Elastic Beanstalk.

Pour configurer les rapports améliorés sur l'état, définissez le rôle de service, le profil d'instance et les options de configuration Amazon CloudWatch pour votre environnement. Vous pouvez le faire de trois façons : en définissant les options de configuration dans le dossier `.ebextensions`, avec des configurations enregistrées ou en les configurant directement dans le paramètre `create-environment` de l'appel `option-settings`.

Afin d'utiliser l'API, les SDK ou l'interface de ligne de commande (CLI) AWS pour créer un environnement prenant en charge les rapports améliorés sur l'état, procédez comme suit :

- Créez un rôle de service et un profil d'instance avec les [autorisations \(p. 21\)](#) appropriées.
- Créez un nouvel environnement avec une nouvelle [version de plateforme \(p. 31\)](#)
- Définissez les [options de configuration \(p. 658\)](#) du type de système d'état, du profil d'instance et du rôle de service.

Utilisez les options de configuration suivantes dans les espaces de noms `aws:elasticbeanstalk:healthreporting:system`, `aws:autoscaling:launchconfiguration` et `aws:elasticbeanstalk:environment` afin de configurer votre environnement pour les rapports améliorés sur l'état.

### Options de configuration des rapports améliorés sur l'état

`SystemType`

Espace de nom : `aws:elasticbeanstalk:healthreporting:system`

Pour activer les rapports améliorés sur l'état, définissez l'option sur **enhanced**.

`IamInstanceProfile`

Espace de nom : `aws:autoscaling:launchconfiguration`

Choisissez le nom d'un profil d'instance configuré pour être utilisé avec Elastic Beanstalk.

`ServiceRole`

Espace de nom : `aws:elasticbeanstalk:environment`

Choisissez le nom d'un rôle de service configuré pour être utilisé avec Elastic Beanstalk.

`ConfigDocument` (facultatif)

Espace de nom : `aws:elasticbeanstalk:healthreporting:system`

Document JSON qui définit les métriques de l'instance et de l'environnement à publier dans CloudWatch.  
Exemples :

```
{  
    "CloudWatchMetrics":  
    {  
        "Environment":  
        {  
            "ApplicationLatencyP99.9":60,  
            "InstancesSevere":60  
        }  
        "Instance":  
        {  
            "ApplicationLatencyP85":60,  
            "CPUUser": 60  
        }  
        "Version":1  
    }  
}
```

#### Note

Les documents de configuration peuvent exiger une mise en forme spéciale, comme des guillemets d'échappement, en fonction de la façon dont vous les fournissez à Elastic Beanstalk. Pour obtenir des exemples, consultez la section [Fourniture des documents de configuration des métriques personnalisées \(p. 866\)](#).

## Format de journal d'intégrité améliorée

AWS Elastic Beanstalk Les plateformes utilisent un format de journal de serveur web personnalisé pour transmettre efficacement des informations sur les demandes HTTP pour le système de rapports améliorés sur l'état de santé. Le système analyse les journaux, identifie les problèmes et définit en conséquence l'état de santé de l'instance et de l'environnement. Si vous désactivez le proxy de serveur web dans votre environnement et que vous traitez les demandes directement depuis le conteneur web, vous pouvez toujours utiliser pleinement les rapports améliorés sur l'état en configurant votre serveur de sorte à générer des journaux à l'emplacement et au format utilisés par l'[agent de vérification de l'état Elastic Beanstalk \(p. 840\)](#).

#### Note

Les informations de cette page concernent uniquement les plateformes Linux. Sur la plateforme Windows Server, Elastic Beanstalk reçoit les informations sur les demandes HTTP directement à partir du serveur web IIS. Pour plus d'informations, consultez [Capture des métriques du serveur web dans IIS sous Windows Server \(p. 858\)](#).

## Configuration de journal de serveur web

Les plateformes Elastic Beanstalk sont configurées de sorte à générer deux journaux contenant des informations sur les demandes HTTP. La première est au format détaillé et fournit des informations complètes sur la demande, y compris les informations de l'agent utilisateur du demandeur et un horodatage contrôlable de visu.

/var/log/nginx/access.log

L'exemple suivant provient d'un proxy nginx exécuté dans un environnement de serveur web Ruby, mais le format est similaire pour Apache.

```
172.31.24.3 - - [23/Jul/2015:00:21:20 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0  
(x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"  
"177.72.242.17"
```

```
172.31.24.3 -- [23/Jul/2015:00:21:21 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0  
(x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"  
"177.72.242.17"  
172.31.24.3 -- [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0  
(x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"  
"177.72.242.17"  
172.31.24.3 -- [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0  
(x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"  
"177.72.242.17"  
172.31.24.3 -- [23/Jul/2015:00:21:22 +0000] "GET / HTTP/1.1" 200 11 "-" "curl/7.22.0  
(x86_64-pc-linux-gnu) libcurl/7.22.0 OpenSSL/1.0.1 zlib/1.2.3.4 libidn/1.23 librtmp/2.3"  
"177.72.242.17"
```

Le deuxième journal est au format court. Il comporte des informations pertinentes uniquement pour la création de rapports d'intégrité améliorée. Ce journal est sorti vers un sous-dossier nommé `healthd` et tourne chaque heure. Les anciens journaux sont supprimés immédiatement après la rotation.

/var/log/nginx/healthd/application.log.2015-07-23-00

L'exemple suivant montre un journal au format lisible par la machine.

```
1437609879.311"/"200"0.083"0.083"177.72.242.17  
1437609879.874"/"200"0.347"0.347"177.72.242.17  
1437609880.006"/bad/path"404"0.001"0.001"177.72.242.17  
1437609880.058"/"200"0.530"0.530"177.72.242.17  
1437609880.928"/bad/path"404"0.001"0.001"177.72.242.17
```

Le format des journaux d'intégrité améliorée inclut les informations suivantes :

- Le moment de la demande, en heure Unix
- Le chemin d'accès de la demande
- Le code de statut HTTP pour le résultat
- La durée des demandes
- Le temps en amont
- L'en-tête HTTP `X-Forwarded-For`

Pour les proxys nginx, les heures sont indiquées en secondes à virgule flottante, avec trois décimales. Pour Apache, les millisecondes entières sont utilisées.

#### Note

Si un avertissement similaire au suivant s'affiche dans un fichier journal, où `DATE-TIME` correspond à une date et une heure, et que vous utilisez un proxy personnalisé, comme dans un environnement Docker multi-conteneurs, vous devez utiliser un fichier `.ebextension` pour configurer votre environnement afin que `healthd` puisse lire vos fichiers journaux :

```
W, [DATE-TIME #1922] WARN -- : log file "/var/log/nginx/healthd/  
application.log.DATE-TIME" does not exist
```

Vous pouvez commencer par le fichier `.ebextension` dans l'[exemple Docker multi-conteneurs](#).

/etc/nginx/conf.d/webapp\_healthd.conf

L'exemple suivant montre la configuration de journal pour nginx avec le format de journal `healthd` mis en évidence.

```
upstream my_app {
```

```
    server unix:///var/run/puma/my_app.sock;
}

log_format healthd '$msec"$uri"'
                  '$status"$request_time"$upstream_response_time"'
                  '$http_x_forwarded_for';

server {
    listen 80;
    server_name _ localhost; # need to listen to localhost for worker tier

    if ($time_iso8601 ~ "^(\d{4})-(\d{2})-(\d{2})T(\d{2})") {
        set $year $1;
        set $month $2;
        set $day $3;
        set $hour $4;
    }

    access_log /var/log/nginx/access.log main;
    access_log /var/log/nginx/healthd/application.log.$year-$month-$day-$hour healthd;

    location / {
        proxy_pass http://my_app; # match the name of upstream directive which is defined above
        proxy_set_header Host $host;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }

    location /assets {
        alias /var/app/current/public/assets;
        gzip_static on;
        gzip on;
        expires max;
        add_header Cache-Control public;
    }

    location /public {
        alias /var/app/current/public;
        gzip_static on;
        gzip on;
        expires max;
        add_header Cache-Control public;
    }
}
```

/etc/httpd/conf.d/healthd.conf

L'exemple suivant montre la configuration des journaux pour Apache.

```
LogFormat "%{$_t}\%U\%S\%D\%D\%{X-Forwarded-For}i" healthd
CustomLog "|/usr/sbin/rotatelogs /var/log/httpd/healthd/application.log.%Y-%m-%d-%H 3600"
           healthd
```

## Génération de journaux pour la création de rapports d'intégrité améliorée

Pour fournir des journaux à l'agent d'intégrité, vous devez procéder comme suit :

- Sortir des journaux dans le bon format, comme illustré dans la section précédente
- Sortir des journaux dans /var/log/nginx/healthd/
- Nommer des journaux à l'aide du format suivant : application.log.\$year-\$month-\$day-\$hour
- Effectuer une rotation des journaux une fois par heure

- Ne pas tronquer de journaux

## Notifications et dépannage

Cette page répertorie des exemples de messages de cause pour des problèmes courants et des liens pour obtenir plus d'informations. Les messages de cause s'affichent sur la page de [présentation de l'environnement \(p. 830\)](#) de la console Elastic Beanstalk et sont enregistrés dans des événements ([p. 879](#)) lorsque des problèmes d'état persistent entre plusieurs vérifications.

### Deployments

Elastic Beanstalk surveille la cohérence de votre environnement à la suite de déploiements. En cas de défaillance d'un déploiement propagé, la version de votre application s'exécutant sur les instances de votre environnement peut varier. Cela peut se produire si un déploiement réussit sur un ou plusieurs lots mais échoue avant que tous les lots aient abouti.

Version d'application incorrecte trouvée sur 2 instances sur. Version attendue « v1 » (déploiement 1).

Version d'application incorrecte sur des instances d'environnement. Version attendue « v1 » (déploiement 1).

La version d'application attendue ne s'exécute pas sur tout ou partie des instances dans un environnement.

Version d'application incorrecte « v2 » (déploiement 2). Version attendue « v1 » (déploiement 1).

L'application déployée sur une instance diffère de la version attendue. En cas de défaillance d'un déploiement, la version attendue revient à la version du dernier déploiement ayant abouti. Dans l'exemple ci-dessus, le premier déploiement (version « v1 ») a abouti, mais le deuxième déploiement (version « v2 ») a échoué. Toutes les instances exécutant « v2 » sont considérées comme défectueuses.

Pour résoudre ce problème, démarrez un autre déploiement. Vous pouvez [redéployer une version précédente \(p. 476\)](#) dont vous savez qu'elle fonctionne, ou configurer votre environnement pour [ignorer les vérifications de l'état \(p. 480\)](#) au cours du déploiement et redéployer la nouvelle version pour forcer le déploiement à aboutir.

Vous pouvez également identifier et résilier les instances qui exécutent la mauvaise version d'application. Elastic Beanstalk lance des instances avec la version appropriée pour remplacer toutes les instances que vous résiliez. Utilisez la [commande d'état de l'interface de ligne de commande EB \(p. 1053\)](#) pour identifier les instances qui exécutent la mauvaise version d'application.

### Serveur d'application

15 % des demandes signalent une erreur avec HTTP 4xx

20 % des demandes à ELB signalent une erreur avec HTTP 4xx.

Un pourcentage élevé de demandes HTTP à une instance ou un environnement échouent avec des erreurs 4xx.

Un code de statut de série 400 indique que l'utilisateur a soumis une demande erronée, telle que la demande d'une page qui n'existe pas (404 : Fichier introuvable) ou à laquelle l'utilisateur n'a pas accès à (403 Interdit). Un petit nombre d'erreurs 404 n'est pas rare, mais un grand nombre pourrait signifier qu'il y a des liens internes ou externes vers des pages indisponibles. Ces problèmes peuvent être résolus en réparant des liens internes erronés et en ajoutant des redirections pour des liens externes erronés.

5 % des demandes échouent avec HTTP 5xx

3 % des demandes à ELB échouent avec HTTP 5xx.

Un pourcentage élevé de demandes HTTP à une instance ou un environnement échoue avec les codes de statut de série 500.

Un code de statut de série 500 indique que le serveur d'applications a rencontré une erreur interne. Ces problèmes indiquent qu'il y a une erreur dans le code de votre application et ils doivent être identifiés et corrigés rapidement.

95 % de l'UC est en cours d'utilisation

Sur une instance, l'agent d'état rapporte un pourcentage extrêmement élevé d'utilisation de l'UC et définit l'état de l'instance sur Avertissement ou Dégradé.

Mettez à l'échelle votre environnement pour réduire la charge des instances.

## Instance de travail

20 messages en attente dans la file d'attente (il y a 25 secondes)

Des demandes sont ajoutées à la file d'attente de votre environnement de travail plus vite qu'elles ne peuvent être traitées. Mettez à l'échelle votre environnement pour accroître la capacité.

5 messages dans la file d'attente de lettres mortes (il y a 15 secondes)

Des demandes de travail échouent régulièrement et sont ajoutées à la [the section called “Files d'attente de lettres mortes” \(p. 525\)](#). Vérifiez les demandes dans la file d'attente de lettres mortes pour voir pourquoi elles échouent.

## Autres ressources

4 instances actives est inférieur à la taille minimale 5 du groupe Auto Scaling

Le nombre d'instances s'exécutant dans votre environnement est inférieur au nombre minimal configuré pour le groupe Auto Scaling.

Des notifications du groupe Auto Scaling (nom du groupe) ont été supprimées ou modifiées

Les notifications configurées pour votre groupe Auto Scaling ont été modifiées en dehors d'Elastic Beanstalk.

# Gestion des alarmes

Vous pouvez créer des alarmes pour les métriques que vous surveillez en utilisant la console Elastic Beanstalk. Les alarmes vous aident à surveiller les changements apportés à votre environnement AWS Elastic Beanstalk afin que vous puissiez facilement identifier et atténuer les problèmes avant qu'ils ne surviennent. Par exemple, vous pouvez définir une alarme qui vous informe lorsque l'utilisation de l'UC dans un environnement dépasse un certain seuil, veiller à ce que être notifié avant qu'un problème potentiel se produise. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon CloudWatch \(p. 894\)](#).

### Note

Elastic Beanstalk utilise CloudWatch pour la surveillance et les alarmes, ce qui signifie que les coûts CloudWatch sont appliqués à votre compte AWS pour toutes les alarmes que vous utilisez.

Pour plus d'informations sur la surveillance des métriques spécifiques, consultez [Création de rapports d'intégrité de base \(p. 834\)](#).

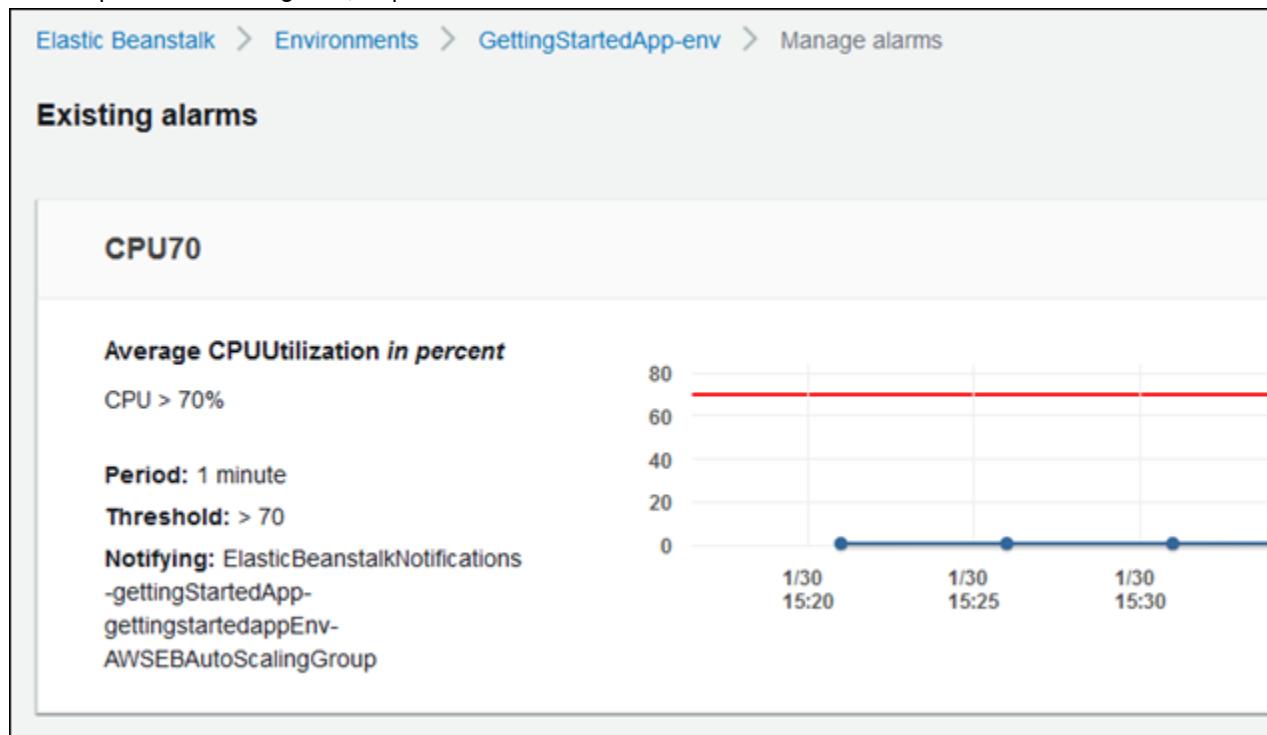
Pour vérifier l'état de vos alarmes

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, cliquez sur Alarms.



La page affiche une liste d'alarmes existantes. Si aucune alarme n'est en état d'alarme, elles sont indiquées par (avertissement).

4. Pour filtrer les alarmes, choisissez le menu déroulant, puis sélectionnez un filtre.
5. Pour modifier ou supprimer une alarme, choisissez (modifier) ou (supprimer), respectivement.

Pour créer une alarme

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Surveillance.

4. Recherchez la métrique pour laquelle vous souhaitez créer une alarme, puis choisissez (alarme). La page Add alarm (Ajouter une alarme) s'affiche.

Elastic Beanstalk > Environments > GettingStartedApp-env > Add alarm

### Add Alarm

**Average CPUUtilization in percent**

Name:

Name should be less than 238 characters in length and can only contain numbers and letters

Description:

Optional.

Period: 1 minute

Threshold: Average CPUUtilization

Change state after:

Notify:

A new SNS topic...

Topic name: ElasticBeanstalkNotifications-gettingStartedApp-gettingsta

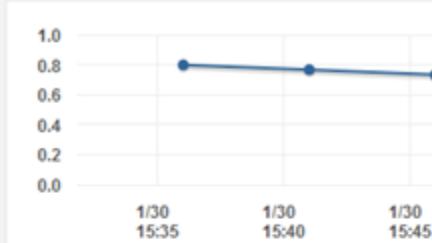
E-mail address:

Notify when state changes to:

OK  
 Alarm  
 Insufficient data

Other Alarms For This Metric

CPU70



5. Entrez les détails relatifs à l'alarme :
- Nom : un nom pour cette alarme.
  - Description (facultatif) : une brève description de ce qu'est cette alarme.
  - Période : l'intervalle de temps entre les lectures.

- Seuil : décrit le comportement et la valeur que la métrique doit dépasser afin de déclencher une alarme.
  - Changer l'état après : le délai après qu'un seuil a été dépassé qui déclenche une modification de l'état de l'alarme.
  - Notifier : rubrique Amazon SNS qui est notifiée lorsqu'une alarme change d'état.
  - Notifier quand l'état passe à :
    - OK : La métrique se trouve dans le seuil défini.
    - Alarme : la métrique a dépassé le seuil défini.
    - Données insuffisantes : l'alarme vient de démarrer, la métrique n'est pas disponible, ou la quantité de données n'est pas suffisante pour permettre à la métrique de déterminer le statut de l'alarme.
6. Choisissez Add (Ajouter). L'état de l'environnement passe au gris pendant la mise à jour de l'environnement. Vous pouvez afficher l'alarme que vous avez créée en choisissant Alarms (Alarmes) dans le volet de navigation.

## Affichage de l'historique des modifications d'un environnement Elastic Beanstalk

Vous pouvez utiliser la console de gestion AWS pour afficher l'historique des modifications de configuration apportées à vos environnements Elastic Beanstalk. Elastic Beanstalk récupère votre historique des modifications à partir d'événements enregistrés dans [AWS CloudTrail](#) et les affiche dans une liste que vous pouvez facilement parcourir et filtrer.

Le panneau Change History (Historique des modifications) affiche les informations suivantes concernant les modifications apportées à vos environnements :

- la date et l'heure auxquelles une modification a été apportée ;
- l'utilisateur IAM responsable d'une modification ;
- l'outil source (l'interface de ligne de commande Elastic Beanstalk (CLI EB) ou la console) utilisé pour effectuer la modification ;
- le paramètre de configuration et les nouvelles valeurs qui ont été définies.

Les données sensibles faisant partie de la modification, telles que les noms des utilisateurs de base de données concernés par la modification, n'apparaissent pas dans le panneau.

Pour afficher l'historique des modifications

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, sélectionnez Change history (Historique des modifications).

The screenshot shows the AWS Elastic Beanstalk interface. On the left, a sidebar titled 'Elastic Beanstalk' lists 'Environments', 'Applications', and 'Change history'. The 'Change history' option is selected and highlighted in orange. Below this, under 'Recent environments', there are three entries: 'GettingStartedApp-env', 'GettingStartedApp-2', and 'flask-docker-env'. To the right, the main content area is titled 'Change history' and displays a table of recent changes. The table has two columns: 'Date' and 'IAM user'. There are ten rows of data, each representing a change made on a specific date and signed by a specific IAM user. The dates range from 2020-10-09T18:55:42Z to 2020-10-16T02:14:15Z.

Date	IAM user
2020-10-16T02:14:15Z	AIDACKCEVSQ6C2EXAMPLE
2020-10-16T02:10:59Z	AIDACKCEVSQ6C2EXAMPLE2
2020-10-16T02:02:50Z	AIDACKCEVSQ6C2EXAMPLE
2020-10-09T21:20:07Z	AIDACKCEVSQ6C2EXAMPLE2
2020-10-09T21:17:01Z	AIDACKCEVSQ6C2EXAMPLE3
2020-10-09T19:05:21Z	AIDACKCEVSQ6C2EXAMPLE3
2020-10-09T19:03:04Z	AIDACKCEVSQ6C2EXAMPLE3
2020-10-09T19:00:04Z	AIDACKCEVSQ6C2EXAMPLE3
2020-10-09T18:55:42Z	AIDACKCEVSQ6C2EXAMPLE3

La page Change History (Historique des modifications) affiche une liste des modifications de configuration apportées à vos environnements Elastic Beanstalk. Vous pouvez parcourir la liste en sélectionnant < (précédent) ou > (suivant) ou en sélectionnant un numéro de page spécifique. Sous la colonne Configuration changes (Modifications de configuration), sélectionnez l'icône flèche pour basculer entre le développement et la réduction de la liste des modifications sous l'en-tête Changes

made (Modifications apportées). Utilisez la barre de recherche pour filtrer vos résultats à partir de la liste de l'historique des modifications. Vous pouvez saisir n'importe quelle chaîne pour affiner la liste des modifications qui s'affichent.

Remarques sur le filtrage des résultats affichés :

- Le filtre de recherche n'est pas sensible à la casse.
- Vous pouvez filtrer les modifications affichées en fonction des informations figurant dans la colonne Configuration changes (Modifications de configuration), même si elles ne sont pas visibles parce qu'elles sont réduites dans Changes made (Modifications apportées).
- Vous ne pouvez filtrer que les résultats affichés. Toutefois, le filtre ne change pas même si vous sélectionnez le bouton pour accéder à une autre page et afficher plus de résultats. Vos résultats filtrés s'ajoutent également au jeu de résultats de la page suivante.

Les exemples suivants montrent comment filtrer les données affichées sur l'écran précédent :

- Saisissez **GettingStartedApp-env** dans la zone de recherche pour affiner les résultats et inclure uniquement les modifications apportées à l'environnement nommé GettingStartedApp-env.
- Saisissez **example3** dans la zone de recherche pour affiner les résultats et inclure uniquement les modifications apportées par les utilisateurs IAM dont le nom d'utilisateur contient la chaîne exemple3.
- Saisissez **2020-10** dans la zone de recherche pour affiner les résultats et inclure uniquement les modifications apportées au cours du mois d'octobre 2020. Définissez la valeur de recherche sur **2020-10-16** pour filtrer davantage les résultats affichés et inclure uniquement les modifications effectuées le jour du 16 octobre 2020.
- Saisissez **proxy:staticfiles** dans la zone de recherche pour affiner les résultats et inclure uniquement les modifications apportées à l'espace de nom nommé `aws:elasticbeanstalk:environment:proxy:staticfiles`. Les lignes affichées sont le résultat du filtre. Cela est vrai même pour les résultats qui sont réduits sous Changes made (Modifications apportées).

## Affichage du flux d'événements d'un environnement Elastic Beanstalk

Vous pouvez utiliser la console de gestion AWS pour accéder aux événements et aux notifications associés à votre application.

Pour afficher les événements

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le volet de navigation, sélectionnez Événements.

Time	Type	Details
2020-03-09 17:14:06 UTC-0700	INFO	createConfigurationTemplate completed successfully
2020-03-09 17:14:06 UTC-0700	INFO	createConfigurationTemplate is starting.
2020-03-03 04:16:55 UTC-0800	INFO	Environment health has transitioned from Int... update completed 85 seconds ago and took
2020-03-03 04:16:07 UTC-0800	INFO	Environment update completed successfully
2020-03-03 04:16:07 UTC-0800	INFO	Successfully deployed new configuration to...

La page Événements affiche une liste de tous les événements enregistrés pour l'environnement. Vous pouvez parcourir la liste en choisissant < (précédent), > (suivant) ou les numéros de page. Vous pouvez filtrer selon le type d'événement affiché à l'aide de la liste déroulante Severity (Gravité).

L'interface de ligne de commande EB (p. 1027) et l'AWS CLI fournissent toutes les deux des commandes permettant de récupérer les événements. Si vous gérez votre environnement via l'interface de ligne de commande EB, utilisez [eb events \(p. 1090\)](#) pour imprimer la liste des événements. Cette commande inclut également une option --follow, qui continue d'afficher les nouveaux événements jusqu'à ce que vous appuyiez sur Ctrl+C pour arrêter la sortie.

Pour extraire des événements via l'AWS CLI, utilisez la commande `describe-events` et spécifiez l'environnement par son nom ou son ID :

```
$ aws elasticbeanstalk describe-events --environment-id e-gbjzqccra3
{
    "Events": [
        {
            "ApplicationName": "elastic-beanstalk-example",
            "EnvironmentName": "elasticBeanstalkExa-env",
            "Severity": "INFO",
```

```
    "RequestId": "a4c7bfd6-2043-11e5-91e2-9114455c358a",
    "Message": "Environment update completed successfully.",
    "EventDate": "2015-07-01T22:52:12.639Z"
},
...
```

Pour plus d'informations sur les outils de ligne de commande, consultez [Outils \(p. 1027\)](#).

## Affichage de la liste des instances de serveur et connexion à ces instances

Vous pouvez afficher une liste des instances Amazon EC2 exécutant votre environnement d'applications AWS Elastic Beanstalk via la console Elastic Beanstalk. Vous pouvez vous connecter aux instances en utilisant tout client SSH. Vous pouvez vous connecter aux instances exécutant Windows à l'aide de Bureau à distance.

Quelques remarques concernant les environnements de développement spécifiques :

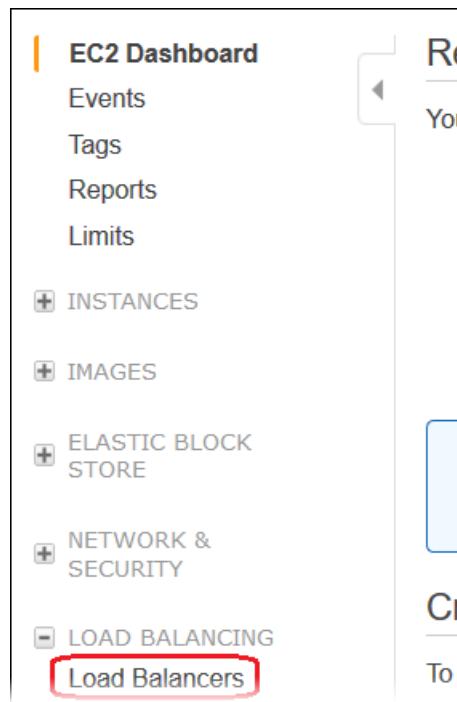
- Pour plus d'informations sur le répertorage et la connexion à des instances de serveur à l'aide de AWS Toolkit for Eclipse, consultez [Affichage de la liste des instances de serveur et connexion à ces instances \(p. 156\)](#).
- Pour plus d'informations sur le répertorage et la connexion à des instances de serveur à l'aide de AWS Toolkit for Visual Studio, consultez [Affichage de la liste des instances de serveur et connexion à ces instances \(p. 246\)](#).

### Important

Avant de pouvoir accéder à vos instances Amazon EC2 provisionnées par Elastic Beanstalk, vous devez créer une paire de clés Amazon EC2 et configurer vos instances Amazon EC2 provisionnées par Elastic Beanstalk pour utiliser la paire de clés Amazon EC2. Vous pouvez configurer vos paires de clés Amazon EC2 via la console de gestion AWS. Pour de plus amples informations sur la création d'une paire de clés pour Amazon EC2, veuillez consulter le Guide de démarrage d'Amazon EC2. Pour plus d'informations sur la façon de configurer vos instances Amazon EC2 pour utiliser une paire de clés Amazon EC2, consultez [EC2 key pair \(p. 628\)](#). Par défaut, Elastic Beanstalk n'active pas les connexions à distance aux instances EC2 dans un conteneur Windows, sauf pour les conteneurs Windows hérités. (Elastic Beanstalk configure des instances EC2 dans des conteneurs Windows existants pour utiliser le port 3389 pour des connexions RDP.) Vous pouvez activer des connexions à distance à vos instances EC2 exécutant Windows en ajoutant une règle à un groupe de sécurité qui autorise le trafic entrant pour les instances. Nous vous recommandons vivement de supprimer la règle lorsque vous mettez fin à votre connexion à distance. Vous pouvez ajouter la règle à nouveau la prochaine fois que vous avez besoin de vous connecter à distance. Pour de plus amples informations, veuillez consulter [Ajout d'une règle pour le trafic RDP entrant vers une instance Windows](#) et [Connexion à votre instance Windows](#) dans le Guide de l'utilisateur Amazon Elastic Compute Cloud pour Microsoft Windows.

Pour afficher les instances Amazon EC2 d'un environnement et s'y connecter

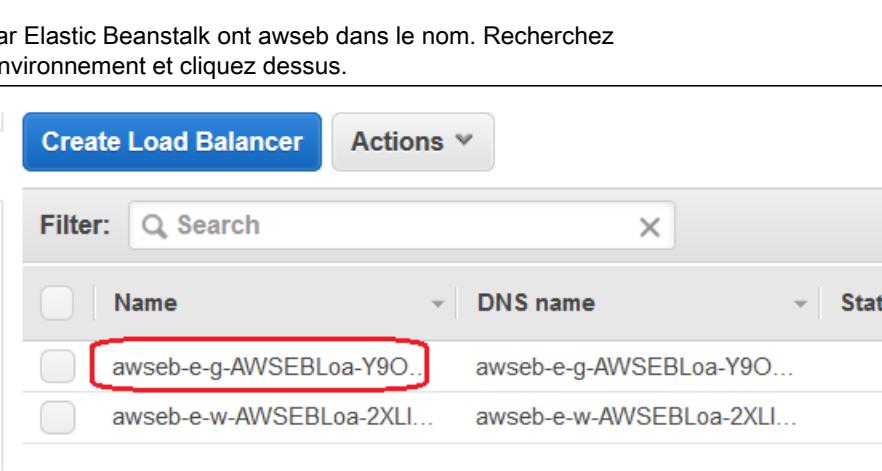
1. Ouvrez la console Amazon EC2 à l'adresse <https://console.aws.amazon.com/ec2/>.
2. Dans le panneau de navigation de la console, choisissez Équilibreurs de charge.



The screenshot shows the AWS EC2 Dashboard. On the left sidebar, under the 'LOAD BALANCING' section, the 'Load Balancers' link is highlighted with a red box. The main content area is titled 'Resources' and displays statistics for Amazon EC2 resources in the US East (N. Virginia) region:

Category	Value
Running Instances	2
Dedicated Hosts	0
Volumes	2
Key Pairs	0
Placement Groups	0

Below the resource stats, there is a promotional message: "Just need a simple virtual private server? Get everything you need to ju... networking – for a low, predictable price. [Try Amazon Lightsail for free.](#)"



The screenshot shows the AWS EC2 Instances page. The left sidebar has a 'LOAD BALANCING' section with a 'Load Balancers' link. The main content area is titled 'Create Instance' and contains the text: "To start using Amazon EC2 you will want to launch a virtual server, known as an instance. You can launch an instance by selecting one of the pre-configured Amazon Machine Images (AMIs) or creating your own from scratch." Below this, there is a table listing existing load balancers:

Name	DNS name	Status
awseb-e-g-AWSEBLoa-Y9O...	awseb-e-g-AWSEBLoa-Y9O...	Active
awseb-e-w-AWSEBLoa-2XLI...	awseb-e-w-AWSEBLoa-2XLI...	Active

Row 1 (awseb-e-g-AWSEBLoa-Y9O...) is highlighted with a red box.

3. Les équilibreurs de charge créés par Elastic Beanstalk ont awseb dans le nom. Recherchez l'équilibrleur de charge pour votre environnement et cliquez dessus.

4. Dans le volet inférieur de la console, choisissez l'onglet Instances.

The screenshot shows the AWS Elastic Beanstalk Instances tab for a load balancer named "awseb-e-g-AWSEBLoa-Y9OOSHJQWI". The "Instances" tab is selected. A red box highlights the "Instances" tab and the instance ID "i-97d76d0e" in the list. The table has columns: Instance ID, Name, and Availability Zone. One row is visible: "i-97d76d0e", "Default-Environment", and "us-east-1c".

Instance ID	Name	Availability Zone
i-97d76d0e	Default-Environment	us-east-1c

Une liste des instances qu'utilise l'équilibrer de charge pour votre environnement Elastic Beanstalk s'affiche. Notez un ID d'instance auquel vous souhaitez vous connecter.

5. Dans le panneau de navigation de la console Amazon EC2, choisissez Instances et trouvez votre ID d'instance dans la liste.

The screenshot shows the AWS EC2 Instances page. The left sidebar shows "INSTANCES" with "Instances" selected, highlighted by a red box. The main pane lists instances with a search bar at the top. A red box highlights the instance ID "i-97d76d0e" in the list. The table columns are: Name, Instance ID, and Instance Type. Two rows are visible: "mirfirstrailsap..." with ID "i-08cef90bbb58b3b5" and "t1.micro", and "Default-Envir..." with ID "i-97d76d0e" and "t1.micro".

Name	Instance ID	Instance Type
mirfirstrailsap...	i-08cef90bbb58b3b5	t1.micro
Default-Envir...	i-97d76d0e	t1.micro

6. Effectuez un clic droit sur l'ID d'instance pour l'instance Amazon EC2 en cours d'exécution dans l'équilibrer de charge de votre environnement, puis sélectionnez Connect dans le menu contextuel.
7. Notez adresse DNS publique de l'instance sur l'onglet Description.
8. Connectez-vous à une instance exécutant Linux en utilisant le client SSH de votre choix, puis tapez `ssh -i .ec2/mykeypair.pem ec2-user@<public-DNS-of-the-instance>`.

Pour de plus amples informations sur la connexion à une instance Linux Amazon EC2, veuillez consulter [Premiers pas avec les instances Amazon EC2 Linux](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

Si votre environnement Elastic Beanstalk utilise la [plateforme .NET sur Windows Server \(p. 196\)](#), veuillez consulter [Premiers pas avec les instances Amazon EC2 Windows](#) dans le Guide de l'utilisateur Amazon EC2 pour les instances Windows.

# Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk

Les instances Amazon EC2 de votre environnement Elastic Beanstalk génèrent des journaux que vous pouvez afficher pour résoudre les problèmes avec vos fichiers de configuration ou d'application. Les journaux créés par le serveur web, le serveur d'applications, les scripts de plateforme Elastic Beanstalk et AWS CloudFormation sont stockés localement sur des instances individuelles. Vous pouvez les récupérer facilement avec la [console de gestion d'environnement \(p. 429\)](#) ou l'interface de ligne de commande EB. Vous pouvez également configurer votre environnement pour diffuser en temps réel les journaux dans Amazon CloudWatch Logs.

Les journaux des processus sont les 100 dernières lignes des fichiers journaux les plus couramment utilisés : les journaux opérationnels Elastic Beanstalk et les journaux provenant du serveur web ou du serveur d'applications. Lorsque vous demandez des journaux de queue dans la console de gestion d'environnement ou avec eb logs, une instance dans votre environnement concatène les entrées du journal les plus récentes dans un fichier texte unique et les télécharge sur Amazon S3.

Les journaux de groupe sont des journaux complets pour un plus large éventail de fichiers journaux, y compris des journaux yum et cron et plusieurs journaux AWS CloudFormation. Lorsque vous demandez des journaux de groupe, une instance de votre environnement rassemble les fichiers journaux complets dans une archive ZIP et les télécharge sur Amazon S3.

## Note

Les plateformes Elastic Beanstalk Windows Server ne prennent pas en charge les journaux groupés.

Pour télécharger les journaux soumis à rotation sur Amazon S3, les instances de votre environnement doivent avoir un [profil d'instance \(p. 22\)](#) avec l'autorisation d'écrire sur votre compartiment Elastic Beanstalk Amazon S3. Ces autorisations sont incluses dans le profil d'instance par défaut qu'Elastic Beanstalk vous invite à créer lorsque vous lancez un environnement dans la console Elastic Beanstalk pour la première fois.

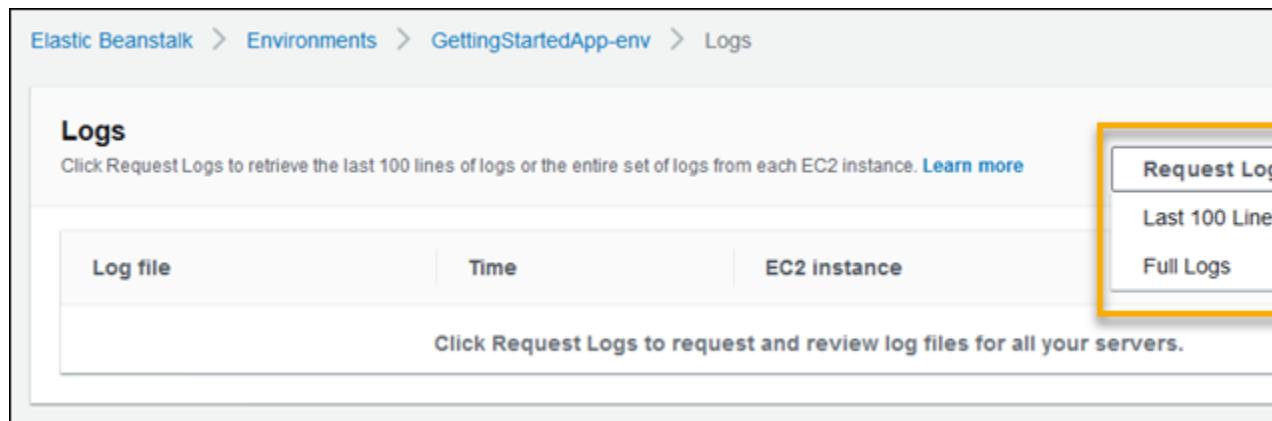
## Pour récupérer des journaux d'instance

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

## Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le volet de navigation, sélectionnez Logs.
4. Choisissez Request Logs (Journaux de demande), puis choisissez le type de journaux à récupérer. Pour obtenir des journaux de processus, choisissez Last 100 Lines (100 dernières lignes). Pour obtenir des journaux de bundle, choisissez Full Logs (Journaux complets).



- Quand Elastic Beanstalk a fini de récupérer vos journaux, choisissez Download (Télécharger).

Elastic Beanstalk stocke les journaux de queue et de bundle dans un compartiment Amazon S3 et génère une URL Amazon S3 pré-signée que vous pouvez utiliser pour accéder à vos journaux. Elastic Beanstalk supprime les fichiers d'Amazon S3 après une durée de 15 minutes.

**Warning**

Quiconque possédant l'URL Amazon S3 pré-signée peut consulter les fichiers avant qu'ils ne soient supprimés. Faites en sorte que seules les parties approuvées aient accès à l'URL.

**Note**

Votre stratégie utilisateur doit disposer de l'autorisation `s3:DeleteObject`. Elastic Beanstalk utilise vos autorisations utilisateur pour supprimer les journaux d'Amazon S3.

Pour conserver des journaux, vous pouvez configurer votre environnement afin de publier les journaux dans Amazon S3 automatiquement après leur rotation. Pour activer la rotation des journaux dans Amazon S3, suivez la procédure présentée dans [Configuration de l'affichage des journaux d'instance \(p. 641\)](#). Les instances dans votre environnement essaient de télécharger des journaux qui ont fait l'objet d'une rotation une fois par heure.

Si votre application génère des journaux dans un emplacement qui ne fait pas partie de la configuration par défaut de la plateforme de votre environnement, vous pouvez étendre la configuration par défaut à l'aide des fichiers de configuration ([.ebextensions \(p. 737\)](#)). Vous pouvez ajouter les fichiers journaux de votre application aux journaux de processus, aux journaux de groupe ou à la rotation des journaux.

Pour la diffusion des journaux en temps réel et le stockage à long terme, configurez l'environnement pour [diffuser les journaux vers Amazon CloudWatch Logs \(p. 889\)](#).

**Sections**

- [Emplacement des journaux sur les instances Amazon EC2 \(p. 886\)](#)
- [Emplacement des journaux dans Amazon S3 \(p. 886\)](#)
- [Paramètres de rotation des journaux sous Linux \(p. 887\)](#)
- [Extension de la configuration de tâche de journal par défaut \(p. 887\)](#)
- [Diffusion de fichiers journaux vers les Amazon CloudWatch Logs \(p. 889\)](#)

## Emplacement des journaux sur les instances Amazon EC2

Les journaux sont stockés dans des emplacements standard des instances Amazon EC2 de votre environnement. Elastic Beanstalk génère les journaux suivants.

Linux

- `/var/log/eb-activity.log`
- `/var/log/eb-commandprocessor.log`

Windows Server

- `C:\Program Files\Amazon\ElasticBeanstalk\logs\`
- `C:\cfn\logs\cfn-init.log`

Ces journaux contiennent des messages sur les activités de déploiement, y compris des messages liés aux fichiers de configuration ([.ebextensions \(p. 737\)](#)).

Chaque serveur d'applications et web stocke des journaux dans son propre dossier :

- Apache : `/var/log/httpd/`
- IIS : `C:\inetpub\wwwroot\`
- Node.js : `/var/log/nodejs/`
- nginx : `/var/log/nginx/`
- Passenger : `/var/app/support/logs/`
- Puma : `/var/log/puma/`
- Python : `/opt/python/log/`
- Tomcat : `/var/log/tomcat8/`

## Emplacement des journaux dans Amazon S3

Lorsque vous demandez les journaux de queue ou de bundle à partir de votre environnement, ou lorsque des instances téléchargent les journaux ayant fait l'objet d'une rotation, ils sont stockés dans votre compartiment Elastic Beanstalk d'Amazon S3. Elastic Beanstalk crée un compartiment nommé `elasticbeanstalk-region-account-id` pour chaque région AWS dans laquelle vous créez des environnements. Dans ce compartiment, les journaux sont stockés dans le chemin d'accès `resources/environments/logs/logtype/environment-id/instance-id`.

Par exemple, les journaux de l'instance `i-0a1fd158`, dans l'environnement Elastic Beanstalk `e-mpcwnwheky` dans la région AWS `us-west-2` dans le compte `123456789012`, sont stockés aux emplacements suivants :

- Journaux de queue :

```
s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/  
tail/e-mpcwnwheky/i-0a1fd158
```

- Journaux de groupe :

```
s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/  
bundle/e-mpcwnwheky/i-0a1fd158
```

- Journaux ayant fait l'objet d'une rotation :

```
s3://elasticbeanstalk-us-west-2-123456789012/resources/environments/logs/  
publish/e-mpcwnwheky/i-0a1fd158
```

#### Note

Vous trouverez l'ID de votre environnement dans la console de gestion d'environnement.

Elastic Beanstalk supprime automatiquement les journaux de queue et de bundle d'Amazon S3 15 minutes après leur création. Les journaux ayant fait l'objet d'une rotation sont conservés jusqu'à ce que vous les supprimiez ou les déplacez vers S3 Glacier.

## Paramètres de rotation des journaux sous Linux

Sur les plates-formes Linux, Elastic Beanstalk utilise `logrotate` pour soumettre les journaux à rotation régulièrement. Si la configuration a été effectuée, une fois qu'un journal a effectué sa rotation localement, la tâche de rotation le sélectionne et le télécharge sur Amazon S3. Les journaux qui ont fait l'objet d'une rotation localement ne s'affichent pas dans les journaux de processus ou de groupe par défaut.

Vous pouvez trouver les fichiers de configuration Elastic Beanstalk pour `logrotate` dans `/etc/logrotate.elasticbeanstalk.hourly/`. Ces paramètres de rotation sont propres à la plateforme et sont susceptibles de changer dans de futures versions de la plateforme. Pour plus d'informations concernant les paramètres disponibles et les exemples de configurations, exécutez `man logrotate`.

Les fichiers de configuration sont invoqués par des tâches cron dans `/etc/cron.hourly/`. Pour obtenir plus d'informations concernant `cron`, exécutez `man cron`.

## Extension de la configuration de tâche de journal par défaut

Elastic Beanstalk utilise des fichiers des sous-répertoires de `/opt/elasticbeanstalk/tasks` (Linux) ou `C:\Program Files\Amazon\ElasticBeanstalk\config` (Windows Server) sur l'instance Amazon EC2 pour configurer les tâches pour les journaux de queue, les journaux de bundle et la rotation des journaux.

Sous Linux :

- Journaux de queue :

```
/opt/elasticbeanstalk/tasks/taillogs.d/
```

- Journaux de groupe :

```
/opt/elasticbeanstalk/tasks/bundlelogs.d/
```

- Journaux ayant fait l'objet d'une rotation :

```
/opt/elasticbeanstalk/tasks/publishlogs.d/
```

Sous Windows Server :

- Journaux de queue :

```
c:\Program Files\Amazon\ElasticBeanstalk\config\taillogs.d\
```

- Journaux ayant fait l'objet d'une rotation :

```
c:\Program Files\Amazon\ElasticBeanstalk\config\publogs.d\
```

Par exemple, le fichier `eb-activity.conf` sous Linux ajoute deux fichiers journaux à la tâche de journaux de processus.

```
/opt/elasticbeanstalk/tasks/taillogs.d/eb-activity.conf
```

```
/var/log/eb-commandprocessor.log  
/var/log/eb-activity.log
```

Vous pouvez utiliser les fichiers de configuration d'environnement ([.ebextensions \(p. 737\)](#)) pour ajouter vos propres fichiers `.conf` à ces dossiers. Un fichier `.conf` répertorie les fichiers journaux spécifiques à votre application, qu'Elastic Beanstalk ajoute aux tâches des fichiers journaux.

Utilisez la section [files \(p. 744\)](#) pour ajouter des fichiers de configuration aux tâches que vous voulez modifier. Par exemple, le texte de configuration suivant ajoute un fichier de configuration du journal à chaque instance dans votre environnement. Ce fichier de configuration du journal, `cloud-init.conf`, ajoute `/var/log/cloud-init.log` aux journaux de processus.

```
files:  
  "/opt/elasticbeanstalk/tasks/taillogs.d/cloud-init.conf" :  
    mode: "000755"  
    owner: root  
    group: root  
    content: |  
      /var/log/cloud-init.log
```

Ajoutez ce texte à un fichier avec l'extension de nom de fichier `.config` à votre bundle de fichiers source sous un dossier nommé `.ebextensions`.

```
~/workspace/my-app  
|-- .ebextensions  
|  '-- tail-logs.config  
|-- index.php  
`-- styles.css
```

Sur les plateformes Linux, vous pouvez également utiliser des caractères génériques dans les configurations de tâche de journal. Ce fichier de configuration ajoute tous les fichiers avec l'extension de nom de fichier `.log` provenant du dossier `log` situé à la racine de l'application aux journaux de bundle.

```
files:  
  "/opt/elasticbeanstalk/tasks/bundlelogs.d/applogs.conf" :  
    mode: "000755"  
    owner: root  
    group: root  
    content: |  
      /var/app/current/log/*.log
```

Les configurations de tâche de journal ne prennent pas en charge les caractères génériques sur les plateformes Windows.

#### Note

Pour vous familiariser avec les procédures de personnalisation des journaux, vous pouvez déployer un exemple d'application à l'aide de l'[interface de ligne de commande EB \(p. 1027\)](#). Pour cela, l'interface de ligne de commande EB crée un répertoire d'application local qui contient

un sous-répertoire `.ebextensions` avec un exemple de configuration. Vous pouvez également utiliser les fichiers journaux de l'exemple d'application pour explorer la fonction d'extraction du journal décrite dans cette rubrique. Pour de plus amples informations sur la création d'un exemple d'application avec l'interface de ligne de commande EB, veuillez consulter [Principes de base de l'interface de ligne de commande EB \(p. 1040\)](#).

Pour de plus amples informations sur l'utilisation des fichiers de configuration, veuillez consulter [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

De la même manière que pour l'extension des journaux de processus et des journaux de groupe, vous pouvez étendre la rotation des journaux à l'aide d'un fichier de configuration. Chaque fois que Elastic Beanstalk fait pivoter ses propres journaux et les télécharge sur Amazon S3, il soumet également à rotation et télécharge vos journaux supplémentaires. L'extension de la rotation des journaux se comporte différemment en fonction du système d'exploitation utilisé par la plateforme. Les sections suivantes décrivent les deux cas possibles.

## Extension de la rotation des journaux sous Linux

Comme expliqué dans [Paramètres de rotation des journaux sous Linux \(p. 887\)](#), Elastic Beanstalk utilise `logrotate` pour soumettre les journaux à rotation sur les plateformes Linux. Lorsque vous configurez les fichiers journaux de votre application pour la rotation des fichiers, l'application n'a pas besoin de créer de copies des fichiers journaux. Elastic Beanstalk configure `logrotate` pour créer une copie des fichiers journaux de votre application pour chaque rotation. Par conséquent, l'application doit conserver les fichiers journaux déverrouillés lorsqu'elle n'écrit pas activement dans ces journaux.

## Extension de la rotation des journaux sous Windows Server

Sur Windows Server, lorsque vous configurez les fichiers journaux de votre application pour la rotation, l'application doit effectuer une rotation régulière des fichiers journaux. Elastic Beanstalk recherche les fichiers dont le nom commence par le modèle que vous avez configuré et les sélectionne pour les charger vers Amazon S3. En outre, les points dans le nom du fichier sont ignorés et Elastic Beanstalk considère que le nom du fichier journal de base s'arrête au point.

Elastic Beanstalk charge toutes les versions d'un fichier journal de base, à l'exception de la plus récente qu'il considère comme le fichier journal actif de l'application, qui peut parfois être verrouillé. Votre application peut, par conséquent, garder le fichier journal actif verrouillé entre les rotations.

Par exemple, votre application écrit dans un fichier journal nommé `my_log.log`, et vous spécifiez ce nom dans votre fichier `.conf`. L'application effectue une rotation périodique du fichier. Pendant le cycle de rotation d'Elastic Beanstalk, les fichiers suivants se trouvent dans le dossier des fichiers journaux : `my_log.log`, `my_log.0800.log`, `my_log.0830.log`. Elastic Beanstalk considère tous ces fichiers comme des versions du nom de base `my_log`. Le fichier `my_log.log` comporte l'heure de modification la plus récente. Ainsi, Elastic Beanstalk charge uniquement les deux autres fichiers, `my_log.0800.log` et `my_log.0830.log`.

## Diffusion de fichiers journaux vers les Amazon CloudWatch Logs

Vous pouvez configurer votre environnement pour diffuser les journaux sur Amazon CloudWatch Logs dans la console Elastic Beanstalk ou à l'aide des [options de configuration \(p. 658\)](#). Avec CloudWatch Logs, chaque instance de votre environnement diffuse les journaux vers des groupes de journaux que vous pouvez configurer pour être conservés pendant des semaines ou des années, même après l'arrêt de votre environnement.

L'ensemble des journaux diffusés varie selon l'environnement, mais il inclut toujours `eb-activity.log` et les journaux d'accès provenant du serveur proxy nginx ou Apache qui s'exécute devant votre application.

Vous pouvez configurer la diffusion de journaux dans la console Elastic Beanstalk [pendant la création de l'environnement \(p. 448\)](#) ou [pour un environnement existant \(p. 641\)](#). Dans l'exemple suivant, les journaux sont conservés jusqu'à sept jours, même lorsque l'environnement est résilié.

The screenshot shows the 'Instance log streaming to CloudWatch Logs' configuration section. It includes fields for Log groups, Log streaming (with a note about standard CloudWatch charges), Retention (set to 7 days), and Lifecycle (set to 'Keep logs after terminating environment').

Le [fichier de configuration \(p. 737\)](#) suivant active la diffusion de journaux avec une conservation de 180 jours, même si l'environnement a été résilié.

Example .ebextensions/log-streaming.config

```
option_settings:  
  aws:elasticbeanstalk:cloudwatch:logs:  
    StreamLogs: true  
    DeleteOnTerminate: false  
    RetentionInDays: 180
```

# Utilisation d'Elastic Beanstalk avec d'autres services AWS

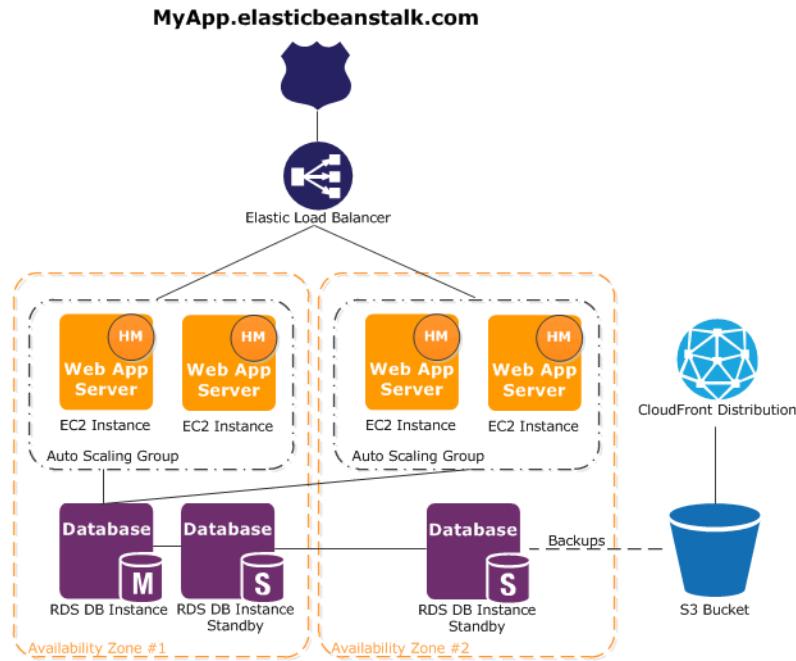
Pour implémenter les environnements de votre application, Elastic Beanstalk gère les ressources d'autres services AWS ou utilise leurs fonctionnalités. En outre, Elastic Beanstalk s'intègre aux services AWS qu'il n'utilise pas directement dans le cadre de vos environnements. Les rubriques de cette section décrivent plusieurs façons d'utiliser ces services supplémentaires avec votre application Elastic Beanstalk.

## Rubriques

- [Présentation de l'architecture \(p. 891\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon CloudFront \(p. 892\)](#)
- [Journalisation des appels d'API Elastic Beanstalk avec AWS CloudTrail \(p. 893\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon CloudWatch \(p. 894\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon CloudWatch Logs \(p. 895\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon EventBridge \(p. 906\)](#)
- [Recherche et suivi des ressources Elastic Beanstalk avec AWS Config \(p. 913\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon DynamoDB \(p. 917\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon ElastiCache \(p. 918\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon Elastic File System \(p. 918\)](#)
- [Utilisation d'Elastic Beanstalk avec AWS Identity and Access Management \(p. 920\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon RDS \(p. 991\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon S3 \(p. 1003\)](#)
- [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#)

## Présentation de l'architecture

Le diagramme suivant illustre un exemple d'architecture d'Elastic Beanstalk à travers plusieurs zones de disponibilité travaillant avec d'autres produits AWS tels qu'Amazon CloudFront, Amazon Simple Storage Service (Amazon S3) et Amazon Relational Database Service (Amazon RDS).



Pour planifier la tolérance aux pannes, il est recommandé d'avoir N+1 instances Amazon EC2 et de répartir vos instances sur plusieurs zones de disponibilité. Dans le cas improbable où une seule zone de disponibilité s'arrête, vous aurez toujours vos autres instances Amazon EC2 en cours d'exécution dans une autre zone de disponibilité. Vous pouvez ajuster Amazon EC2 Auto Scaling afin d'autoriser un nombre minimal d'instances, ainsi que plusieurs zones de disponibilité. Pour obtenir des instructions sur la façon de procéder, veuillez consulter [Groupe Auto Scaling pour votre environnement Elastic Beanstalk \(p. 547\)](#). Pour plus d'informations sur la création d'applications tolérantes aux pannes, accédez à [Création d'applications tolérantes aux pannes sur AWS](#).

Les sections suivantes traitent plus en détail de l'intégration à Amazon CloudFront, Amazon CloudWatch, Amazon DynamoDB Amazon ElastiCache, Amazon RDS, Amazon Route 53, Amazon Simple Storage Service, Amazon VPC et IAM.

## Utilisation d'Elastic Beanstalk avec Amazon CloudFront

Amazon CloudFront est service Web qui accélère la distribution de vos contenus Web statiques et dynamiques, tels que vos fichiers multimédias, images et contenus .html, .css et .php, à destination des utilisateurs finaux. CloudFront diffuse votre contenu à travers un réseau mondial d'emplacements périphériques. Lorsqu'un utilisateur final demande un contenu que vous diffusez via CloudFront, il est acheminé vers l'emplacement périphérique qui fournit la latence la plus basse, afin que le contenu soit diffusé avec les meilleures performances possibles. Si ce contenu se trouve déjà dans l'emplacement périphérique, CloudFront le diffuse immédiatement. Si le contenu ne se trouve pas encore dans l'emplacement périphérique, CloudFront l'extracte d'un compartiment Amazon S3 ou d'un serveur HTTP (par exemple, un serveur Web) que vous avez identifié comme étant l'origine de la version définitive de votre contenu.

Après avoir créé et déployé votre application Elastic Beanstalk, vous pouvez vous inscrire à CloudFront et commencer à utiliser CloudFront pour distribuer votre contenu. Pour de plus amples informations sur CloudFront, veuillez consulter le [Manuel du développeur Amazon CloudFront](#).

# Journalisation des appels d'API Elastic Beanstalk avec AWS CloudTrail

Elastic Beanstalk est intégré à AWS CloudTrail, service qui enregistre les actions effectuées par un utilisateur, un rôle ou un service AWS dans Elastic Beanstalk. CloudTrail capture tous les appels d'API pour Elastic Beanstalk en tant qu'événements, y compris les appels de la console Elastic Beanstalk, de l'interface de ligne de commande EB et de votre code vers les API Elastic Beanstalk. Si vous créez un journal de suivi, vous pouvez diffuser en continu les événements CloudTrail dans un compartiment Amazon S3, y compris les événements pour Elastic Beanstalk. Si vous ne configurez pas de journal de suivi, vous pouvez toujours afficher les événements les plus récents dans la console CloudTrail dans Event history (Historique des événements). Avec les informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été envoyée à Elastic Beanstalk, l'adresse IP à partir de laquelle la demande a été effectuée, l'auteur et la date de la demande, ainsi que d'autres détails.

Pour de plus amples informations sur CloudTrail, veuillez consulter le [Guide de l'utilisateur AWS CloudTrail](#).

## Informations Elastic Beanstalk dans CloudTrail

CloudTrail est activé dans votre compte AWS lors de la création de ce dernier. Quand une activité a lieu dans Elastic Beanstalk, cette activité est enregistrée dans un événement CloudTrail avec d'autres événements de services AWS dans Event history (Historique des événements). Vous pouvez afficher, rechercher et télécharger des événements récents dans votre compte AWS. Pour de plus amples informations, veuillez consulter [Affichage des événements avec l'historique des événements CloudTrail](#).

Pour un enregistrement continu des événements dans votre compte AWS, y compris les événements pour Elastic Beanstalk, créez un journal d'activité. Un journal de suivi permet à CloudTrail de livrer des fichiers journaux dans un compartiment Amazon S3. Par défaut, lorsque vous créez un journal de suivi dans la console, il s'applique à toutes les régions. Le journal de suivi consigne les événements de toutes les régions dans la partition AWS et livre les fichiers journaux dans le compartiment Amazon S3 de votre choix. En outre, vous pouvez configurer d'autres services AWS pour analyser en profondeur les données d'événements collectées dans les journaux CloudTrail et prendre les mesures nécessaires. Pour plus d'informations, consultez :

- [Présentation de la création d'un journal de suivi](#)
- [Intégrations et services pris en charge par CloudTrail](#)
- [Configuration des notifications Amazon SNS pour CloudTrail](#)
- [Réception de fichiers journaux CloudTrail de plusieurs régions et Réception de fichiers journaux CloudTrail de plusieurs comptes](#)

Toutes les actions Elastic Beanstalk sont enregistrées par CloudTrail et sont documentées dans la [Référence d'API AWS Elastic Beanstalk](#). À titre d'exemple, les appels vers les actions `DescribeApplications`, `UpdateEnvironment` et `ListTagsForResource` génèrent des entrées dans les fichiers journaux CloudTrail.

Chaque événement ou entrée du journal contient des informations sur la personne qui a généré la demande. Les informations relatives à l'identité permettent de déterminer les éléments suivants :

- Si la demande a été effectuée avec les informations d'identification utilisateur racine ou IAM.
- Si la demande a été effectuée avec des informations d'identification de sécurité temporaires pour un rôle ou un utilisateur fédéré.
- Si la demande a été effectuée par un autre service AWS.

Pour de plus amples informations, veuillez consulter l'[élément userIdentity CloudTrail](#).

## Présentation des entrées du fichier journal Elastic Beanstalk

Un journal de suivi est une configuration qui permet la livraison d'événements sous forme de fichiers journaux vers un compartiment Amazon S3 que vous spécifiez. Les fichiers journaux CloudTrail peuvent contenir une ou plusieurs entrées de journal. Un événement représente une demande individuelle émise à partir d'une source quelconque et comprend des informations sur l'action demandée, la date et l'heure de l'action, les paramètres de la demande, etc. Les fichiers journaux CloudTrail ne constituent pas une trace de pile ordonnée d'appels d'API publics. Ils ne suivent aucun ordre précis.

L'exemple suivant montre une entrée de journal CloudTrail qui illustre l'action `UpdateEnvironment` appelée par un utilisateur IAM nommé `intern`, pour l'environnement `sample-env` dans l'application `sample-app`.

```
{
  "Records": [
    {
      "eventVersion": "1.05",
      "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIXDAYQEXAMPLEUMLYNGL",
        "arn": "arn:aws:iam::123456789012:user/intern",
        "accountId": "123456789012",
        "accessKeyId": "ASXIAGXAMPLEQULKNXV",
        "userName": "intern",
        "sessionContext": {
          "attributes": {
            "mfaAuthenticated": "false",
            "creationDate": "2016-04-22T00:23:24Z"
          }
        },
        "invokedBy": "signin.amazonaws.com"
      },
      "eventTime": "2016-04-22T00:24:14Z",
      "eventSource": "elasticbeanstalk.amazonaws.com",
      "eventName": "UpdateEnvironment",
      "awsRegion": "us-west-2",
      "sourceIPAddress": "255.255.255.54",
      "userAgent": "signin.amazonaws.com",
      "requestParameters": {
        "applicationName": "sample-app",
        "environmentName": "sample-env",
        "optionSettings": []
      },
      "responseElements": null,
      "requestID": "84ae9ecf-0280-17ce-8612-705c7b132321",
      "eventID": "e48b6a08-c6be-4a22-99e1-c53139cbfb18",
      "eventType": "AwsApiCall",
      "recipientAccountId": "123456789012"
    }
  ]
}
```

## Utilisation d'Elastic Beanstalk avec Amazon CloudWatch

Amazon CloudWatch vous permet de surveiller, de gérer et de publier différentes métriques, et de configurer des actions d'alarme basées sur les données des métriques. La surveillance Amazon

CloudWatch vous permet de collecter, d'analyser et de visualiser les métriques relatives au système et aux applications. Vous pouvez ainsi prendre vos décisions plus rapidement et en toute confiance, dans le domaine opérationnel et commercial.

Vous pouvez utiliser Amazon CloudWatch pour collecter des métriques concernant vos ressources Amazon Web Services (AWS), telles que les performances de vos instances Amazon EC2. Vous pouvez également publier vos propres métriques directement sur Amazon CloudWatch. Grâce aux alarmes Amazon CloudWatch, vous pouvez appliquer plus facilement vos décisions, car elles vous permettent d'envoyer des notifications ou de modifier automatiquement les ressources que vous surveillez, en fonction des règles que vous définissez. Par exemple, vous pouvez créer des alarmes qui initient des actions Amazon EC2 Auto Scaling et Amazon Simple Notification Service (Amazon SNS) en votre nom.

Elastic Beanstalk utilise automatiquement Amazon CloudWatch pour vous aider à surveiller l'état de votre application et de votre environnement. Vous pouvez accéder à la console Amazon CloudWatch; pour consulter votre tableau de bord et obtenir une vue d'ensemble de toutes vos ressources et alarmes. Vous pouvez également afficher des métriques supplémentaires ou ajouter des métriques personnalisées.

Pour de plus amples informations sur Amazon CloudWatch, veuillez consulter le [Guide du développeur Amazon CloudWatch](#). Pour obtenir un exemple d'utilisation d'Amazon CloudWatch avec Elastic Beanstalk, veuillez consulter [the section called "Exemple : utilisation de métriques personnalisées Amazon CloudWatch" \(p. 750\)](#).

## Utilisation d'Elastic Beanstalk avec Amazon CloudWatch Logs

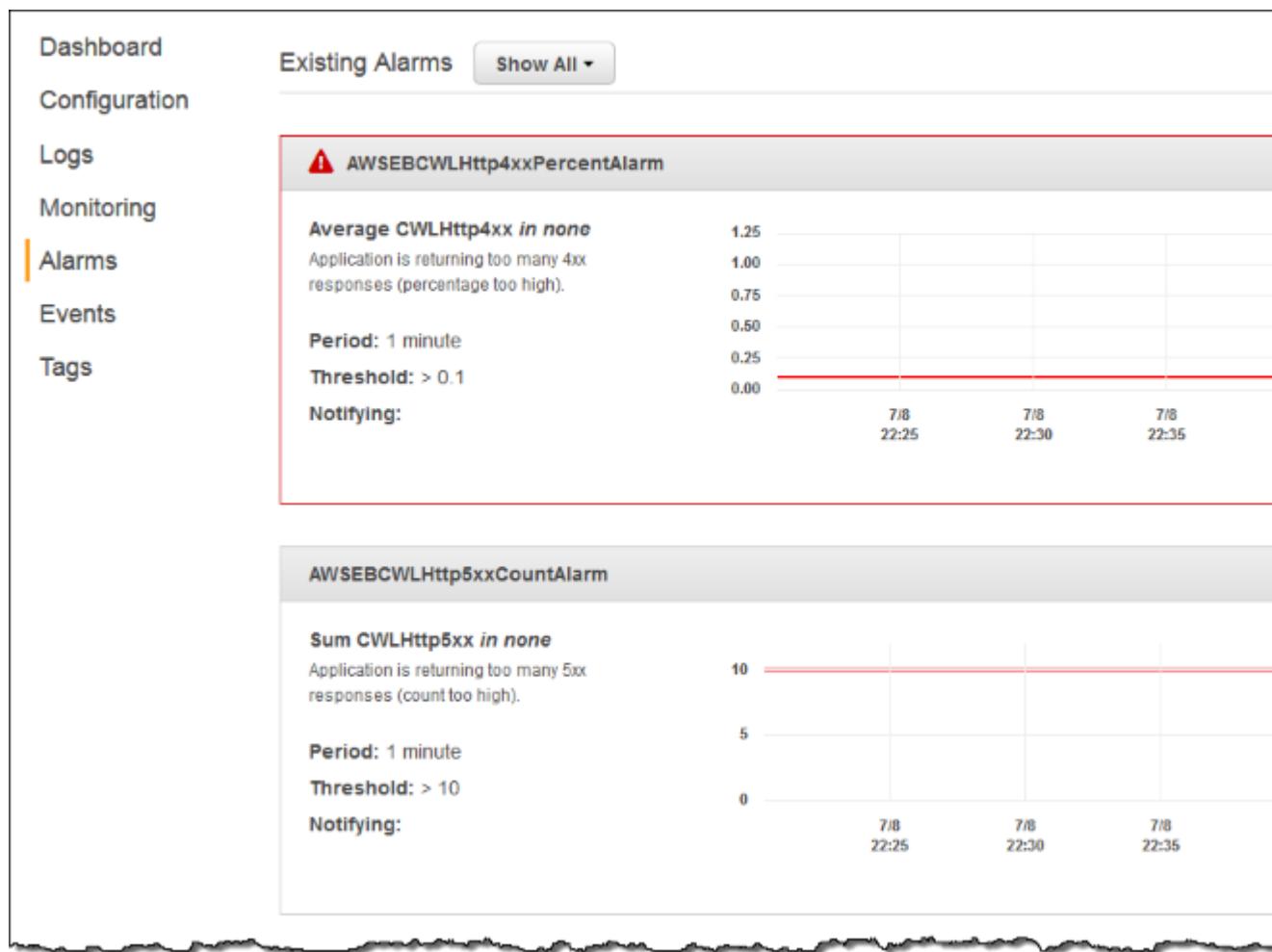
CloudWatch Logs vous permet de surveiller et d'archiver les fichiers journaux de votre application Elastic Beanstalk, de votre système, ainsi que les fichiers journaux personnalisés à partir des instances Amazon EC2 de vos environnements. Vous pouvez également configurer des alarmes qui vous permettent d'entreprendre plus facilement des actions en réponse à des événements de flux de journal spécifiques extraits par vos filtres de métriques. L'agent CloudWatch Logs installé sur chaque instance Amazon EC2 de votre environnement publie des points de données de métriques sur le service CloudWatch pour chaque groupe de journaux que vous configurez. Chaque groupe de journaux applique ses propres modèles de filtres afin de déterminer les événements de flux de journaux à envoyer à CloudWatch en tant que points de données. Les flux de journaux qui appartiennent au même groupe de journaux partagent les mêmes paramètres de contrôle d'accès, de surveillance et de rétention. Vous pouvez configurer Elastic Beanstalk pour diffuser automatiquement des journaux vers le service CloudWatch, comme décrit à la section [Diffusion de journaux d'instance vers CloudWatch Logs \(p. 901\)](#). Pour de plus amples informations sur CloudWatch Logs, notamment la terminologie et les concepts, veuillez consulter le [Guide de l'utilisateur Amazon CloudWatch Logs](#).

Outre les journaux d'instance, si vous activez les [rapports améliorés sur l'état \(p. 837\)](#) pour votre environnement, vous pouvez configurer ce dernier de sorte à diffuser les informations d'état vers CloudWatch Logs. Voir [Diffusion d'informations sur l'état de l'environnement Elastic Beanstalk vers Amazon CloudWatch Logs \(p. 903\)](#).

La figure suivante affiche la page Monitoring (Surveillance) et les graphiques d'un environnement configuré avec l'intégration CloudWatch Logs. Les exemples de métriques dans cet environnement sont nommés CWLHttp4xx et CWLHttp5xx. L'un des graphiques montre que la métrique CWLHttp4xx a déclenché une alarme en fonction des conditions spécifiées dans les fichiers de configuration.



La figure suivante présente la page Alarmes et des graphiques pour les exemples d'alarmes nommés AWSEBCWLHttp4xxPercentAlarm et AWSEBCWLHttp5xxCountAlarm qui correspondent aux métriques CWLHttp4xx et CWLHttp5xx, respectivement.



#### Rubriques

- [Conditions préalables pour la diffusion des journaux d'instance vers CloudWatch Logs \(p. 897\)](#)
- [Méthode de configuration de CloudWatch Logs par Elastic Beanstalk \(p. 898\)](#)
- [Diffusion de journaux d'instance vers CloudWatch Logs \(p. 901\)](#)
- [Résolution des problèmes liés à l'intégration de CloudWatch Logs \(p. 903\)](#)
- [Diffusion d'informations sur l'état de l'environnement Elastic Beanstalk vers Amazon CloudWatch Logs \(p. 903\)](#)

## Conditions préalables pour la diffusion des journaux d'instance vers CloudWatch Logs

Pour activer la diffusion de journaux des instances Amazon EC2 de votre environnement vers CloudWatch Logs, vous devez remplir les conditions suivantes :

- Plateforme – Étant donné que cette fonctionnalité est disponible uniquement dans les versions de plateforme publiées en même temps que [cette version](#) ou après, si vous utilisez une version de plateforme antérieure, mettez à jour votre environnement vers la configuration actuelle.

- Si vous ne disposez pas de la stratégie gérée Elastic Beanstalk AWSElasticBeanstalkWebTier ou AWSElasticBeanstalkWorkerTier dans votre [profil d'instance Elastic Beanstalk \(p. 22\)](#), vous devez ajouter le code suivant à votre profil pour activer cette fonctionnalité.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "logs:PutLogEvents",  
                "logs>CreateLogStream"  
            ],  
            "Resource": [  
                "*"  
            ]  
        }  
    ]  
}
```

## Méthode de configuration de CloudWatch Logs par Elastic Beanstalk

Elastic Beanstalk installe un agent CloudWatch Logs avec les paramètres de configuration par défaut sur chaque instance qu'il crée. Pour en savoir plus, veuillez consulter la [Référence de l'agent CloudWatch Logs](#).

Lorsque vous activez la diffusion de journaux d'instance vers CloudWatch Logs, Elastic Beanstalk envoie les fichiers journaux des instances de votre environnement à CloudWatch Logs. Les journaux diffusés varient d'une plateforme à une autre. Le tableau suivant répertorie les différents journaux par plateforme.

Plateforme	Journaux
Docker	<ul style="list-style-type: none"><li>/var/log/eb-engine.log</li><li>/var/log/eb-hooks.log</li><li>/var/log/docker</li><li>/var/log/docker-events.log</li><li>/var/log/eb-docker/containers/eb-current-app/stdouterr.log</li><li>/var/log/nginx/access.log</li><li>/var/log/nginx/error.log</li></ul>
Go	<ul style="list-style-type: none"><li>/var/log/eb-engine.log</li><li>/var/log/eb-hooks.log</li></ul>
Corretto	<ul style="list-style-type: none"><li>/var/log/web.stdout.log</li></ul>
.NET Core sous Linux	<ul style="list-style-type: none"><li>/var/log/nginx/access.log</li><li>/var/log/nginx/error.log</li></ul>
Node.js	<ul style="list-style-type: none"><li>/var/log/eb-engine.log</li><li>/var/log/eb-hooks.log</li></ul>
Python	<ul style="list-style-type: none"><li>/var/log/web.stdout.log</li><li>/var/log/httpd/access_log</li><li>/var/log/httpd/error_log</li><li>/var/log/nginx/access.log</li></ul>

Plateforme	Journaux
	<ul style="list-style-type: none"> <li>• /var/log/nginx/error.log</li> </ul>
Tomcat	<ul style="list-style-type: none"> <li>• /var/log/eb-engine.log</li> </ul>
PHP	<ul style="list-style-type: none"> <li>• /var/log/eb-hooks.log</li> <li>• /var/log/httpd/access_log</li> <li>• /var/log/httpd/error_log</li> <li>• /var/log/nginx/access.log</li> <li>• /var/log/nginx/error.log</li> </ul>
.NET sous Windows Server	<ul style="list-style-type: none"> <li>• C:\inetpub\logs\LogFiles\W3SVC1\u_ex*.log</li> <li>• C:\Program Files\Amazon\ElasticBeanstalk\logs\AWSDeployment.log</li> <li>• C:\Program Files\Amazon\ElasticBeanstalk\logs\Hooks.log</li> </ul>
Ruby	<ul style="list-style-type: none"> <li>• /var/log/eb-engine.log</li> <li>• /var/log/eb-hooks.log</li> <li>• /var/log/puma/puma.log</li> <li>• /var/log/web.stdout.log</li> <li>• /var/log/nginx/access.log</li> <li>• /var/log/nginx/error.log</li> </ul>

## Fichiers journaux sur les plateformes AMI Amazon Linux

Le tableau suivant répertorie les fichiers journaux diffusés à partir d'instances sur des branches de plateforme en fonction de l'AMI Amazon Linux (anciennement Amazon Linux 2), par plateforme.

Plateforme	Journaux
Docker	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/nginx/error.log</li> <li>• /var/log/docker-events.log</li> <li>• /var/log/docker</li> <li>• /var/log/nginx/access.log</li> <li>• /var/log/eb-docker/containers/eb-current-app/stdouterr.log</li> </ul>
Docker multiconteneur (générique)	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/ecs/ecs-init.log</li> <li>• /var/log/eb-ecs-mgr.log</li> <li>• /var/log/ecs/ecs-agent.log</li> <li>• /var/log/docker-events.log</li> </ul>
Glassfish (Docker préconfiguré)	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/nginx/error.log</li> <li>• /var/log/docker-events.log</li> <li>• /var/log/docker</li> <li>• /var/log/nginx/access.log</li> </ul>
Go (Docker préconfiguré)	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/nginx/error.log</li> </ul>

Plateforme	Journaux
	<ul style="list-style-type: none"> <li>• /var/log/docker-events.log</li> <li>• /var/log/docker</li> <li>• /var/log/nginx/access.log</li> </ul>
Python (Docker préconfiguré)	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/nginx/error.log</li> <li>• /var/log/docker-events.log</li> <li>• /var/log/docker</li> <li>• /var/log/nginx/access.log</li> </ul>
Go	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/nginx/error.log</li> <li>• /var/log/nginx/access.log</li> </ul>
Java SE	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/nginx/access.log</li> <li>• /var/log/nginx/error.log</li> <li>• /var/log/web-1.error.log</li> <li>• /var/log/web-1.log</li> </ul>
Tomcat	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/httpd/error_log</li> <li>• /var/log/httpd/access_log</li> <li>• /var/log/nginx/error_log</li> <li>• /var/log/nginx/access_log</li> </ul>
Node.js	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/nodejs/nodejs.log</li> <li>• /var/log/nginx/error.log</li> <li>• /var/log/nginx/access.log</li> <li>• /var/log/httpd/error.log</li> <li>• /var/log/httpd/access.log</li> </ul>
PHP	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/httpd/error_log</li> <li>• /var/log/httpd/access_log</li> </ul>
Python	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/httpd/error_log</li> <li>• /var/log/httpd/access_log</li> <li>• /opt/python/log/supervisord.log</li> </ul>
Ruby (Puma)	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/log/nginx/error.log</li> <li>• /var/log/puma/puma.log</li> <li>• /var/log/nginx/access.log</li> </ul>

Plateforme	Journaux
Ruby (Passenger)	<ul style="list-style-type: none"> <li>• /var/log/eb-activity.log</li> <li>• /var/app/support/logs/passenger.log</li> <li>• /var/app/support/logs/access.log</li> <li>• /var/app/support/logs/error.log</li> </ul>

Elastic Beanstalk configure des groupes de journaux dans CloudWatch Logs pour les différents fichiers journaux qu'il diffuse. Pour extraire des fichiers journaux spécifiques de CloudWatch Logs, vous devez connaître le nom du groupe de journaux correspondant. Le schéma d'attribution de noms des groupes de journaux dépend du système d'exploitation utilisé par la plateforme.

Pour les plateformes Linux, préfixez l'emplacement du fichier journal de l'instance avec `/aws/elasticbeanstalk/environment_name` pour obtenir le nom du groupe de journaux. Par exemple, pour extraire le fichier `/var/log/nginx/error.log`, indiquez le nom du groupe de journaux `/aws/elasticbeanstalk/environment_name/var/log/nginx/error.log`.

Pour les plateformes Windows, consultez le tableau suivant pour connaître le groupe de journaux correspondant à chaque fichier journal.

Fichier journal d'instance	Groupe de journaux
C:\Program Files\Amazon\ElasticBeanstalk\logs\AWSDeployment.log	/aws/elasticbeanstalk/<environment-name>/EBDeploy-Log
C:\Program Files\Amazon\ElasticBeanstalk\logs\Hooks.log	/aws/elasticbeanstalk/<environment-name>/EBHooks-Log
C:\inetpub\logs\LogFiles (ensemble du répertoire)	/aws/elasticbeanstalk/<environment-name>/IIS-Log

## Diffusion de journaux d'instance vers CloudWatch Logs

Vous pouvez activer la diffusion de journaux d'instance vers CloudWatch Logs à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande EB ou des options de configuration.

Avant de l'activer, configurez les autorisations IAM à utiliser avec l'agent CloudWatch Logs. Vous pouvez attacher la stratégie personnalisée ci-après au [profil d'instance \(p. 22\)](#) que vous attribuez à votre environnement.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents",
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams"
      ],
      "Resource": [
        "arn:aws:logs:region:account:log-group:log_group_name:*"
      ]
    }
  ]
}
```

```
        "*"
    ]
}
}
```

## Diffusion de journaux d'instance à l'aide de la console Elastic Beanstalk

Pour diffuser des journaux d'instance vers CloudWatch Logs

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Sous Diffusion des journaux d'instance vers CloudWatch Logs :
  - Activez Diffusion des journaux.
  - Définissez Conservation sur le nombre de jours de conservation des journaux.
  - Sélectionnez le paramètre de cycle de vie qui détermine si les journaux sont enregistrés une fois l'environnement résilié.
6. Choisissez Apply.

Après avoir activé la diffusion de journaux, vous pouvez revenir à la page ou à la catégorie de configuration Logiciels pour trouver le lien Groupes de journaux. Cliquez sur ce lien pour afficher vos journaux dans la console CloudWatch.

## Diffusion de journaux d'instance à l'aide de l'interface de ligne de commande EB

Pour activer la diffusion des journaux d'instance vers CloudWatch Logs à l'aide de l'interface de ligne de commande EB, utilisez la commande [eb logs \(p. 1100\)](#).

```
$ eb logs --cloudwatch-logs enable
```

Vous pouvez également utiliser la commande eb logs pour extraire des journaux à partir de CloudWatch Logs. Vous pouvez récupérer tous les journaux d'instance de l'environnement ou utiliser les nombreuses options de la commande pour spécifier des sous-ensembles de journaux à extraire. Par exemple, la commande suivante extrait l'ensemble complet des journaux d'instance de votre environnement et les enregistre dans un répertoire sous .elasticbeanstalk/logs.

```
$ eb logs --all
```

En particulier, l'option --log-group vous permet d'extraire les journaux d'instance d'un groupe de journaux spécifique, correspondant à un fichier journal d'instance spécifique. Pour ce faire, vous devez

connaître le nom du groupe de journaux correspondant au fichier journal que vous souhaitez récupérer. Vous pouvez trouver ces informations dans [Méthode de configuration de CloudWatch Logs par Elastic Beanstalk \(p. 898\)](#).

## Diffusion des journaux d'instance à l'aide de fichiers de configuration

Lorsque vous créez ou mettez à jour un environnement, vous pouvez utiliser un fichier de configuration pour installer et configurer la diffusion des journaux d'instance vers CloudWatch Logs. L'exemple de fichier de configuration suivant active la diffusion des journaux d'instance par défaut. Elastic Beanstalk diffuse l'ensemble de fichiers journaux par défaut correspondant à la plateforme de votre environnement. Pour utiliser l'exemple, copiez le texte dans un fichier avec l'extension `.config` dans le répertoire `.ebextensions` au niveau supérieur du bundle de fichiers source de votre application.

```
option_settings:  
  - namespace: aws:elasticbeanstalk:cloudwatch:logs  
    option_name: StreamLogs  
    value: true
```

## Diffusion de fichiers journaux personnalisés

L'intégration d'Elastic Beanstalk à CloudWatch Logs ne prend pas directement en charge la diffusion des fichiers journaux personnalisés générés par votre application. Pour diffuser des journaux personnalisés, utilisez un fichier de configuration pour installer directement l'agent CloudWatch Logs et configurer les fichiers à transmettre. Pour obtenir un exemple de fichier de configuration, veuillez consulter [logs-streamtocloudwatch-linux.config](#).

### Note

L'exemple ne fonctionne pas sur la plateforme Windows.

Pour de plus amples informations sur la configuration de CloudWatch Logs, veuillez consulter la [Référence de l'agent CloudWatch Logs](#) dans le Guide de l'utilisateur Amazon CloudWatch Logs.

## Résolution des problèmes liés à l'intégration de CloudWatch Logs

Si vous ne trouvez pas certains des journaux d'instance de l'environnement que vous recherchez dans CloudWatch Logs, passez en revue les problèmes courants suivants :

- Votre rôle IAM ne dispose pas des autorisations IAM requises.
- Vous avez lancé votre environnement dans une Région AWS qui ne prend pas en charge CloudWatch Logs.
- L'un de vos fichiers journaux personnalisés n'existe pas dans le chemin que vous avez spécifié.

## Diffusion d'informations sur l'état de l'environnement Elastic Beanstalk vers Amazon CloudWatch Logs

Si vous activez les [rapports améliorés sur l'état \(p. 837\)](#) pour votre environnement, vous pouvez configurer ce dernier de sorte à diffuser des informations sur l'état vers CloudWatch Logs. Cette diffusion ne dépend

pas de la diffusion des journaux d'instance Amazon EC2. Cette rubrique décrit la diffusion des informations d'intégrité d'environnement. Pour plus d'informations sur la diffusion des journaux d'instance, consultez [Utilisation d'Elastic Beanstalk avec Amazon CloudWatch Logs \(p. 895\)](#).

Lorsque vous configurez la diffusion des informations sur l'état de l'environnement, Elastic Beanstalk crée un groupe de journaux CloudWatch Logs pour l'état de l'environnement. Le nom de ce groupe de journaux est `/aws/elasticbeanstalk/environment-name/environment-health.log`. Dans ce groupe de journaux, Elastic Beanstalk crée des flux de journaux nommés `YYYY-MM-DD#<hash-suffix>` (il peut y avoir plusieurs flux de journaux par date).

Lorsque l'état de santé de l'environnement change, Elastic Beanstalk ajoute un enregistrement au flux de journaux d'état. L'enregistrement représente la transition d'état d'intégrité, c'est-à-dire le nouveau statut et une description de la cause du changement. Par exemple, un statut d'environnement peut passer à Grave parce que l'équilibrer de charge est en panne. Pour obtenir une description des statuts d'intégrité améliorée, consultez [Couleurs et états utilisés dans les rapports d'intégrité \(p. 854\)](#).

## Conditions préalables pour la diffusion des informations sur l'état d'un environnement vers CloudWatch Logs

Pour activer la diffusion des informations sur l'état d'un environnement vers CloudWatch Logs, vous devez respecter les conditions suivantes :

- Plateforme – Vous devez utiliser une version de plateforme qui prend en charge les rapports améliorés sur l'état.
- Autorisations – Vous devez accorder certaines autorisations de journalisation à Elastic Beanstalk afin qu'il puisse agir en votre nom pour diffuser les informations sur l'état de votre environnement. Si votre environnement n'utilise pas un rôle de service créé pour lui par Elastic Beanstalk, `aws-elasticbeanstalk-service-role` ou le rôle lié à un service de votre compte, `AWSServiceRoleForElasticBeanstalk`, veillez à ajouter les autorisations suivantes à votre rôle de service personnalisé.

```
{  
    "Effect": "Allow",  
    "Action": [  
        "logs:DescribeLogStreams",  
        "logs>CreateLogStream",  
        "logs:PutLogEvents"  
    ],  
    "Resource": "arn:aws:logs:*::log-group:/aws/elasticbeanstalk/*:log-stream:*"  
}
```

## Diffusion des journaux d'état de l'environnement vers CloudWatch Logs

Vous pouvez activer la diffusion des informations d'état d'un environnement vers CloudWatch Logs à l'aide de la console Elastic Beanstalk, de l'interface de ligne de commande EB ou des options de configuration.

### Diffusion des journaux d'état de l'environnement à l'aide de la console Elastic Beanstalk

Pour diffuser les journaux d'état de l'environnement vers CloudWatch Logs

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Monitoring (Surveillance), choisissez Edit (Modifier).
5. Sous Rapports sur l'état de santé, assurez-vous que l'option de rapports Système est définie sur Amélioré.
6. Sous Health event streaming to CloudWatch Logs (Diffusion d'événements de santé vers CloudWatch Logs)
  - Activez Diffusion des journaux.
  - Définissez Conservation sur le nombre de jours de conservation des journaux.
  - Sélectionnez le paramètre de cycle de vie qui détermine si les journaux sont enregistrés une fois l'environnement résilié.
7. Choisissez Apply.

Après avoir activé la diffusion de journaux, vous pouvez revenir à la page ou à la catégorie de configuration Surveillance pour trouver le lien Groupe de journaux. Cliquez sur ce lien pour afficher les journaux d'état de votre environnement dans la console CloudWatch.

## Diffusion des journaux d'intégrité d'environnement à l'aide de l'interface de ligne de commande EB

Pour activer la diffusion des journaux d'état de l'environnement vers CloudWatch Logs à l'aide de l'interface de ligne de commande EB, utilisez la commande [eb logs \(p. 1100\)](#).

```
$ eb logs --cloudwatch-logs enable --cloudwatch-log-source environment-health
```

Vous pouvez également utiliser la commande eb logs pour extraire des journaux à partir de CloudWatch Logs. Par exemple, la commande suivante extrait l'ensemble des journaux d'intégrité de votre environnement et les enregistre dans un répertoire sous .elasticbeanstalk/logs.

```
$ eb logs --all --cloudwatch-log-source environment-health
```

## Diffusion des journaux d'intégrité d'environnement à l'aide de fichiers de configuration

Lorsque vous créez ou mettez à jour un environnement, vous pouvez utiliser un fichier de configuration pour installer et configurer la diffusion des journaux d'état de l'environnement vers CloudWatch Logs. Pour utiliser l'exemple ci-dessous, copiez le texte dans un fichier avec l'extension .config dans le répertoire .ebextensions au niveau supérieur du bundle de fichiers source de votre application. Cet exemple configure Elastic Beanstalk pour activer la diffusion des journaux d'état de l'environnement, conserver les journaux après l'arrêt de l'environnement et les enregistrer pendant 30 jours.

### Example Fichier de configuration de diffusion des journaux sur l'état de santé

```
#####
## Sets up Elastic Beanstalk to stream environment health information
```

```
## to Amazon CloudWatch Logs.  
## Works only for environments that have enhanced health reporting enabled.  
#####
option_settings:  
    aws:elasticbeanstalk:cloudwatch:logs:health:  
        HealthStreamingEnabled: true  
        ##### Settings below this line are optional.  
        # DeleteOnTerminate: Delete the log group when the environment is  
        # terminated. Default is false. If false, the health data is kept  
        # RetentionInDays days.  
        DeleteOnTerminate: false  
        # RetentionInDays: The number of days to keep the archived health data  
        # before it expires, if DeleteOnTerminate isn't set. Default is 7 days.  
        RetentionInDays: 30
```

Pour obtenir les valeurs par défaut et les valeurs valides des options, consultez

[aws:elasticbeanstalk:cloudwatch:logs:health \(p. 698\)](#).

## Utilisation d'Elastic Beanstalk avec Amazon EventBridge

Avec Amazon EventBridge, vous pouvez configurer des règles basées sur des événements qui surveillent vos ressources Elastic Beanstalk et lancer des actions ciblées utilisant d'autres services AWS. Par exemple, vous pouvez définir une règle pour l'envoi de notifications par e-mail en signalant une rubrique Amazon SNS chaque fois que l'état de santé d'un environnement de production passe à Warning (Avertissement). Vous pouvez également définir une fonction Lambda pour transmettre une notification à Slack chaque fois que l'état de santé de votre environnement passe à Degraded (Dégradé) ou à Severe (Grave).

Vous pouvez créer des règles dans Amazon EventBridge pour agir sur l'un des événements Elastic Beanstalk suivants :

- Changements d'état des opérations d'environnement (y compris les opérations de création, de mise à jour et de résiliation). L'événement indique si le changement d'état a démarré, réussi ou échoué.
- Changements d'état pour d'autres ressources. Outre les environnements, les autres ressources surveillées incluent les équilibreurs de charge, les groupes Auto Scaling et les instances.
- Transition d'état pour les environnements. L'événement indique l'endroit où l'état de l'environnement est passé d'un état de santé à un autre.
- Changement d'état pour les mises à jour gérées. L'événement indique si le changement d'état a démarré, réussi ou échoué.

Pour capturer des événements Elastic Beanstalk spécifiques qui vous intéressent, définissez des modèles spécifiques à ces événements, qu'EventBridge peut utiliser pour détecter les événements. Les modèles d'événement ont la même structure que les événements auxquels ils correspondent. Le modèle place entre guillemets les champs que vous voulez faire correspondre et fournit les valeurs que vous recherchez. Les événements sont générés sur la base du meilleur effort. Ils sont remis à EventBridge par Elastic Beanstalk en temps quasi réel dans des conditions opérationnelles normales. Toutefois, certaines situations peuvent retarder ou empêcher la livraison d'un événement.

Pour obtenir la liste des champs contenus dans les événements Elastic Beanstalk et leurs valeurs de chaîne possibles, veuillez consulter [Mappage des champs d'événement Elastic Beanstalk \(p. 911\)](#).

Pour de plus amples informations sur le fonctionnement des règles EventBridge avec les modèles d'événements, veuillez consulter [Événements et modèles d'événements dans EventBridge](#).

## Surveiller une ressource Elastic Beanstalk avec EventBridge

Avec EventBridge, vous pouvez créer des règles définissant les mesures à prendre lorsqu'Elastic Beanstalk émet des événements pour ses ressources. Par exemple, vous pouvez créer une règle qui vous envoie un e-mail chaque fois que l'état d'un environnement change.

La console EventBridge dispose d'une option de Pre-defined pattern (Motif prédéfini) pour créer des modèles d'événements Elastic Beanstalk. Si vous sélectionnez cette option dans la console EventBridge lorsque vous créez une règle, vous pouvez créer rapidement un modèle d'événement Elastic Beanstalk. Il suffit de sélectionner les champs et les valeurs de l'événement. Au fur et à mesure que vous effectuez des sélections, la console crée et affiche le modèle d'événement. Vous pouvez également modifier manuellement le modèle d'événement que vous créez et pouvez l'enregistrer en tant que modèle personnalisé. La console affiche également un Sample Event (Exemple d'événement) détaillé que vous pouvez copier et coller dans le modèle d'événement que vous créez.

Si vous préférez saisir ou copier et coller un modèle d'événement dans la console EventBridge, vous pouvez choisir d'utiliser l'option Custom pattern (Motif personnalisé) dans la console. Ce faisant, vous n'avez pas besoin de suivre les étapes de sélection des champs et des valeurs décrites précédemment. Cette rubrique propose des exemples de [modèles de correspondance d'événements \(p. 908\)](#) et [d'événements Elastic Beanstalk \(p. 910\)](#) que vous pouvez utiliser.

Pour créer une règle pour un événement de ressource

1. Connectez-vous à AWS à l'aide d'un compte autorisé à utiliser EventBridge et Elastic Beanstalk.
2. Ouvrez la console Amazon EventBridge à l'adresse <https://console.aws.amazon.com/events/>.
3. Choisissez Create rule.
4. Entrez un nom et éventuellement une description pour la règle.
5. Pour Define pattern (Définir un modèle), choisissez Event pattern (Modèle d'événement).
6. Sous Event matching pattern (Modèle de correspondance d'événement), sélectionnez Pre-defined pattern by service (Modèle prédéfini par un service).

### Note

Si vous avez déjà du texte pour un modèle d'événement et que vous n'avez pas besoin que la console EventBridge le crée pour vous, vous pouvez sélectionner Customer pattern (Modèle client). Vous pouvez ensuite saisir ou copier et coller manuellement du texte dans la zone Event Pattern (Modèle d'événement). Sélectionnez Save (Enregistrer), puis passez à l'étape 12.

7. Sélectionnez AWS pour Service Provider (Fournisseur de services).
8. Pour Service name (Nom du service), sélectionnez Elastic Beanstalk.
9. Pour Event type (Type d'événement), sélectionnez Statut Change (Changement de statut).
10. Cette étape explique comment utiliser les champs d'événement detail type (type de détail), statut (état) et severity (gravité) pour Elastic Beanstalk. Au fur et à mesure que vous sélectionnez ces champs et les valeurs de votre choix, la console crée et affiche le modèle d'événement. Vos choix déterminent également comment la console affiche les champs d'événements suivants, en fonction de la hiérarchie des champs et du regroupement de leurs valeurs. Pour plus d'informations, consultez la table [Mappage des champs d'événement Elastic Beanstalk \(p. 911\)](#).
  - Si vous sélectionnez une seule valeur spécifique pour un champ donné, le champ suivant de la hiérarchie s'affiche dans la console. Vous pouvez sélectionner une ou plusieurs valeurs avec le champ affiché. Par exemple, si vous sélectionnez la case d'option de Specific detail type(s) (Type(s) de détail spécifique(s)), la liste déroulante detail types (types de détails) s'affiche. Si vous ne

sélectionnez qu'une seule valeur dans cette liste, les options qui vous permettent d'utiliser le champ suivant s'affichent dans la console. Dans cet exemple, le champ d'état suit. Pour ce champ, vous pouvez sélectionnez Any status (N'importe quel statut) ou Specific status(es) (Statut(s) spécifique(s)).

- Si vous sélectionnez plusieurs valeurs pour un champ donné ou si vous sélectionnez la case d'option Any (N'importe quel) pour ce champ (par exemple, Any status (N'importe quel statut)), la console ne fournit pas d'options pour les champs suivants. Plus précisément, elle ne vous permet pas de choisir des valeurs spécifiques pour les champs suivants de la hiérarchie. Dans cet exemple, le champ severity (gravité) suit le champ status (statut) dans la hiérarchie. La valeur par défaut est Any severity (N'importe quelle gravité) si plus d'un statut est choisi au préalable. Si vous sélectionnez Specific severity(s) (Gravité(s) spécifique(s)), aucun élément n'est répertorié dans la liste de valeurs. La console est conçue de cette façon pour empêcher d'avoir une logique de correspondance ambiguë entre les champs de votre modèle d'événement.

Le champ d'événement environment (environnement) n'est pas affecté par cette hiérarchie, il s'affiche comme décrit à l'étape suivante.

11. Pour Environment (Environnement), sélectionnez Any environment (N'importe quel environnement) ou Specific environment(s) (Environnement(s) spécifique(s)).
  - Si vous sélectionnez Specific environment(s) (Environnement(s) spécifique(s)), vous pouvez sélectionner un ou plusieurs environnements dans la liste déroulante. EventBridge ajoute tous les environnements que vous sélectionnez dans la liste EnvironmentName[ ] de la section detail (détails) du modèle d'événement. Ensuite, votre règle filtre tous les événements pour inclure uniquement les environnements spécifiques que vous sélectionnez.
  - Si vous sélectionnez Any environment (N'importe quel environnements), aucun environnement n'est ajouté à votre modèle d'événement. Pour cette raison, votre règle ne filtre aucun des événements Elastic Beanstalk en se basant sur l'environnement.
12. La section Select event bus (Sélectionner un bus d'événements) est AWS default event bus (bus d'événement AWS par défaut). Laissez la valeur par défaut sélectionnée et vérifiez que l'option Enable the rule on the selected event bus (Activer la règle sur le bus d'événements sélectionné) est activée.
13. Pour Select targets (Sélectionner les cibles), choisissez l'action cible à effectuer lorsqu'un événement de changement d'état de ressource est reçu de la part d'Elastic Beanstalk.

Par exemple, vous pouvez utiliser une rubrique Amazon Simple Notification Service (SNS) pour envoyer un e-mail ou un SMS lorsqu'un événement se produit. Pour ce faire, vous devez créer une rubrique Amazon SNS à l'aide de la console Amazon SNS. Pour en savoir plus, veuillez consulter [Utilisation d'Amazon SNS pour les notifications utilisateur](#).

#### Important

Certaines actions cibles peuvent nécessiter l'utilisation d'autres services entraînant des frais supplémentaires, tels que le service Amazon SNS ou Lambda. Pour de plus amples informations sur la tarification AWS, veuillez consulter <http://aws.amazon.com/pricing/>. Certains services font partie du niveau d'offre gratuite d'AWS. Si vous êtes un nouveau client, vous pouvez essayer ces services gratuitement. Pour de plus amples informations, veuillez consulter <http://aws.amazon.com/free/>.

14. (Facultatif) Sélectionnez Add target (Ajouter une cible) pour spécifier une action cible supplémentaire pour la règle d'événement.
15. Sélectionnez Créer.

## Exemple de modèles d'événements Elastic Beanstalk

Les modèles d'événement ont la même structure que les événements auxquels ils correspondent. Le modèle place entre guillemets les champs que vous voulez faire correspondre et fournit les valeurs que vous recherchez.

- Health status change (Changement de l'état de santé) pour tous les environnements

```
{  
    "source": [  
        "aws.elasticbeanstalk"  
    ],  
    "detail-type": [  
        "Health status change"  
    ]  
}
```

- Health status change (Changement de l'état de santé) pour les environnements suivants : `myEnvironment1` et `myEnvironment2`. Ce modèle d'événement filtre pour ces deux environnements spécifiques, tandis que l'exemple précédent de Health status change (Changement de l'état de santé) qui ne filtre pas envoie des événements pour tous les environnements.

```
{"source": [  
    "aws.elasticbeanstalk"  
],  
    "detail-type": [  
        "Health status change"  
    ],  
    "detail": {  
        "EnvironmentName": [  
            "myEnvironment1",  
            "myEnvironment2"  
        ]  
    }  
}
```

- Elastic Beanstalk resource status change (Changement de l'état de santé des ressources Elastic Beanstalk) pour tous les environnements

```
{  
    "source": [  
        "aws.elasticbeanstalk"  
    ],  
    "detail-type": [  
        "Elastic Beanstalk resource status change"  
    ]  
}
```

- Elastic Beanstalk resource status change (Changement de l'état de ressource Elastic Beanstalk) avec `Status Environment update failed` (Échec de la mise à jour d'environnement) et `Severity ERROR` (ERREUR) pour les environnements suivants : `myEnvironment1` et `myEnvironment2`

```
{"source": [  
    "aws.elasticbeanstalk"  
],  
    "detail-type": [  
        "Elastic Beanstalk resource status change"  
    ],  
    "detail": {  
        "Status": [  
            "Environment update failed"  
        ],  
        "Severity": [  
            "ERROR"  
        ],  
        "EnvironmentName": [  
            "myEnvironment1",  
            "myEnvironment2"  
        ]  
    }  
}
```

```
        ]  
    }  
}
```

- Other resource status change (Autre changement d'état des ressources) pour les équilibreurs de charge, les groupes Auto Scaling et les instances

```
{  
    "source": [  
        "aws.elasticbeanstalk"  
    ],  
    "detail-type": [  
        "Other resource status change"  
    ]  
}
```

- Managed update status change (Changement d'état des mises à jour gérées) pour tous les environnements

```
{  
    "source": [  
        "aws.elasticbeanstalk"  
    ],  
    "detail-type": [  
        "Managed update status change"  
    ]  
}
```

- Pour capturer tous les événements à partir d'Elastic Beanstalk, excluez la section detail-type :

```
{  
    "source": [  
        "aws.elasticbeanstalk"  
    ]  
}
```

## Exemple d'événement Elastic Beanstalk

Voici un exemple d'événement Elastic Beanstalk pour un changement d'état de ressource :

```
{  
    "version": "0",  
    "id": "1234a678-1b23-c123-12fd3f456e78",  
    "detail-type": "Elastic Beanstalk resource status change",  
    "source": "aws.elasticbeanstalk",  
    "account": "111122223333",  
    "time": "2020-11-03T00:31:54Z",  
    "region": "us-east-1",  
    "resources": [  
        "arn:aws:elasticbeanstalk:us-east-1:111122223333:environment/myApplication/  
myEnvironment"  
    ],  
    "detail": {  
        "Status": "Environment creation started",  
        "EventDate": 1604363513951,  
        "ApplicationName": "myApplication",  
        "Message": "createEnvironment is starting.",  
        "EnvironmentName": "myEnvironment",  
        "Severity": "INFO"  
    }  
}
```

}

Voici un exemple d'événement Elastic Beanstalk pour un changement d'état de santé :

```
{
    "version": "0",
    "id": "1234a678-1b23-c123-12fd3f456e78",
    "detail-type": "Health status change",
    "source": "aws.elasticbeanstalk",
    "account": "111122223333",
    "time": "2020-11-03T00:34:48Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:elasticbeanstalk:us-east-1:111122223333:environment/myApplication/
myEnvironment"
    ],
    "detail": {
        "Status": "Environment health changed",
        "EventDate": 1604363687870,
        "ApplicationName": "myApplication",
        "Message": "Environment health has transitioned from Pending to Ok. Initialization completed 1 second ago and took 2 minutes.",
        "EnvironmentName": "myEnvironment",
        "Severity": "INFO"
    }
}
```

## Mappe des champs d'événement Elastic Beanstalk

Le tableau suivant mappe les champs d'événement Elastic Beanstalk et leurs valeurs de chaîne possibles au champ EventBridge detail-type. Pour de plus amples informations sur le fonctionnement d'EventBridge avec les modèles d'événements pour un service, veuillez consulter [Événements et modèles d'événements dans EventBridge](#).

Champ EventBridge detail-type	Champ Elastic Beanstalk Status	Champ Elastic Beanstalk Severity	Champ Elastic Beanstalk Message
Elastic Beanstalk resource status change	Environment creation started	INFO	createEnvironment is starting.
	Environment creation successful	INFO	createEnvironment completed successfully.
	Environment creation successful	INFO	Launched environment: <Environment Name>. However, there were issues during launch. See event log for details.
	Environment creation failed	ERROR	Failed to launch environment.
	Environment update started	INFO	Environment update is starting.

Champ EventBridge detail-type	Champ Elastic Beanstalk Status	Champ Elastic Beanstalk Severity	Champ Elastic Beanstalk Message
	Environment update successful	INFO	Environment update completed successfully.
	Environment update failed	ERROR	Failed to deploy configuration.
	Environment termination started	INFO	terminateEnvironment is starting.
	Environment termination successful	INFO	terminateEnvironment completed successfully.
	Environment termination failed	INFO	The environment termination step failed because at least one of the environment termination workflows failed.
Other resource status change	Auto Scaling group created	INFO	createEnvironment is starting.
	Auto Scaling group deleted	INFO	createEnvironment is starting.
	Instance added	INFO	Added instance [i-123456789a12b1234] to your environment.
	Instance removed	INFO	Removed instance [i-123456789a12b1234] from your environment.
	Load balancer created	INFO	Created load balancer named: <LB Name>
	Load balancer deleted	INFO	Deleted load balancer named: <LB Name>
Health status change	Environment health changed	INFO/WARN	Environment health has transitioned to <healthStatus>.
	Environment health changed	INFO/WARN	Environment health has transitioned from <healthStatus> to <healthStatus>.
Managed update status change	Managed update started	INFO	Managed platform update is in-progress.
	Managed update failed	INFO	Managed update failed, retrying in %s minutes.

# Recherche et suivi des ressources Elastic Beanstalk avec AWS Config

[AWS Config](#) fournit une vue détaillée de la configuration des ressources AWS de votre compte AWS. Vous pouvez voir comment les ressources sont liées, obtenir un historique des changements de configuration, et voir comment les configurations et les relations changent au fil du temps. Vous pouvez utiliser AWS Config pour définir des règles permettant d'évaluer les configurations de ressource en termes de conformité des données.

Plusieurs types de ressources Elastic Beanstalk sont intégrés à AWS Config:

- Applications
- Versions de l'application
- Environnements

La section suivante montre comment configurer AWS Config pour enregistrer les ressources de ce type.

Pour plus d'informations sur AWS Config, consultez le [Guide du développeur AWS Config](#). Pour obtenir des informations sur la tarification, consultez la [page des informations de tarification d'AWS Config](#).

## Configuration d'AWS Config

Pour configurer initialement AWS Config, veuillez consulter les rubriques suivantes dans le [Guide du développeur AWS Config](#).

- [Configuration d'AWS Config avec la console](#)
- [Configuration de AWS Config avec la AWS CLI](#)

## Configuration de AWS Config pour enregistrer les ressources Elastic Beanstalk

Par défaut, AWS Config enregistre les changements de configuration pour tous les types de ressources régionales qu'il détecte dans la région dans laquelle votre environnement s'exécute. Vous pouvez personnaliser AWS Config pour enregistrer uniquement les modifications de types de ressources spécifiques ou les modifications apportées aux ressources globales.

Par exemple, vous pouvez configurer AWS Config pour enregistrer les modifications pour les ressources Elastic Beanstalk et un sous-ensemble d'autres ressources AWS démarré par Elastic Beanstalk pour vous. À l'aide de la [console AWS Config](#), vous pouvez sélectionner Elastic Beanstalk comme ressource dans la page Paramètres AWS Config dans le champ Types spécifiques. À partir de là, vous pouvez choisir d'enregistrer n'importe quel type de ressource Elastic Beanstalk : Application, ApplicationVersion et Environnement.

La figure suivante montre la page Settings (Paramètres) d'AWS Config, avec les types de ressources Elastic Beanstalk que vous pouvez choisir d'enregistrer : Application, ApplicationVersion (Version de l'application) et Environment (Environnement).

The screenshot shows the 'Settings' page in the AWS Config console. At the top, it says 'Recording is on' with a 'Turn off' button. Below that, under 'Resource types to record', it says 'Select the types of AWS resources for which you want AWS Config to record configuration changes. By default, AWS Config records configuration changes for all supported resources. You can also choose to record configuration changes for supported global resources in this region.' There are two main sections: 'All resources' and 'Specific types'. Under 'All resources', there are two checkboxes: 'Record all resources supported in this region' and 'Include global resources (e.g., AWS IAM resources)'. Under 'Specific types', there is a dropdown menu with 'ElasticBeanstalk' selected. A sub-menu for 'ElasticBeanstalk' is open, showing 'Application' (which is highlighted in yellow), 'ApplicationVersion', and 'Environment'. Other options like 'Amazon S3 bucket' and 'Your bucket receives configuration records' are also listed. At the bottom, there is a 'Create a bucket' button.

Une fois que vous avez sélectionné quelques types de ressources, voici à quoi ressemble la liste Specific types (Types spécifiques).

The screenshot shows the 'Settings' page in the AWS Config console. It displays the 'Resource types to record' section with the same instructions as the previous screenshot. Under 'Specific types', there is a list of selected resources: 'ElasticBeanstalk: Application', 'ElasticBeanstalk: Environment', and 'EC2: Instance'. Each item has a small 'x' icon to its right.

Pour en savoir plus sur les ressources régionales et globales, et sur la procédure de personnalisation complète, consultez [Sélection des ressources enregistrées par AWS Config](#).

## Affichage des détails de configuration Elastic Beanstalk dans la console AWS Config

Vous pouvez utiliser la console AWS Config pour rechercher des ressources Elastic Beanstalk et obtenir des détails actuels et historiques sur leurs configurations. L'exemple suivant montre comment rechercher des informations sur un environnement Elastic Beanstalk.

Pour trouver un environnement Elastic Beanstalk dans la console AWS Config

1. Ouvrez la [console AWS Config](#).
2. Choisissez Ressources.
3. Sur la page Resource Inventory (Inventaire des ressources), choisissez Resources (Ressources).
4. Ouvrez le menu Resource type (Type de ressource), faites-le défiler jusqu'à ElasticBeanstalk, puis choisissez un ou plusieurs types de ressources Elastic Beanstalk.

Note

Pour afficher les détails de configuration d'autres ressources créées par Elastic Beanstalk pour votre application, choisissez des types de ressources supplémentaires. Par exemple, vous pouvez choisir Instance sous EC2.

5. Choisissez Recherche. Voir 2 dans la figure suivante.

The screenshot shows the AWS Config Resource inventory interface. On the left, there is a dropdown menu with the following options: VPNGateway, Volume, **ElasticBeanstalk**, Application, ApplicationVersion, Environment, ElasticLoadBalancing, LoadBalancer, ElasticLoadBalancingV2, LoadBalancer, and ... . The 'Environment' option is highlighted with a yellow box and has a yellow circle with the number '1' above it. To the right of the dropdown is a search bar labeled 'Resource identifier (optional)'. Below the search bar is a blue 'Look up' button. A yellow speech bubble with the number '2' is pointing to the 'Look up' button. At the bottom of the interface, there is a table with columns: Config timeline, Compliance, and Manage resource. The table contains several rows of resource data.

6. Choisissez un ID de ressource dans la liste des ressources affichée par AWS Config.

The screenshot shows the AWS Config Resource inventory interface. On the left, there is a dropdown menu with the following options: EC2 Instance, **ElasticBeanstalk Application**, and **ElasticBeanstalk Environment**. The 'ElasticBeanstalk Environment' option is highlighted with a yellow box. To the right of the dropdown is a search bar labeled 'Resource identifier (optional)'. Below the search bar is a checkbox labeled 'Include deleted resources'. At the bottom of the interface, there is a table with columns: Resource type, Config timeline, Compliance, and Manage resource. The table contains three rows of resource data, with the last row ('ElasticBeanstalk Environment') highlighted with a yellow box.

AWS Config affiche les détails de configuration et d'autres informations sur la ressource que vous avez sélectionnée.

AWS Elastic Beanstalk Manuel du développeur  
Affichage des détails de configuration Elastic  
Beanstalk dans la console AWS Config

ElasticBeanstalk Environment e-yaumygtbwr

on February 09, 2018 4:03:54 PM Pacific Standard Time (UTC-08:00)

Manage resource

Now

05<sup>th</sup> February 2018 4:34:35 PM 1 Changes

06<sup>th</sup> February 2018 3:43:45 PM 7 Changes

07<sup>th</sup> February 2018 11:43:44 PM 7 Changes

View Details

▼ Configuration Details

Amazon Resource Name arn:aws:elasticbeanstalk:us-east-1:270205402845:environment/config-demo/ConfigDemo-env

Resource type AWS::ElasticBeanstalk::Environment

Resource ID e-yaumygtbwr

Resource name ConfigDemo-env

Availability zone Not Applicable

Created on February 05, 2018 3:45:05 PM

Tags (3) elasticbeanstalk:envi... elasticbeanstalk:envi... Name:ConfigDemo-...

▶ Relationships 5

▶ Changes 7

▶ CloudTrail Events 0

Pour voir tous les détails de la configuration enregistrée, choisissez View Details (Afficher les détails).

Show JSON

Now

View Details

Close

Object

version: "1.3"

accountId: "270205402845"

configurationItemCaptureTime: "2018-02-08T07:43:44.957Z"

configurationItemStatus: "OK"

configurationStateId: "1518075824957"

configurationItemMD5Hash: ""

arn:

"arn:aws:elasticbeanstalk:us-east-1:270205402845:environment/config-demo/ConfigDemo-env"

resourceType: "AWS::ElasticBeanstalk::Environment"

resourceId: "e-yaumygtbwr"

resourceName: "ConfigDemo-env"

awsRegion: "us-east-1"

availabilityZone: "Not Applicable"

resourceCreationTime: "2018-02-05T23:45:05.861Z"

tags: Object

elasticbeanstalk:environment-name: "ConfigDemo-env"

Pour en savoir plus sur d'autres manières de trouver une ressource et d'afficher des informations sur cette page, veuillez consulter [Affichage des configurations et de l'historique des ressources AWS](#) dans le Guide du développeur AWS Config .

## Évaluation des ressources Elastic Beanstalk à l'aide des règles AWS Config

Vous pouvez créer des règles AWS Config, qui représentent les paramètres de configuration idéaux pour vos ressources Elastic Beanstalk. Vous pouvez utiliser des règles de configuration gérées par AWS prédéfinies ou définir des règles personnalisées. AWS Config surveille en permanence les changements apportés à la configuration de vos ressources afin de déterminer si ces changements ne vont pas à l'encontre de l'une des conditions de vos règles. La console AWS Config affiche l'état de conformité de vos règles et ressources.

Si une ressource enfreint une règle et est signalée comme non conforme, AWS Config peut vous avertir à l'aide d'une rubrique [Amazon Simple Notification Service \(Amazon SNS\)](#). Pour programmer la consommation des données contenues dans ces alertes AWS Config, utilisez une file d'attente [Amazon Simple Queue Service \(Amazon SQS\)](#) comme point de terminaison de notification pour la rubrique Amazon SNS. Par exemple, vous pouvez écrire du code qui démarre un flux de travail lorsque quelqu'un modifie la configuration de groupe Auto Scaling de votre environnement.

Pour en savoir plus sur la configuration et l'utilisation de règles, veuillez consulter [Évaluation des ressources à l'aide de règles AWS Config](#) dans le Guide du développeur AWS Config .

## Utilisation d'Elastic Beanstalk avec Amazon DynamoDB

Amazon DynamoDB est un service de base de données NoSQL entièrement géré, offrant des performances exceptionnelles et prévisibles en termes de rapidité et d'évolutivité. Si vous êtes développeur, vous pouvez utiliser DynamoDB pour créer une table de base de données pouvant stocker et récupérer n'importe quel volume de données, et traiter tout niveau de trafic de demandes. DynamoDB répartit automatiquement les données et le trafic de la table sur un nombre suffisant de serveurs afin de gérer la capacité de demandes spécifiée par le client et le volume de données stockées, tout en assurant la cohérence et la rapidité des performances. Enregistrés sur des disques SSD (Solid State Drives), tous les éléments de données sont répliqués automatiquement sur plusieurs zones de disponibilité d'une région AWS, assurant ainsi la haute disponibilité et la durabilité des données.

Si vous utilisez des [tâches périodiques \(p. 525\)](#) dans un environnement de travail, Elastic Beanstalk crée une table DynamoDB et l'utilise pour effectuer l'élection de leader et stocker des informations sur la tâche. Chaque instance de l'environnement tente d'écrire vers la table à intervalles réguliers de quelques secondes pour devenir leader et effectuer la tâche lorsqu'elle est prévue.

Vous pouvez utiliser des [fichiers de configuration \(p. 737\)](#) pour créer une table DynamoDB pour votre application. Consultez la page [eb-node-express-sample](#) sur GitHub pour obtenir un exemple d'application Node.js qui crée une table avec un fichier de configuration et s'y connecte avec le AWS kit SDK pour JavaScript dans Node.js. Pour obtenir un exemple de procédure d'utilisation de DynamoDB avec PHP, consultez [Exemple : DynamoDB, CloudWatch et SNS \(p. 775\)](#). Pour obtenir un exemple qui utilise le AWS SDK for Java, consultez la page relative à la [gestion de l'état de session Tomcat avec DynamoDB AWS SDK for Java](#) dans la documentation .

Lorsque vous créez une table DynamoDB à l'aide de fichiers de configuration, cette table n'est pas rattachée au cycle de vie de votre environnement et n'est pas supprimée lorsque vous arrêtez votre environnement. Pour être sûr que des informations personnelles ne sont pas inutilement conservées, supprimez ces enregistrements lorsque vous n'en avez plus besoin ou supprimez la table.

Pour de plus amples informations sur DynamoDB, consultez le [Manuel du développeur DynamoDB](#).

## Utilisation d'Elastic Beanstalk avec Amazon ElastiCache

Amazon ElastiCache est un service web qui permet de configurer, gérer et mettre à l'échelle des environnements de cache en mémoire distribués dans le cloud. Il fournit un cache en mémoire haute performance, évolutif et économique, tout en éliminant la complexité associée au déploiement et à la gestion d'un environnement de cache distribué. ElastiCache est conforme au protocole avec Redis et Memcached, donc le code, les applications et les outils les plus courants que vous utilisez aujourd'hui avec vos environnements Redis et Memcached existants fonctionneront de manière transparente avec le service. Pour de plus amples informations sur ElastiCache, veuillez consulter la page produit [Amazon ElastiCache](#).

Pour utiliser Elastic Beanstalk avec Amazon ElastiCache

1. Créez un cluster ElastiCache.
  - Pour obtenir des instructions sur la création d'un cluster ElastiCache avec Redis, veuillez consulter [Mise en route avec Amazon ElastiCache for Redis](#) dans le Guide de l'utilisateur ElastiCache pour Redis.
  - Pour obtenir des instructions sur la création d'un cluster ElastiCache avec Memcached, veuillez consulter [Mise en route avec Amazon ElastiCache for Memcached](#) dans le Guide de l'utilisateur d'ElastiCache pour Memcached.
2. Configurez votre groupe de sécurité ElastiCache pour autoriser l'accès à partir du groupe de sécurité Amazon EC2 utilisé par votre application Elastic Beanstalk. Pour obtenir des instructions sur la façon de trouver le nom de votre groupe de sécurité EC2 à l'aide de la console de gestion AWS, veuillez consulter [Groupes de sécurité \(p. 542\)](#) sur la page du document Instances EC2.
  - Pour de plus amples informations sur Redis, veuillez consulter [Autoriser l'accès](#) dans le Guide de l'utilisateur ElastiCache pour Redis.
  - Pour de plus amples informations sur Memcached, veuillez consulter [Autoriser l'accès](#) dans le Guide de l'utilisateur ElastiCache pour Memcached.

Vous pouvez utiliser des fichiers de configuration pour personnaliser votre environnement Elastic Beanstalk afin d'utiliser ElastiCache. Pour obtenir des exemples de fichiers de configuration intégrant ElastiCache à Elastic Beanstalk, veuillez consulter [Exemple : ElastiCache \(p. 766\)](#).

## Utilisation d'Elastic Beanstalk avec Amazon Elastic File System

Avec Amazon Elastic File System (Amazon EFS), vous pouvez créer des systèmes de fichiers réseau pouvant être montés par des instances dans plusieurs zones de disponibilité. Un système de fichiers Amazon EFS est une ressource d'AWS qui utilise des groupes de sécurité pour contrôler l'accès via le réseau dans votre VPC par défaut ou personnalisé.

Dans un environnement Elastic Beanstalk, vous pouvez utiliser Amazon EFS pour créer un répertoire partagé qui stocke les fichiers chargés ou modifiés par les utilisateurs de votre application. Votre application peut traiter un volume monté Amazon EFS comme un stockage local ; vous n'avez donc pas à modifier le code de votre application lorsque vous augmentez le nombre d'instances.

Pour de plus amples informations sur Amazon EFS, veuillez consulter le [Guide de l'utilisateur Amazon Elastic File System](#).

#### Sections

- [Fichiers de configuration \(p. 919\)](#)
- [Systèmes de fichiers chiffrés \(p. 919\)](#)
- [Exemples d'applications \(p. 919\)](#)
- [Nettoyage de systèmes de fichiers \(p. 920\)](#)

## Fichiers de configuration

Elastic Beanstalk fournit des [fichiers de configuration \(p. 737\)](#) que vous pouvez utiliser pour créer et monter des systèmes de fichiers Amazon EFS. Vous pouvez créer un volume Amazon EFS dans le cadre de votre environnement, ou monter un volume Amazon EFS que vous avez créé indépendamment d'Elastic Beanstalk.

- [storage-efs-createfilesystem.config](#) : utilise la clé Resources pour créer un nouveau système de fichiers et des points de montage dans Amazon EFS. Toutes les instances de votre environnement peuvent se connecter au même système de fichiers pour bénéficier d'un stockage partagé et évolutif. Utilisez [storage-efs-mountfilesystem.config](#) pour monter le système de fichiers sur chaque instance.

#### Ressources internes

Toutes les ressources que vous créez avec des fichiers de configuration sont liées au cycle de vie de votre environnement et seront perdues si vous mettez votre environnement hors service ou supprimez le fichier de configuration.

- [storage-efs-mountfilesystem.config](#) : monte un système de fichiers Amazon EFS dans un chemin d'accès local sur les instances de votre environnement. Vous pouvez créer le volume dans le cadre de l'environnement avec [storage-efs-createfilesystem.config](#), ou à l'extérieur de votre environnement à l'aide de la console Amazon EFS, de l'AWS CLI ou du kit SDK AWS.

Pour utiliser les fichiers de configuration, commencez par créer votre système de fichiers Amazon EFS avec [storage-efs-createfilesystem.config](#). Suivez les instructions du fichier de configuration et ajoutez celui-ci au répertoire [.ebextensions \(p. 737\)](#) dans votre code source pour créer le système de fichiers dans votre VPC.

Déployez votre code source mis à jour dans votre environnement Elastic Beanstalk pour vérifier que le système de fichiers est créé avec succès. Ensuite, ajoutez [storage-efs-mountfilesystem.config](#) pour monter le système de fichiers sur les instances de votre environnement. En faisant ceci dans deux déploiements distincts, vous garantissez que le système de fichiers reste intact si l'opération de montage échoue. Si vous effectuez ces deux étapes dans le même déploiement, un problème avec l'une des étapes entraînerait une mise hors service du système de fichiers si le déploiement échoue.

## Systèmes de fichiers chiffrés

Amazon EFS prend en charge les systèmes de fichiers chiffrés. Le fichier de configuration [storage-efs-createfilesystem.config](#) présenté dans cette rubrique définit deux options personnalisées que vous pouvez utiliser pour créer un système de fichiers chiffré Amazon EFS. Pour plus d'informations, suivez les instructions du fichier de configuration.

## Exemples d'applications

Elastic Beanstalk fournit également des exemples d'applications qui utilisent Amazon EFS pour le stockage partagé. Les deux projets sont des fichiers de configuration que vous pouvez utiliser avec un programme

d'installation WordPress ou Drupal standard pour exécuter un blog ou un autre système de gestion de contenu dans un environnement à charge équilibrée. Lorsqu'un utilisateur charge une photo ou un autre support, celui-ci est stocké dans un système de fichiers Amazon EFS, ce qui évite de devoir utiliser un plug-in pour stocker les fichiers chargés dans Amazon S3.

- [Load Balanced WordPress \(WordPress à charge équilibrée\)](#) : fichiers de configuration pour l'installation de WordPress en toute sécurité et son exécution dans un environnement Elastic Beanstalk à charge équilibrée.
- [Load Balanced Drupal \(Drupal à charge équilibrée\)](#) : fichiers de configuration et instructions pour l'installation de Drupal en toute sécurité et son exécution dans un environnement Elastic Beanstalk à charge équilibrée.

## Nettoyage de systèmes de fichiers

Si vous avez créé un système de fichiers Amazon EFS à l'aide d'un fichier de configuration dans le cadre de votre environnement Elastic Beanstalk, Elastic Beanstalk supprime le système de fichiers lorsque vous arrêtez l'environnement. Pour réduire les coûts de stockage d'une application en cours d'exécution, supprimez régulièrement les fichiers dont votre application n'a pas besoin ou assurez-vous que le code d'application gère correctement le cycle de vie de fichier.

En outre, si vous avez créé un système de fichiers Amazon EFS à l'extérieur d'un environnement Elastic Beanstalk et que vous l'avez monté sur les instances de l'environnement, sachez qu'Elastic Beanstalk ne le supprime pas lorsque vous arrêtez l'environnement. Pour être sûr que vos informations personnelles ne sont pas inutilement conservées, supprimez les fichiers stockés par votre application si vous n'en avez plus besoin ou supprimez le système de fichiers.

## Utilisation d'Elastic Beanstalk avec AWS Identity and Access Management

AWS Identity and Access Management (IAM) vous permet de contrôler en toute sécurité l'accès à vos ressources AWS. Cette section inclut des documents de référence concernant l'utilisation des stratégies IAM, des profils d'instance et des rôles de service.

Pour obtenir une présentation des autorisations, consultez [Rôles de service, profils d'instance et stratégies utilisateur \(p. 21\)](#). Pour la plupart des environnements, le rôle de service et le profil d'instance que la console Elastic Beanstalk vous invite à créer lorsque vous lancez votre premier environnement disposent de toutes les autorisations nécessaires. De même, les [stratégies gérées \(p. 946\)](#) fournies par Elastic Beanstalk pour l'accès complet et l'accès en lecture seule incluent toutes les autorisations utilisateur requises pour un usage quotidien.

Le [Guide de l'utilisateur IAM](#) inclut une description détaillée des autorisations AWS.

### Rubriques

- [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#)
- [Gestion des rôles de service Elastic Beanstalk \(p. 926\)](#)
- [Utilisation des rôles liés à un service pour Elastic Beanstalk \(p. 935\)](#)
- [Gestion des stratégies utilisateur Elastic Beanstalk \(p. 946\)](#)
- [Format Amazon Resource Name \(ARN\) pour Elastic Beanstalk \(p. 951\)](#)
- [Ressources et conditions pour les actions Elastic Beanstalk \(p. 952\)](#)
- [Utilisation de balises pour contrôler l'accès aux ressources Elastic Beanstalk \(p. 978\)](#)
- [Exemples de stratégies basées sur des stratégies gérées \(p. 981\)](#)
- [Exemples de stratégies basées sur des autorisations de ressource \(p. 984\)](#)

## Gestion des profils d'instance Elastic Beanstalk

Un profil d'instance est un conteneur pour un rôle AWS Identity and Access Management (IAM) que vous pouvez utiliser pour transmettre les informations liées au rôle à une instance Amazon EC2 lorsque l'instance démarre. Lorsque vous lancez un environnement à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande (CLI) EB, Elastic Beanstalk crée un profil d'instance par défaut, appelé `aws-elasticbeanstalk-ec2-role`, et lui alloue des stratégies gérées comportant des autorisations par défaut.

Elastic Beanstalk fournit trois stratégies gérées : une pour le niveau serveur web, une pour le niveau de travail et une avec des autorisations supplémentaires requises pour les environnements Docker conteneur multiple. La console affecte toutes ces stratégies au rôle attaché au profil d'instance par défaut. Les stratégies suivent.

## Stratégies de profil d'instance gérées

- AWS>ElasticBeanstalkWebTier – Accorde des autorisations pour permettre à l'application de télécharger les journaux dans Amazon S3 et les informations de débogage dans AWS X-Ray.

```
{ "Version": "2012-10-17", "Statement": [ { "Sid": "BucketAccess", "Action": [ "s3:Get*", "s3>List*", "s3:PutObject" ], "Effect": "Allow", "Resource": [ "arn:aws:s3:::elasticbeanstalk-*", "arn:aws:s3:::elasticbeanstalk-*/*" ] }, { "Sid": "XRayAccess", "Action": [ "xray:PutTraceSegments", "xray:PutTelemetryRecords", "xray:GetSamplingRules", "xray:GetSamplingTargets", "xray:GetSamplingStatisticSummaries" ], "Effect": "Allow", "Resource": "*" }, { "Sid": "CloudWatchLogsAccess", "Action": [ "logs:PutLogEvents", "logs>CreateLogStream", "logs:DescribeLogStreams", "logs:DescribeLogGroups" ], "Effect": "Allow", "Resource": [ "arn:aws:logs:*:*:log-group:/aws/elasticbeanstalk*" ] }, { "Sid": "ElasticBeanstalkHealthAccess",
```

```
        "Action": [
            "elasticbeanstalk:PutInstanceStatistics"
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:elasticbeanstalk:::application/*",
            "arn:aws:elasticbeanstalk:::environment/*"
        ]
    }
}
```

- AWSBeanstalkWorkerTier – Accorde des autorisations pour les chargements de journaux, le débogage, la publication de métriques et les tâches d'instance de travail, y compris la gestion des files d'attente, le choix de l'instance principale et les tâches périodiques.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "MetricsAccess",
            "Action": [
                "cloudwatch:PutMetricData"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Sid": "XRayAccess",
            "Action": [
                "xray:PutTraceSegments",
                "xray:PutTelemetryRecords",
                "xray:GetSamplingRules",
                "xray:GetSamplingTargets",
                "xray:GetSamplingStatisticSummaries"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Sid": "QueueAccess",
            "Action": [
                "sns:ChangeMessageVisibility",
                "sns:DeleteMessage",
                "sns:ReceiveMessage",
                "sns:SendMessage"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Sid": "BucketAccess",
            "Action": [
                "s3:Get*",
                "s3>List*",
                "s3:PutObject"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:s3:::elasticbeanstalk-*",
                "arn:aws:s3:::elasticbeanstalk-*/*"
            ]
        },
        {
            "Sid": "DynamoPeriodicTasks",
```

```

    "Action": [
        "dynamodb:BatchGetItem",
        "dynamodb:BatchWriteItem",
        "dynamodb>DeleteItem",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Query",
        "dynamodb:Scan",
        "dynamodb:UpdateItem"
    ],
    "Effect": "Allow",
    "Resource": [
        "arn:aws:dynamodb:*::table/*-stack-AWSEBWorkerCronLeaderRegistry*"
    ]
},
{
    "Sid": "CloudWatchLogsAccess",
    "Action": [
        "logs:PutLogEvents",
        "logs>CreateLogStream"
    ],
    "Effect": "Allow",
    "Resource": [
        "arn:aws:logs:*::log-group:/aws/elasticbeanstalk*"
    ]
},
{
    "Sid": "ElasticBeanstalkHealthAccess",
    "Action": [
        "elasticbeanstalk:PutInstanceStatistics"
    ],
    "Effect": "Allow",
    "Resource": [
        "arn:aws:elasticbeanstalk:*::application/*",
        "arn:aws:elasticbeanstalk:*::environment/*"
    ]
}
]
}

```

- AWSElasticBeanstalkMulticontainerDocker – Accorde des autorisations à Amazon Elastic Container Service pour la coordination des tâches de cluster.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ECSAccess",
            "Effect": "Allow",
            "Action": [
                "ecs:Poll",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:DiscoverPollEndpoint",
                "ecs:StartTelemetrySession",
                "ecs:RegisterContainerInstance",
                "ecs:DeregisterContainerInstance",
                "ecs:DescribeContainerInstances",
                "ecs:Submit"
            ],
            "Resource": "*"
        }
    ]
}

```

Pour autoriser les instances EC2 dans votre environnement à assumer le rôle `aws-elasticbeanstalk-ec2-role`, le profil d'instance spécifie Amazon EC2 comme une entité de confiance dans la stratégie de relation d'approbation de la manière suivante :

```
{  
    "Version": "2008-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Principal": {  
                "Service": "ec2.amazonaws.com"  
            },  
            "Action": "sts:AssumeRole"  
        }  
    ]  
}
```

Pour personnaliser les autorisations, vous pouvez soit ajouter des stratégies au rôle attaché au profil d'instance par défaut, soit créer votre propre profil d'instance avec un ensemble limité d'autorisations.

#### Sections

- [Vérification des autorisations attribuées au profil d'instance par défaut \(p. 924\)](#)
- [Mise à jour d'un profil d'instance par défaut obsolète \(p. 924\)](#)
- [Ajout d'autorisations au profil d'instance par défaut \(p. 925\)](#)
- [Création d'un profil d'instance \(p. 925\)](#)
- [Profils d'instance avec les plateformes Amazon Linux 2 \(p. 925\)](#)

## Vérification des autorisations attribuées au profil d'instance par défaut

Les autorisations allouées à votre profil d'instance par défaut peuvent varier en fonction de la date à laquelle il a été créé, de la date du dernier lancement d'un environnement et du client que vous avez utilisé. Vous pouvez vérifier les autorisations sur le profil d'instance par défaut dans la console IAM.

Pour vérifier les autorisations du profil d'instance par défaut

1. Ouvrez la page [Roles \(Rôles\)](#) dans la console IAM.
2. Choisissez `aws-elasticbeanstalk-ec2-role`.
3. Sur l'onglet Autorisations, vérifiez la liste de stratégies attachée au rôle.
4. Pour voir les autorisations octroyées par une stratégie, choisissez la stratégie.

## Mise à jour d'un profil d'instance par défaut obsolète

Si le profil d'instance par défaut ne dispose pas des autorisations requises, vous pouvez le mettre à jour en [créant un environnement \(p. 439\)](#) dans la console de gestion d'environnement Elastic Beanstalk.

Vous pouvez sinon ajouter manuellement les stratégies gérées au rôle attaché au profil d'instance par défaut.

Pour ajouter des stratégies gérées au rôle attaché au profil d'instance par défaut

1. Ouvrez la page [Roles \(Rôles\)](#) dans la console IAM.
2. Choisissez `aws-elasticbeanstalk-ec2-role`.

3. Sous l'onglet Autorisations, choisissez Attach policies (Attacher des stratégies).
4. Saisissez **AWSElasticBeanstalk** pour filtrer les stratégies.
5. Sélectionnez les stratégies suivantes, puis choisissez Attacher une stratégie :
  - AWSElasticBeanstalkWebTier
  - AWSElasticBeanstalkWorkerTier
  - AWSElasticBeanstalkMulticontainerDocker

## Ajout d'autorisations au profil d'instance par défaut

Si votre application accède à des ressources ou des API AWS auxquelles des autorisations ne sont pas octroyées dans le profil d'instance par défaut, ajoutez des stratégies qui accordent des autorisations dans la console IAM.

Pour ajouter des stratégies au rôle attaché au profil d'instance par défaut

1. Ouvrez la page [Roles \(Rôles\)](#) dans la console IAM.
2. Choisissez aws-elasticbeanstalk-ec2-role.
3. Sous l'onglet Autorisations, choisissez Attach policies (Attacher des stratégies).
4. Sélectionnez la stratégie gérée relative aux services supplémentaires utilisés par votre application. Par exemple, `AmazonS3FullAccess` ou `AmazonDynamoDBFullAccess`.
5. Choisissez Attacher la stratégie.

## Création d'un profil d'instance

Un profil d'instance est une enveloppe autour d'un rôle IAM standard qui permet à une instance EC2 d'assumer le rôle. Vous pouvez créer des profils d'instance supplémentaires pour personnaliser des autorisations pour différentes applications ou pour créer un profil d'instance qui n'accorde pas d'autorisations pour le niveau de travail ou les environnements Docker conteneur multiple, si vous n'utilisez pas ces fonctionnalités.

Pour créer un profil d'instance

1. Ouvrez la page [Roles \(Rôles\)](#) dans la console IAM.
2. Sélectionnez Créer un rôle.
3. Sous AWS service (Service AWS), sélectionnez EC2.
4. Sélectionnez Étape suivante : autorisations.
5. Associez les stratégies gérées adéquates fournies par Elastic Beanstalk et toutes les stratégies supplémentaires fournissant les autorisations dont votre application a besoin.
6. Choisissez Next: Tags (Suivant : Balises).
7. (Facultatif) Ajoutez des balises au rôle.
8. Choisissez Next: Review (Suivant : Vérification).
9. Entrez un nom pour le rôle.
10. Sélectionnez Create role (Créer un rôle).

## Profils d'instance avec les plateformes Amazon Linux 2

Pour fonctionner correctement, les plateformes Amazon Linux 2 nécessitent un profil d'instance. Par exemple, toutes les versions de plateforme Amazon Linux 2 permettent une intégrité améliorée par défaut

lors de la création de l'environnement. Les instances ont besoin des autorisations appropriées pour collecter et signaler des informations d'intégrité améliorée.

## Gestion des rôles de service Elastic Beanstalk

Pour gérer et surveiller votre environnement, AWS Elastic Beanstalk effectue des actions sur les ressources de l'environnement en votre nom. Elastic Beanstalk a besoin de certaines autorisations pour effectuer ces actions, et il assume les fonctions de service AWS Identity and Access Management (IAM) pour obtenir ces autorisations.

Elastic Beanstalk a besoin d'utiliser des informations d'identification de sécurité temporaires chaque fois qu'il assume un rôle de service. Pour obtenir ces informations d'identification, Elastic Beanstalk envoie une requête à AWS Security Token Service (AWS STS) sur un point de terminaison spécifique à une région. Pour de plus amples informations, veuillez consulter [Informations d'identification de sécurité temporaires](#) dans le Guide de l'utilisateur IAM.

### Note

Si le point de terminaison AWS STS de la région où se trouve votre environnement est désactivé, Elastic Beanstalk envoie la demande à un autre point de terminaison qui ne peut pas être désactivé. Ce point de terminaison est associé à une région différente. Il s'agit par conséquent d'une demande inter-région. Pour de plus amples informations, veuillez consulter [Activation et désactivation de AWS STS dans une région AWS](#) dans le Guide de l'utilisateur IAM.

## Gestion des rôles de service à l'aide de la console Elastic Beanstalk et de l'interface de ligne de commande (CLI) EB

Vous pouvez utiliser la console Elastic Beanstalk et l'interface de ligne de commande (CLI) EB pour définir des rôles de service pour votre environnement avec un ensemble suffisant d'autorisations. Ils créent un rôle de service par défaut et utilisent des stratégies gérées.

### Stratégies gérées des rôles de service

Elastic Beanstalk fournit une stratégie gérée pour la [surveillance améliorée de l'état \(p. 837\)](#), et une autre qui comporte les autorisations supplémentaires requises pour les [mises à jour gérées de la plateforme \(p. 501\)](#). La console et l'interface de ligne de commande (CLI) EB attribuent ces deux stratégies au rôle de service par défaut qu'elles créent pour vous. Ces stratégies doivent être utilisées uniquement pour ce rôle de service par défaut. Elles ne doivent pas être utilisées avec d'autres utilisateurs ou rôles dans vos comptes.

#### `AWSElasticBeanstalkEnhancedHealth`

Cette stratégie accorde des autorisations pour qu'Elastic Beanstalk surveille l'état de l'instance et de l'environnement.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "elasticloadbalancing:DescribeInstanceHealth",  
                "elasticloadbalancing:DescribeLoadBalancers",  
                "elasticloadbalancing:DescribeTargetHealth",  
                "ec2:DescribeInstances",  
                "ec2:DescribeInstanceStatus",  
                "ec2:GetConsoleOutput",  
                "ec2:StartInstances",  
                "ec2:StopInstances"  
            ]  
        }  
    ]  
}
```

```
        "ec2:AssociateAddress",
        "ec2:DescribeAddresses",
        "ec2:DescribeSecurityGroups",
        "sns:GetQueueAttributes",
        "sns:GetQueueUrl",
        "autoscaling:DescribeAutoScalingGroups",
        "autoscaling:DescribeAutoScalingInstances",
        "autoscaling:DescribeScalingActivities",
        "autoscaling:DescribeNotificationConfigurations",
        "sns:Publish"
    ],
    "Resource": [
        "*"
    ]
}
]
```

#### [AWS Elastic Beanstalk Managed Updates Customer Role Policy](#)

Cette stratégie octroie des autorisations à Elastic Beanstalk pour mettre à jour les environnements en votre nom afin d'effectuer des mises à jour de plateforme gérées.

##### Groupements d'autorisations de niveau service

Cette stratégie est groupée en instructions basées sur le jeu d'autorisations fourni.

- *ElasticBeanstalkPermissions* – Ce groupe d'autorisations sert à appeler les actions de service Elastic Beanstalk (API Elastic Beanstalk).
- *AllowPassRoleToElasticBeanstalkAndDownstreamServices* – Ce groupe d'autorisations permet de transmettre n'importe quel rôle à Elastic Beanstalk et à d'autres services aval comme AWS CloudFormation.
- *ReadOnlyPermissions* – Ce groupe d'autorisations sert à collecter des informations sur l'environnement en cours d'exécution.
- *\*OperationPermissions* – Les groupes avec ce schéma de nommage servent à appeler les opérations nécessaires pour effectuer les mises à jour de la plateforme.
- *\*BroadOperationPermissions* – Les groupes avec ce schéma de nommage servent à appeler les opérations nécessaires pour effectuer les mises à jour de la plateforme. Ils incluent également des autorisations étendues pour la prise en charge d'environnements hérités.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ElasticBeanstalkPermissions",
            "Effect": "Allow",
            "Action": [
                "elasticbeanstalk:*"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowPassRoleToElasticBeanstalkAndDownstreamServices",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "elasticbeanstalk.amazonaws.com",

```

```
        "ec2.amazonaws.com",
        "ec2.amazonaws.com.cn",
        "autoscaling.amazonaws.com",
        "elasticloadbalancing.amazonaws.com",
        "ecs.amazonaws.com",
        "cloudformation.amazonaws.com"
    ]
}
}
},
{
"Sid": "ReadOnlyPermissions",
"Effect": "Allow",
>Action": [
    "autoscaling:DescribeAccountLimits",
    "autoscaling:DescribeAutoScalingGroups",
    "autoscaling:DescribeAutoScalingInstances",
    "autoscaling:DescribeLaunchConfigurations",
    "autoscaling:DescribeLoadBalancers",
    "autoscaling:DescribeNotificationConfigurations",
    "autoscaling:DescribeScalingActivities",
    "autoscaling:DescribeScheduledActions",
    "ec2:DescribeAccountAttributes",
    "ec2:DescribeAddresses",
    "ec2:DescribeAvailabilityZones",
    "ec2:DescribeImages",
    "ec2:DescribeInstanceAttribute",
    "ec2:DescribeInstances",
    "ec2:DescribeKeyPairs",
    "ec2:DescribeLaunchTemplates",
    "ec2:DescribeLaunchTemplateVersions",
    "ec2:DescribeSecurityGroups",
    "ec2:DescribeSnapshots",
    "ec2:DescribeSpotInstanceRequests",
    "ec2:DescribeSubnets",
    "ec2:DescribeVpcClassicLink",
    "ec2:DescribeVpcs",
    "elasticloadbalancing:DescribeInstanceHealth",
    "elasticloadbalancing:DescribeLoadBalancers",
    "elasticloadbalancing:DescribeTargetGroups",
    "elasticloadbalancing:DescribeTargetHealth",
    "logs:DescribeLogGroups",
    "rds:DescribeDBEngineVersions",
    "rds:DescribeDBInstances",
    "rds:DescribeOrderableDBInstanceStateOptions",
    "sns>ListSubscriptionsByTopic"
],
"Resource": [
    "*"
]
},
{
"Sid": "EC2BroadOperationPermissions",
"Effect": "Allow",
>Action": [
    "ec2:AllocateAddress",
    "ec2:AssociateAddress",
    "ec2:AuthorizeSecurityGroupEgress",
    "ec2:AuthorizeSecurityGroupIngress",
    "ec2>CreateLaunchTemplate",
    "ec2>CreateLaunchTemplateVersion",
    "ec2>CreateSecurityGroup",
    "ec2>DeleteLaunchTemplate",
    "ec2>DeleteLaunchTemplateVersions",
    "ec2>DeleteSecurityGroup",
    "ec2:DisassociateAddress",
    "ec2:ModifyNetworkInterfaceAttribute"
]
]
```

```
        "ec2:ReleaseAddress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress"
    ],
    "Resource": "*"
},
{
    "Sid": "EC2RunInstancesOperationPermissions",
    "Effect": "Allow",
    "Action": "ec2:RunInstances",
    "Resource": "*",
    "Condition": {
        "ArnLike": {
            "ec2:LaunchTemplate": "arn:aws:ec2:::launch-template/*"
        }
    }
},
{
    "Sid": "EC2TerminateInstancesOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "ec2:TerminateInstances"
    ],
    "Resource": "arn:aws:ec2:::instance/*",
    "Condition": {
        "StringLike": {
            "ec2:ResourceTag/aws:cloudformation:stack-id": [
                "arn:aws:cloudformation:::stack/awseb-e-*",
                "arn:aws:cloudformation:::stack/eb-*"
            ]
        }
    }
},
{
    "Sid": "ECSBroadOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "ecs>CreateCluster",
        "ecs>DescribeClusters",
        "ecs:RegisterTaskDefinition"
    ],
    "Resource": "*"
},
{
    "Sid": "ECSDeleteClusterOperationPermissions",
    "Effect": "Allow",
    "Action": "ecs>DeleteCluster",
    "Resource": "arn:aws:ecs:::cluster/awseb-*"
},
{
    "Sid": "ASGOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "autoscaling:AttachInstances",
        "autoscaling>CreateAutoScalingGroup",
        "autoscaling>CreateLaunchConfiguration",
        "autoscaling>DeleteLaunchConfiguration",
        "autoscaling>DeleteAutoScalingGroup",
        "autoscaling>DeleteScheduledAction",
        "autoscaling:DetachInstances",
        "autoscaling>DeletePolicy",
        "autoscaling:PutScalingPolicy",
        "autoscaling:PutScheduledUpdateGroupAction",
        "autoscaling:PutNotificationConfiguration",
        "autoscaling:ResumeProcesses",
        "autoscaling:SetDesiredCapacity",
```

```

        "autoscaling:SuspendProcesses",
        "autoscaling:TerminateInstanceInAutoScalingGroup",
        "autoscaling:UpdateAutoScalingGroup"
    ],
    "Resource": [
        "arn:aws:autoscaling:*:launchConfiguration:launchConfigurationName/
awseb-e-*",
        "arn:aws:autoscaling:*:launchConfiguration:launchConfigurationName/eb-
*",
        "arn:aws:autoscaling:*:autoScalingGroup:autoScalingGroupName/awseb-e-
*",
        "arn:aws:autoscaling:*:autoScalingGroup:autoScalingGroupName/eb-*"
    ]
},
{
    "Sid": "CFNOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "cloudformation:/*"
    ],
    "Resource": [
        "arn:aws:cloudformation:stack/awseb-*",
        "arn:aws:cloudformation:stack/eb-*"
    ]
},
{
    "Sid": "ELBOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer",
        "elasticloadbalancing:ConfigureHealthCheck",
        "elasticloadbalancing>CreateLoadBalancer",
        "elasticloadbalancing>DeleteLoadBalancer",
        "elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
        "elasticloadbalancing:DeregisterTargets",
        "elasticloadbalancing:RegisterInstancesWithLoadBalancer",
        "elasticloadbalancing:RegisterTargets"
    ],
    "Resource": [
        "arn:aws:elasticloadbalancing:targetgroup/awseb-*",
        "arn:aws:elasticloadbalancing:targetgroup/eb-*",
        "arn:aws:elasticloadbalancing:loadbalancer/awseb-*",
        "arn:aws:elasticloadbalancing:loadbalancer/eb-*",
        "arn:aws:elasticloadbalancing:loadbalancer/*/awseb-/*/*",
        "arn:aws:elasticloadbalancing:loadbalancer/*/eb-/*/*"
    ]
},
{
    "Sid": "CWLogsOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "logs>CreateLogGroup",
        "logs>DeleteLogGroup",
        "logs:PutRetentionPolicy"
    ],
    "Resource": "arn:aws:logs:log-group:/aws/elasticbeanstalk/*"
},
{
    "Sid": "S3ObjectOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "s3>DeleteObject",
        "s3>GetObject",
        "s3>GetObjectAcl",
        "s3>GetObjectVersion",
        "s3>GetObjectVersionAcl",

```

```

        "s3:PutObject",
        "s3:PutObjectAcl",
        "s3:PutObjectVersionAcl"
    ],
    "Resource": "arn:aws:s3:::elasticbeanstalk-*/*"
},
{
    "Sid": "S3BucketOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "s3:GetBucketLocation",
        "s3:GetBucketPolicy",
        "s3>ListBucket",
        "s3:PutBucketPolicy"
    ],
    "Resource": "arn:aws:s3:::elasticbeanstalk-*"
},
{
    "Sid": "SNSSOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "sns>CreateTopic",
        "sns:GetTopicAttributes",
        "sns:SetTopicAttributes",
        "sns:Subscribe"
    ],
    "Resource": "arn:aws:sns:*::ElasticBeanstalkNotifications-*"
},
{
    "Sid": "SQSOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "sns:GetQueueAttributes",
        "sns:GetQueueUrl"
    ],
    "Resource": [
        "arn:aws:sqs:*::awseb-e-*",
        "arn:aws:sqs:*::eb-*"
    ]
},
{
    "Sid": "CWPutMetricAlarmOperationPermissions",
    "Effect": "Allow",
    "Action": [
        "cloudwatch:PutMetricAlarm"
    ],
    "Resource": [
        "arn:aws:cloudwatch::*:alarm:awseb-*",
        "arn:aws:cloudwatch::*:alarm:eb-*"
    ]
}
]
}

```

Pour afficher le contenu d'une stratégie gérée, vous pouvez aussi utiliser la [page Policies \(Stratégies\)](#) de la console IAM.

#### Note

Par le passé, Elastic Beanstalk prenait en charge la stratégie de rôle de service gérée AWSElasticBeanstalkService. Cette stratégie a été remplacée par AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy. Vous pouvez toujours voir et utiliser la stratégie précédente dans la console IAM. Nous vous recommandons toutefois d'utiliser la nouvelle stratégie gérée (AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy). Ajoutez

des stratégies personnalisées pour accorder des autorisations aux ressources personnalisées, le cas échéant.

## Utilisation de la console Elastic Beanstalk

Lorsque vous lancez un environnement dans la console Elastic Beanstalk, la console crée un rôle de service par défaut, appelé `aws-elasticbeanstalk-service-role`, auquel elle associe des stratégies gérées comportant des autorisations par défaut.

Pour permettre à Elastic Beanstalk d'assumer le rôle `aws-elasticbeanstalk-service-role`, le rôle de service spécifique Elastic Beanstalk comme entité de confiance dans la stratégie de relation d'approbation.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "",  
            "Effect": "Allow",  
            "Principal": {  
                "Service": "elasticbeanstalk.amazonaws.com"  
            },  
            "Action": "sts:AssumeRole",  
            "Condition": {  
                "StringEquals": {  
                    "sts:ExternalId": "elasticbeanstalk"  
                }  
            }  
        }  
    ]  
}
```

Lorsque vous activez les [mises à jour de plateforme gérée \(p. 501\)](#) pour votre environnement, Elastic Beanstalk assume un rôle de service de mises à jour gérées distinct pour effectuer les mises à jour gérées. Par défaut, la console Elastic Beanstalk utilise le même rôle de service généré, `aws-elasticbeanstalk-service-role`, pour le rôle de service de mises à jour gérées. Si vous modifiez votre rôle de service par défaut, la console définit le rôle de service de mises à jour gérées pour qu'il utilise le rôle lié au service de mises à jour gérées, `AWSServiceRoleForElasticBeanstalkManagedUpdates`. Pour plus d'informations sur les rôles liés à un service, consultez [the section called “Utilisation des rôles liés à un service” \(p. 933\)](#).

### Note

En raison de problèmes d'autorisation, le service Elastic Beanstalk ne réussit pas toujours à créer ce rôle lié au service pour vous. Par conséquent, la console tente de le créer explicitement. Pour vous assurer que votre compte dispose de ce rôle lié au service, créez un environnement au moins une fois à l'aide de la console et configurez les mises à jour gérées pour qu'elles soient activées avant de créer l'environnement.

## Utilisation de l'interface de ligne de commande (CLI) EB

Si vous lancez un environnement à l'aide de la commande [the section called “eb create” \(p. 1078\)](#) de l'interface de ligne de commande Elastic Beanstalk (EB CLI) et que vous ne spécifiez pas de rôle de service au moyen de l'option `--service-role`, Elastic Beanstalk crée le rôle de service par défaut `aws-elasticbeanstalk-service-role`. Si le rôle de service par défaut existe déjà, Elastic Beanstalk l'utilise pour le nouvel environnement. La console Elastic Beanstalk effectue également des actions similaires dans ces circonstances.

À la différence de la console, vous ne pouvez pas spécifier de rôle de service de mises à jour gérées lorsque vous utilisez une option de commande EB CLI. Si vous activez les mises à jour gérées pour votre

environnement, vous devez définir le rôle de service de mises à jour gérées au moyen des options de configuration. L'exemple suivant active les mises à jour gérées et utilise le rôle de service par défaut en tant que rôle de service de mises à jour gérées.

#### Example .ebextensions/managed-platform-update.config

```
option_settings:  
  aws:elasticbeanstalk:managedactions:  
    ManagedActionsEnabled: true  
    PreferredStartTime: "Tue:09:00"  
    ServiceRoleForManagedUpdates: "aws-elasticbeanstalk-service-role"  
  aws:elasticbeanstalk:managedactions:platformupdate:  
    UpdateLevel: patch  
    InstanceRefreshEnabled: true
```

## Gestion des rôles de service à l'aide de l'API Elastic Beanstalk

Lorsque vous utilisez l'action `CreateEnvironment` de l'API Elastic Beanstalk pour créer un environnement, spécifiez un rôle de service avec l'option de configuration `ServiceRole` dans l'espace de noms `aws:elasticbeanstalk:environment` ([p. 700](#)). Pour plus d'informations sur l'utilisation de la surveillance améliorée de l'état avec l'API Elastic Beanstalk, veuillez consulter [Utilisation des rapports améliorés sur l'état à l'aide de l'API Elastic Beanstalk \(p. 869\)](#).

En outre, si vous activez les [mises à jour de plateforme gérée \(p. 501\)](#) pour votre environnement, vous pouvez spécifier un rôle de service de mises à jour gérées à l'aide de l'option `ServiceRoleForManagedUpdates` de l'espace de noms `aws:elasticbeanstalk:managedactions` ([p. 707](#)).

## Utilisation des rôles liés à un service

Un rôle lié à un service est un type de fonction de service unique prédéfini par Elastic Beanstalk pour inclure toutes les autorisations exigées par le service pour appeler d'autres services AWS en votre nom. Le rôle lié à un service est associé à votre compte. Elastic Beanstalk le crée une fois, puis le réutilise lors de la création d'environnements supplémentaires. Pour plus d'informations sur l'utilisation des rôles liés à un service avec les environnements Elastic Beanstalk, veuillez consulter [Utilisation des rôles liés à un service pour Elastic Beanstalk \(p. 935\)](#).

Si vous créez un environnement à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de rôle de service, Elastic Beanstalk crée un [rôle lié au service de surveillance \(p. 935\)](#) pour votre compte, s'il n'en existe pas déjà un. Elastic Beanstalk utilise ce rôle pour le nouvel environnement. Vous pouvez également utiliser IAM pour créer à l'avance le rôle lié au service de surveillance de votre compte. Une fois que votre compte dispose de ce rôle, vous pouvez l'utiliser pour créer un environnement à l'aide de l'API Elastic Beanstalk, de la console Elastic Beanstalk ou de l'interface de ligne de commande (CLI) EB.

Si vous activez les [mises à jour de plateforme gérées \(p. 501\)](#) pour l'environnement et que vous spécifiez `AWSServiceRoleForElasticBeanstalkManagedUpdates` comme valeur pour l'option `ServiceRoleForManagedUpdates` de l'espace de noms `aws:elasticbeanstalk:managedactions` ([p. 707](#)), Elastic Beanstalk crée un [rôle lié au service managed-updates \(p. 941\)](#) pour votre compte, s'il n'en existe pas déjà un. Elastic Beanstalk utilise le rôle pour effectuer des mises à jour gérées pour le nouvel environnement.

### Note

Quand Elastic Beanstalk tente de créer les rôles de surveillance et de mises à jour gérées liés au service pour votre compte lorsque vous créez un environnement, vous devez disposer de l'autorisation `iam:CreateServiceLinkedRole`. Si vous ne disposez pas de cette autorisation, la création de l'environnement échoue et un message expliquant le problème s'affiche.

En lieu et place, un autre utilisateur ayant l'autorisation de créer des rôles liés à un service peut utiliser IAM pour créer le rôle de service lié à l'avance. En employant cette méthode, vous n'avez pas besoin de l'autorisation `iam:CreateServiceLinkedRole` pour créer votre environnement.

## Vérification des autorisations de rôle de service par défaut

Les autorisations accordées par votre rôle de service par défaut peuvent varier selon la date à laquelle il a été créé, la date du dernier lancement d'un environnement et le client que vous avez utilisé. Dans la console IAM, vous pouvez vérifier les autorisations accordées par le rôle de service par défaut.

Pour vérifier les autorisations du rôle de service par défaut

1. Dans la console IAM, ouvrez la [page Roles \(Rôles\)](#).
2. Choisissez `aws-elasticbeanstalk-service-role`.
3. Sur l'onglet Autorisations, vérifiez la liste de stratégies attachée au rôle.
4. Pour afficher les autorisations octroyées à une stratégie, sélectionnez la stratégie.

## Mise à jour d'un rôle de service par défaut obsolète

Si le rôle de service par défaut ne dispose pas des autorisations requises, vous pouvez le mettre à jour en [créant un environnement \(p. 439\)](#) dans la console de gestion d'environnement Elastic Beanstalk.

En variante, vous pouvez ajouter manuellement les stratégies gérées au rôle de service par défaut.

Pour ajouter des stratégies gérées au rôle de service par défaut

1. Dans la console IAM, ouvrez la [page Roles \(Rôles\)](#).
2. Choisissez `aws-elasticbeanstalk-service-role`.
3. Sous l'onglet Autorisations, choisissez **Attach policies** (Attacher des stratégies).
4. Entrez **AWSElasticBeanstalk** pour filtrer les stratégies.
5. Sélectionnez les stratégies suivantes, puis choisissez **Attacher une stratégie** :
  - `AWSElasticBeanstalkEnhancedHealth`
  - `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy`

## Ajout d'autorisations au rôle de service par défaut

Si votre application inclut des fichiers de configuration qui font référence à des ressources AWS pour lesquelles les autorisations ne sont pas incluses dans la fonction du service par défaut, Elastic Beanstalk peut avoir besoin d'autorisations supplémentaires. Ces autorisations supplémentaires sont nécessaires pour résoudre ces références lorsque le service traite les fichiers de configuration lors d'une mise à jour gérée. S'il manque des autorisations, la mise à jour échoue et Elastic Beanstalk renvoie un message indiquant les autorisations dont il a besoin. Suivez ces étapes pour ajouter des autorisations pour les services supplémentaires au rôle de service par défaut dans la console IAM.

Pour ajouter des stratégies supplémentaires au rôle de service par défaut

1. Dans la console IAM, ouvrez la [page Roles \(Rôles\)](#).
2. Choisissez `aws-elasticbeanstalk-service-role`.
3. Sous l'onglet Autorisations, choisissez **Attach policies** (Attacher des stratégies).
4. Sélectionnez la stratégie gérée relative aux services supplémentaires utilisés par votre application. Par exemple, `AmazonAPIGatewayAdministrator` ou `AmazonElasticFileSystemFullAccess`.
5. Choisissez **Attacher la stratégie**.

## Création d'un rôle de service

Si vous ne pouvez pas utiliser le rôle de service par défaut, créez un rôle de service.

Pour créer un rôle de service

1. Dans la console IAM, ouvrez la [page Roles \(Rôles\)](#).
2. Sélectionnez Create role (Créer un rôle).
3. Sous AWS service (Service AWS), choisissez AWS Elastic Beanstalk, puis sélectionnez votre cas d'utilisation.
4. Sélectionnez Étape suivante : autorisations.
5. Associez les stratégies gérées `AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy` et `AWSElasticBeanstalkEnhancedHealth` et toutes les stratégies supplémentaires fournissant les autorisations dont votre application a besoin.
6. Choisissez Next: Tags (Suivant : Balises).
7. (Facultatif) Ajoutez des balises au rôle.
8. Choisissez Next: Review (Suivant : Vérification).
9. Entrez un nom pour le rôle.
10. Sélectionnez Create role (Créer un rôle).

Appliquez votre rôle de service personnalisé lorsque vous créez un environnement à l'aide de [l'assistant de création d'environnement \(p. 442\)](#) ou de l'option `--service-role` de la commande `eb create (p. 1078)`.

## Utilisation des rôles liés à un service pour Elastic Beanstalk

AWS Elastic Beanstalk utilise des [rôles AWS Identity and Access Management \(IAM\) liés à un service](#). Un rôle lié à un service est un type unique de rôle IAM lié directement à Elastic Beanstalk. Les rôles liés à un service sont prédéfinis par Elastic Beanstalk et comprennent toutes les autorisations nécessaires au service pour appeler d'autres services AWS en votre nom.

Elastic Beanstalk définit quelques types de rôles liés au service :

- Rôle lié à un service de surveillance – Permet à Elastic Beanstalk de surveiller l'état des environnements en cours d'exécution et de publier des notifications d'événements d'état.
- Rôle lié à un service de maintenance – Permet à Elastic Beanstalk d'effectuer régulièrement des activités de maintenance pour vos environnements en cours d'exécution.
- Rôle lié à un service de mises à jour gérées – Permet à Elastic Beanstalk d'effectuer des mises à jour planifiées de vos environnements en cours d'exécution.

### Rubriques

- [Rôle lié à un service de surveillance \(p. 935\)](#)
- [Rôle lié à un service de maintenance \(p. 938\)](#)
- [Rôle lié à un service de mises à jour gérées \(p. 941\)](#)

## Rôle lié à un service de surveillance

AWS Elastic Beanstalk utilise des [rôles AWS Identity and Access Management \(IAM\) liés à un service](#). Un rôle lié à un service est un type unique de rôle IAM lié directement à Elastic Beanstalk. Les rôles liés

à un service sont prédefinis par Elastic Beanstalk et comprennent toutes les autorisations nécessaires au service pour appeler d'autres services AWS en votre nom.

Un rôle lié à un service simplifie la configuration d'AWS Lambda, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. AWS Lambda définit les autorisations de ses rôles liés à un service et, sauf indication contraire, seul AWS Lambda peut assumer ses rôles. Les autorisations définies comprennent la stratégie d'approbation et la stratégie d'autorisation. De plus, cette stratégie d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Votre ressources Elastic Beanstalk sont ainsi protégées car vous ne pouvez pas involontairement supprimer d'autorisation pour accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez [Services AWS qui fonctionnent avec IAM](#) et recherchez les services avec un Oui dans la colonne Rôle lié à un service. Choisissez un Oui ayant un lien permettant de consulter la documentation du rôle lié à un service, pour ce service.

## Autorisations des rôles liés à un service pour Elastic Beanstalk

Elastic Beanstalk utilise le rôle lié à un service nommé AWSServiceRoleForElasticBeanstalk – Permet à Elastic Beanstalk de surveiller l'état des environnements en cours d'exécution et de publier des notifications d'événements d'état.

Le rôle lié à un service AWSServiceRoleForElasticBeanstalk approuve les services suivants pour assumer le rôle :

- elasticbeanstalk.amazonaws.com

La stratégie d'autorisations du rôle lié à un service AWSServiceRoleForElasticBeanstalk contient toutes les autorisations dont Elastic Beanstalk a besoin pour exécuter des actions en votre nom :

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllowCloudformationReadOperationsOnElasticBeanstalkStacks",  
            "Effect": "Allow",  
            "Action": [  
                "cloudformation:DescribeStackResource",  
                "cloudformation:DescribeStackResources",  
                "cloudformation:DescribeStacks"  
            ],  
            "Resource": [  
                "arn:aws:cloudformation:*:*:stack/awseb-*",  
                "arn:aws:cloudformation:*:*:stack/eb-*"  
            ]  
        },  
        {  
            "Sid": "AllowOperations",  
            "Effect": "Allow",  
            "Action": [  
                "autoscaling:DescribeAutoScalingGroups",  
                "autoscaling:DescribeAutoScalingInstances",  
                "autoscaling:DescribeNotificationConfigurations",  
                "autoscaling:DescribeScalingActivities",  
                "autoscaling:PutNotificationConfiguration",  
                "ec2:DescribeInstanceStatus",  
                "ec2:AssociateAddress",  
                "ec2:DescribeAddresses",  
                "ec2:DescribeInstances".  
            ]  
        }  
    ]  
}
```

```
        "ec2:DescribeSecurityGroups",
        "elasticloadbalancing:DescribeInstanceHealth",
        "elasticloadbalancing:DescribeLoadBalancers",
        "elasticloadbalancing:DescribeTargetHealth",
        "elasticloadbalancing:DescribeTargetGroups",
        "sns:GetQueueAttributes",
        "sns:GetQueueUrl",
        "sns:Publish"
    ],
    "Resource": [
        "*"
    ]
}
]
```

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour de plus amples informations, veuillez consulter [Autorisations de rôles liés à un service](#) dans le Guide de l'utilisateur IAM.

Sinon, vous pouvez utiliser une stratégie gérée par AWS pour [fournir un accès complet \(p. 946\)](#) à Elastic Beanstalk.

## Création d'un rôle lié à un service pour Elastic Beanstalk

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de rôle de service, Elastic Beanstalk crée un rôle lié à un service en votre nom.

### Important

Si vous utilisez le service Elastic Beanstalk avant le 27 septembre 2017, date à laquelle le service a commencé à prendre en charge le rôle lié à un service AWSServiceRoleForElasticBeanstalk, et que votre compte en avait besoin, Elastic Beanstalk a créé le rôle AWSServiceRoleForElasticBeanstalk dans votre compte. Pour de plus amples informations, veuillez consulter [Un nouveau rôle est apparu dans mon compte IAM](#).

Quand Elastic Beanstalk tente de créer le rôle lié à un service AWSServiceRoleForElasticBeanstalk pour votre compte lorsque vous créez un environnement, vous devez disposer de l'autorisation `iam:CreateServiceLinkedRole`. Si vous n'avez pas cette autorisation, la création d'environnement échoue, et un message expliquant le problème s'affiche.

En lieu et place, un autre utilisateur ayant l'autorisation de créer des rôles liés à un service peut utiliser IAM pour précréer le rôle lié à un service à l'avance. Vous pouvez ensuite créer votre environnement, même sans avoir l'autorisation `iam:CreateServiceLinkedRole`.

Vous pouvez utiliser la console IAM pour créer un rôle lié à un service avec le cas d'utilisation Elastic Beanstalk. Dans l'interface de ligne de commande (CLI) IAM ou l'API IAM, créez un rôle lié à un service avec le nom de service `elasticbeanstalk.amazonaws.com`. Pour de plus amples informations, veuillez consulter [Création d'un rôle lié à un service](#) dans le Guide de l'utilisateur IAM. Si vous supprimez ce rôle lié à un service, vous pouvez utiliser ce même processus pour créer le rôle à nouveau.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de rôle de service, Elastic Beanstalk crée à nouveau un rôle lié à un service en votre nom.

## Modification d'un rôle lié à un service pour Elastic Beanstalk

Elastic Beanstalk ne vous permet pas de modifier le rôle lié à un service AWSServiceRoleForElasticBeanstalk. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez

pas modifier le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour de plus amples informations, veuillez consulter [Modification d'un rôle lié à un service](#) dans le Guide de l'utilisateur IAM.

## Suppression d'un rôle lié à un service pour Elastic Beanstalk

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

### Nettoyage d'un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez d'abord vous assurer que tous les environnements Elastic Beanstalk utilisent un rôle de service différent ou sont arrêtés.

#### Note

Si le service Elastic Beanstalk utilise le rôle lié à un service lorsque vous essayez d'arrêter les environnements, l'arrêt peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour arrêter un environnement Elastic Beanstalk qui utilise le rôle AWSServiceRoleForElasticBeanstalk (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Consultez [eb terminate \(p. 1120\)](#) pour plus de détails sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'interface de ligne de commande (CLI) EB.

Pour de plus amples informations sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'API, veuillez consulter [TerminateEnvironment](#).

### Suppression manuelle du rôle lié à un service

Utilisez la console IAM, l'interface de ligne de commande (CLI) IAM ou l'API IAM pour supprimer le rôle lié à un service AWSServiceRoleForElasticBeanstalk. Pour de plus amples informations, veuillez consulter [Suppression d'un rôle lié à un service](#) dans le Guide de l'utilisateur IAM.

## Régions prises en charge pour les rôles liés à un service Elastic Beanstalk

Elastic Beanstalk prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour de plus amples informations, veuillez consulter [Points de terminaison et quotas AWS Elastic Beanstalk](#).

## Rôle lié à un service de maintenance

AWS Elastic Beanstalk utilise des [rôles AWS Identity and Access Management \(IAM\) liés à un service](#). Un rôle lié à un service est un type unique de rôle IAM lié directement à Elastic Beanstalk. Les rôles liés

à un service sont prédéfinis par Elastic Beanstalk et comprennent toutes les autorisations nécessaires au service pour appeler d'autres services AWS en votre nom.

Un rôle lié à un service simplifie la configuration d'Elastic Beanstalk, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Elastic Beanstalk définit les autorisations de ses rôles liés à un service et, sauf indication contraire, seul Elastic Beanstalk peut assumer ses rôles. Les autorisations définies comprennent la stratégie d'approbation et la stratégie d'autorisation. De plus, cette stratégie d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Votre ressources Elastic Beanstalk sont ainsi protégées car vous ne pouvez pas involontairement supprimer d'autorisation pour accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez [Services AWS qui fonctionnent avec IAM](#) et recherchez les services avec un Oui dans la colonne Rôle lié à un service. Choisissez un Oui ayant un lien permettant de consulter la documentation du rôle lié à un service, pour ce service.

## Autorisations des rôles liés à un service pour Elastic Beanstalk

Elastic Beanstalk utilise le rôle lié à un service AWSServiceRoleForElasticBeanstalkMaintenance – Permet à Elastic Beanstalk d'effectuer régulièrement des activités de maintenance pour vos environnements en cours d'exécution.

Le rôle lié à un service AWSServiceRoleForElasticBeanstalkMaintenance approuve les services suivants pour assumer le rôle :

- `maintenance.elasticbeanstalk.amazonaws.com`

La stratégie d'autorisations du rôle lié à un service AWSServiceRoleForElasticBeanstalkMaintenance contient toutes les autorisations dont Elastic Beanstalk a besoin pour exécuter des actions en votre nom :

```
{  
    "Version": "2012-10-17",  
    "Statement":  
    {  
        "Sid": "AllowCloudformationChangeSetOperationsOnElasticBeanstalkStacks",  
        "Effect": "Allow",  
        "Action": [  
            "cloudformation>CreateChangeSet",  
            "cloudformation>DescribeChangeSet",  
            "cloudformation>ExecuteChangeSet",  
            "cloudformation>DeleteChangeSet",  
            "cloudformation>ListChangeSets",  
            "cloudformation>DescribeStacks"  
        ],  
        "Resource": [  
            "arn:aws:cloudformation:*::stack/awseb-*",  
            "arn:aws:cloudformation:*::stack/eb-*"  
        ]  
    }  
}
```

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour de plus amples informations, veuillez consulter [Autorisations de rôles liés à un service](#) dans le Guide de l'utilisateur IAM.

Sinon, vous pouvez utiliser une stratégie gérée par AWS pour [fournir un accès complet \(p. 946\)](#) à Elastic Beanstalk.

## Création d'un rôle lié à un service pour Elastic Beanstalk

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de profil d'instance, Elastic Beanstalk crée un rôle lié à un service en votre nom.

### Important

Ce rôle lié à un service peut apparaître dans votre compte si vous avez effectué une action dans un autre service qui utilise les fonctions prises en charge par ce rôle. Si vous utilisez le service Elastic Beanstalk avant le 18 avril 2019, date à laquelle le service a commencé à prendre en charge le rôle lié à un service AWS*ServiceRoleForElasticBeanstalkMaintenance*, et que votre compte en avait besoin, Elastic Beanstalk a créé le rôle *AWS*ServiceRoleForElasticBeanstalkMaintenance** dans votre compte. Pour de plus amples informations, veuillez consulter [Un nouveau rôle est apparu dans mon compte IAM](#).

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk et que vous ne spécifiez pas de profil d'instance, Elastic Beanstalk crée à nouveau un rôle lié à un service en votre nom.

## Modification d'un rôle lié à un service pour Elastic Beanstalk

Elastic Beanstalk ne vous permet pas de modifier le rôle lié à un service *AWS*ServiceRoleForElasticBeanstalkMaintenance**. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas modifier le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour de plus amples informations, veuillez consulter [Modification d'un rôle lié à un service](#) dans le Guide de l'utilisateur IAM.

## Suppression d'un rôle lié à un service pour Elastic Beanstalk

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

### Nettoyage d'un rôle lié à un service

Avant que vous puissiez utiliser IAM pour supprimer un rôle lié à un service, vous devez d'abord arrêter tous les environnements Elastic Beanstalk qui utilisent le rôle.

#### Note

Si le service Elastic Beanstalk utilise le rôle lié à un service lorsque vous essayez d'arrêter les environnements, l'arrêt peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour arrêter un environnement Elastic Beanstalk qui utilise le rôle *AWS*ServiceRoleForElasticBeanstalkMaintenance** (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Terminate environment (Arrêter l'environnement).

4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Consultez [eb terminate \(p. 1120\)](#) pour plus de détails sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'interface de ligne de commande (CLI) EB.

Pour de plus amples informations sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'API, veuillez consulter [TerminateEnvironment](#).

#### Suppression manuelle du rôle lié à un service

Utilisez la console IAM, l'interface de ligne de commande (CLI) IAM ou l'API IAM pour supprimer le rôle lié à un service AWSServiceRoleForElasticBeanstalkMaintenance. Pour de plus amples informations, veuillez consulter [Suppression d'un rôle lié à un service](#) dans le Guide de l'utilisateur IAM.

### Régions prises en charge pour les rôles liés à un service Elastic Beanstalk

Elastic Beanstalk prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour de plus amples informations, veuillez consulter [Points de terminaison et quotas AWS Elastic Beanstalk](#).

## Rôle lié à un service de mises à jour gérées

AWS Elastic Beanstalk utilise des [rôles AWS Identity and Access Management \(IAM\) liés à un service](#). Un rôle lié à un service est un type unique de rôle IAM lié directement à Elastic Beanstalk. Les rôles liés à un service sont prédéfinis par Elastic Beanstalk et comprennent toutes les autorisations nécessaires au service pour appeler d'autres services AWS en votre nom.

Un rôle lié à un service simplifie la configuration d'Elastic Beanstalk, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Elastic Beanstalk définit les autorisations de ses rôles liés à un service et, sauf indication contraire, seul Elastic Beanstalk peut assumer ses rôles. Les autorisations définies comprennent la stratégie d'approbation et la stratégie d'autorisation. De plus, cette stratégie d'autorisation ne peut pas être attachée à une autre entité IAM.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Votre ressources Elastic Beanstalk sont ainsi protégées car vous ne pouvez pas involontairement supprimer d'autorisation pour accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés à un service, consultez [Services AWS qui fonctionnent avec IAM](#) et recherchez les services avec un Oui dans la colonne Rôle lié à un service. Choisissez un Oui ayant un lien permettant de consulter la documentation du rôle lié à un service, pour ce service.

### Autorisations des rôles liés à un service pour Elastic Beanstalk

Elastic Beanstalk utilise le rôle lié à un service nommé AWSServiceRoleForElasticBeanstalkManagedUpdates – Permet à Elastic Beanstalk d'effectuer des mises à jour planifiées de vos environnements en cours d'exécution.

Le rôle lié à un service AWSServiceRoleForElasticBeanstalkManagedUpdates approuve les services suivants pour assumer le rôle :

- managedupdates.elasticbeanstalk.amazonaws.com

La stratégie d'autorisations du rôle lié à un service AWSServiceRoleForElasticBeanstalkManagedUpdates contient toutes les autorisations dont Elastic Beanstalk a besoin pour effectuer des actions de mises à jour gérées en votre nom :

{

```
"Version": "2012-10-17",
"Statement": [
    {
        "Sid": "AllowPassRoleToElasticBeanstalkAndDownstreamServices",
        "Effect": "Allow",
        "Action": "iam:PassRole",
        "Resource": "*",
        "Condition": {
            "StringLikeIfExists": {
                "iam:PassedToService": [
                    "elasticbeanstalk.amazonaws.com",
                    "ec2.amazonaws.com",
                    "autoscaling.amazonaws.com",
                    "elasticloadbalancing.amazonaws.com",
                    "ecs.amazonaws.com",
                    "cloudformation.amazonaws.com"
                ]
            }
        }
    },
    {
        "Sid": "SingleInstanceAPIs",
        "Effect": "Allow",
        "Action": [
            "ec2:releaseAddress",
            "ec2:allocateAddress",
            "ec2:DisassociateAddress",
            "ec2:AssociateAddress"
        ],
        "Resource": "*"
    },
    {
        "Sid": "ECS",
        "Effect": "Allow",
        "Action": [
            "ecs:RegisterTaskDefinition",
            "ecs:DeRegisterTaskDefinition",
            "ecs>List*",
            "ecs:Describe*"
        ],
        "Resource": "*"
    },
    {
        "Sid": "ElasticBeanstalkAPIs",
        "Effect": "Allow",
        "Action": [
            "elasticbeanstalk:)"
        ],
        "Resource": "*"
    },
    {
        "Sid": "ReadOnlyAPIs",
        "Effect": "Allow",
        "Action": [
            "cloudformation:Describe*",
            "cloudformation>List*",
            "ec2:Describe*",
            "autoscaling:Describe*",
            "elasticloadbalancing:Describe*"
        ],
        "Resource": "*"
    },
    {
        "Sid": "ASG",
        "Effect": "Allow",
        "Action": [
```

```
        "autoscaling:AttachInstances",
        "autoscaling>CreateAutoScalingGroup",
        "autoscaling>CreateLaunchConfiguration",
        "autoscaling>DeleteAutoScalingGroup",
        "autoscaling>DeleteLaunchConfiguration",
        "autoscaling>DeleteScheduledAction",
        "autoscaling>DetachInstances",
        "autoscaling>PutNotificationConfiguration",
        "autoscaling>PutScalingPolicy",
        "autoscaling>PutScheduledUpdateGroupAction",
        "autoscaling>ResumeProcesses",
        "autoscaling>SuspendProcesses",
        "autoscaling>TerminateInstanceInAutoScalingGroup",
        "autoscaling>UpdateAutoScalingGroup"
    ],
    "Resource": [
        "arn:aws:autoscaling::::launchConfiguration:::launchConfigurationName/
awseb-e-*",
        "arn:aws:autoscaling::::autoScalingGroup:::autoScalingGroupName/awseb-e-*"
    ]
},
{
    "Sid": "CFN",
    "Effect": "Allow",
    "Action": [
        "cloudformation>CreateStack",
        "cloudformation>DeleteStack",
        "cloudformation>GetTemplate",
        "cloudformation>UpdateStack"
    ],
    "Resource": "arn:aws:cloudformation::::stack/awseb-e-*"
},
{
    "Sid": "EC2",
    "Effect": "Allow",
    "Action": [
        "ec2>TerminateInstances"
    ],
    "Resource": "arn:aws:ec2::::instance/*",
    "Condition": {
        "StringLike": {
            "ec2:ResourceTag/aws:cloudformation:stack-id":
"arn:aws:cloudformation::::stack/awseb-e-*"
        }
    }
},
{
    "Sid": "S3Obj",
    "Effect": "Allow",
    "Action": [
        "s3>DeleteObject",
        "s3>GetObject",
        "s3>GetObjectAcl",
        "s3>GetObjectVersion",
        "s3>GetObjectVersionAcl",
        "s3>PutObject",
        "s3>PutObjectAcl",
        "s3>PutObjectVersionAcl"
    ],
    "Resource": "arn:aws:s3::::elasticbeanstalk-*/"
},
{
    "Sid": "S3Bucket",
    "Effect": "Allow",
    "Action": [
        "s3>GetBucketLocation",
        "s3>GetBucketAcl",
        "s3>GetBucketEncryption",
        "s3>GetBucketCors",
        "s3>GetBucketLifecycle",
        "s3>GetBucketMetricsConfiguration",
        "s3>GetBucketPolicy",
        "s3>GetBucketReplication",
        "s3>GetBucketTagging",
        "s3>GetBucketVersioning",
        "s3>GetBucketWebsite"
    ],
    "Resource": "arn:aws:s3::::elasticbeanstalk-*/"
}
]
```

```
        "s3:GetBucketPolicy",
        "s3>ListBucket",
        "s3:PutBucketPolicy"
    ],
    "Resource": "arn:aws:s3:::elasticbeanstalk-*"
},
{
    "Sid": "CWL",
    "Effect": "Allow",
    "Action": [
        "logs>CreateLogGroup",
        "logs>DeleteLogGroup",
        "logs:PutRetentionPolicy"
    ],
    "Resource": "arn:aws:logs:**:log-group:/aws/elasticbeanstalk/*"
},
{
    "Sid": "ELB",
    "Effect": "Allow",
    "Action": [
        "elasticloadbalancing:RegisterTargets",
        "elasticloadbalancing:DeRegisterTargets",
        "elasticloadbalancing:DeregisterInstancesFromLoadBalancer",
        "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
    ],
    "Resource": [
        "arn:aws:elasticloadbalancing:**:targetgroup/awseb-*",
        "arn:aws:elasticloadbalancing:**:loadbalancer/awseb-e-*"
    ]
}
]
```

Vous devez configurer les autorisations de manière à permettre à une entité IAM (comme un utilisateur, un groupe ou un rôle) de créer, modifier ou supprimer un rôle lié à un service. Pour de plus amples informations, veuillez consulter [Autorisations de rôles liés à un service](#) dans le Guide de l'utilisateur IAM.

Sinon, vous pouvez utiliser une stratégie gérée par AWS pour [fournir un accès complet \(p. 946\)](#) à Elastic Beanstalk.

## Création d'un rôle lié à un service pour Elastic Beanstalk

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk, activez les mises à jour gérées et que vous spécifiez `AWSServiceRoleForElasticBeanstalkManagedUpdates` comme valeur pour l'option `ServiceRoleForManagedUpdates` de l'espace de noms [aws:elasticbeanstalk:managedactions \(p. 707\)](#), Elastic Beanstalk crée le rôle lié à un service pour vous.

Quand Elastic Beanstalk tente de créer le rôle lié à un service `AWSServiceRoleForElasticBeanstalkManagedUpdates` pour votre compte lorsque vous créez un environnement, vous devez disposer de l'autorisation `iam>CreateServiceLinkedRole`. Si vous n'avez pas cette autorisation, la création d'environnement échoue, et un message expliquant le problème s'affiche.

En lieu et place, un autre utilisateur ayant l'autorisation de créer des rôles liés à un service peut utiliser IAM pour précréer le rôle lié à un service à l'avance. Vous pouvez ensuite créer votre environnement, même sans avoir l'autorisation `iam>CreateServiceLinkedRole`.

Vous pouvez utiliser la console IAM pour créer un rôle lié à un service avec le cas d'utilisation Elastic Beanstalk Managed Updates (Mises à jour gérées Elastic Beanstalk). Dans l'interface de ligne de commande (CLI) IAM ou l'API IAM, créez un rôle lié à un service avec le nom de service `managedupdates.elasticbeanstalk.amazonaws.com`. Pour de plus amples informations, veuillez

consulter [Création d'un rôle lié à un service](#) dans le Guide de l'utilisateur IAM. Si vous supprimez ce rôle lié à un service, vous pouvez utiliser ce même processus pour créer le rôle à nouveau.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un environnement Elastic Beanstalk à l'aide de l'API Elastic Beanstalk, activez les mises à jour gérées et que vous spécifiez `AWSServiceRoleForElasticBeanstalkManagedUpdates` comme valeur pour l'option `ServiceRoleForManagedUpdates` de l'espace de noms `aws:elasticbeanstalk:managedactions` (p. 707), Elastic Beanstalk crée à nouveau le rôle lié à un service pour vous.

## Modification d'un rôle lié à un service pour Elastic Beanstalk

Elastic Beanstalk ne vous permet pas de modifier le rôle lié à un service `AWSServiceRoleForElasticBeanstalkManagedUpdates`. Une fois que vous avez créé un rôle lié à un service, vous ne pouvez pas modifier le nom du rôle, car plusieurs entités peuvent faire référence à ce rôle. Néanmoins, vous pouvez modifier la description du rôle à l'aide d'IAM. Pour de plus amples informations, veuillez consulter [Modification d'un rôle lié à un service](#) dans le Guide de l'utilisateur IAM.

## Suppression d'un rôle lié à un service pour Elastic Beanstalk

Si vous n'avez plus besoin d'utiliser une fonction ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer les ressources de votre rôle lié à un service avant de pouvoir les supprimer manuellement.

### Nettoyage d'un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez d'abord vous assurer que les environnements Elastic Beanstalk pour lesquels les mises à jour gérées sont activées utilisent un rôle de service différent ou sont arrêtés.

#### Note

Si le service Elastic Beanstalk utilise le rôle lié à un service lorsque vous essayez d'arrêter les environnements, l'arrêt peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour arrêter un environnement Elastic Beanstalk qui utilise le rôle `AWSServiceRoleForElasticBeanstalkManagedUpdates` (console)

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Actions, puis Terminate Environment (Résilier l'environnement).
4. Utilisez la boîte de dialogue à l'écran pour confirmer la résiliation de l'environnement.

Consultez [eb terminate \(p. 1120\)](#) pour plus de détails sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'interface de ligne de commande (CLI) EB.

Pour de plus amples informations sur l'arrêt d'un environnement Elastic Beanstalk à l'aide de l'API, veuillez consulter [TerminateEnvironment](#).

## Suppression manuelle du rôle lié à un service

Utilisez la console IAM, l'interface de ligne de commande (CLI) IAM ou l'API IAM pour supprimer le rôle lié à un service AWS*ServiceRoleForElasticBeanstalkManagedUpdates*. Pour de plus amples informations, veuillez consulter [Suppression d'un rôle lié à un service](#) dans le Guide de l'utilisateur IAM.

## Régions prises en charge pour les rôles liés à un service Elastic Beanstalk

Elastic Beanstalk prend en charge l'utilisation des rôles liés à un service dans toutes les régions où le service est disponible. Pour de plus amples informations, veuillez consulter [Points de terminaison et quotas AWS Elastic Beanstalk](#).

# Gestion des stratégies utilisateur Elastic Beanstalk

AWS Elastic Beanstalk fournit deux stratégies gérées qui vous permettent d'attribuer un accès complet ou un accès en lecture seule à toutes les ressources gérées par Elastic Beanstalk. Vous pouvez attacher les stratégies à des utilisateurs ou à des groupes AWS Identity and Access Management (IAM) ou à des rôles assumés par vos utilisateurs.

### Important

Les stratégies gérées par Elastic Beanstalk ne fournissent pas d'autorisations granulaires : elles accordent toutes les autorisations potentiellement nécessaires à l'utilisation des applications Elastic Beanstalk. Nos stratégies gérées ne couvrent pas non plus les autorisations relatives aux ressources personnalisées que vous pourriez ajouter à votre solution et qui ne sont pas gérées par Elastic Beanstalk. Pour implémenter des autorisations plus granulaires, des autorisations minimales requises ou des autorisations de ressources personnalisées, utilisez des [stratégies personnalisées](#) (p. 947).

### Stratégies utilisateur gérées

- `AdministratorAccessAWSElasticBeanstalk` : accorde à l'utilisateur les autorisations administratives complètes pour créer, modifier et supprimer des applications, des versions d'application, des paramètres de configuration, des environnements Elastic Beanstalk et leurs ressources sous-jacentes.
- `AWSElasticBeanstalkReadOnly` : permet à l'utilisateur d'afficher les applications et les environnements, mais pas d'effectuer des opérations qui les modifient. Elle fournit un accès en lecture seule à toutes les ressources Elastic Beanstalk et aux autres ressources AWS récupérées par la console Elastic Beanstalk. Notez que l'accès en lecture seule ne permet pas d'effectuer des actions telles que le téléchargement des journaux Elastic Beanstalk en vue de les lire. Cela est dû au fait que les journaux sont indexés dans le compartiment Amazon S3, sur lequel Elastic Beanstalk demanderait une autorisation en écriture. Pour de plus amples informations sur l'activation de l'accès aux journaux Elastic Beanstalk, veuillez consulter l'exemple situé à la fin de cette rubrique.

Pour afficher le contenu d'une stratégie gérée, utilisez la [page Policies \(Stratégies\)](#) de la console IAM.

### Note

Auparavant, Elastic Beanstalk prenait en charge deux autres stratégies utilisateur gérées, `AWSElasticBeanstalkFullAccess` et `AWSElasticBeanstalkReadOnlyAccess`. Nous prévoyons de retirer ces anciennes stratégies. Vous pouvez toujours les voir et les utiliser dans la console IAM. Néanmoins, nous vous recommandons de passer aux nouvelles stratégies utilisateur gérées et d'ajouter des stratégies personnalisées pour accorder des autorisations aux ressources personnalisées, le cas échéant.

## Contrôle de l'accès à l'aide de stratégies gérées

Vous pouvez utiliser des stratégies gérées pour accorder un accès complet ou en lecture seule à Elastic Beanstalk. Elastic Beanstalk met à jour ces stratégies automatiquement lorsque des autorisations supplémentaires sont nécessaires pour accéder aux nouvelles fonctionnalités.

Pour appliquer une stratégie gérée à des utilisateurs ou groupes IAM

1. Ouvrez la [page Stratégies dans la console IAM](#).
2. Dans la zone de recherche, tapez **AWSElasticBeanstalk** pour filtrer les stratégies.
3. Dans la liste des stratégies, cochez la case en regard de AWSElasticBeanstalkReadOnly ou de AdministratorAccess-AWSElasticBeanstalk.
4. Choisissez Actions de stratégie, puis choisissez Attacher.
5. Sélectionnez un ou plusieurs utilisateurs et groupes auxquelles attacher la stratégie. Vous pouvez utiliser le menu Filtre et la zone de recherche pour filtrer la liste des entités mandataires.
6. Choisissez Attacher la stratégie.

## Création d'une stratégie utilisateur personnalisée

Vous pouvez créer votre propre stratégie IAM pour autoriser ou refuser des actions d'API propres à Elastic Beanstalk sur des ressources Elastic Beanstalk spécifique et pour contrôler l'accès aux ressources personnalisées qui ne sont pas gérées par Elastic Beanstalk. Pour en savoir plus sur l'association d'une stratégie à un utilisateur ou à un groupe, veuillez consulter la section [Utilisation de stratégies](#) du guide de l'utilisateur IAM. Pour en savoir plus sur la création d'une stratégie personnalisée, veuillez consulter la section [Création de stratégies IAM](#) du guide de l'utilisateur IAM.

### Note

Vous pouvez limiter la façon dont un utilisateur interagit avec l'API Elastic Beanstalk. Toutefois, aucune méthode efficace ne peut empêcher les utilisateurs autorisés à créer les ressources sous-jacentes nécessaires de créer d'autres ressources dans Amazon EC2 et d'autres services. Vous devez considérer ces stratégies comme un moyen efficace pour répartir les responsabilités Elastic Beanstalk, et non comme un moyen de sécuriser toutes les ressources sous-jacentes.

En novembre 2019, Elastic Beanstalk a publié la prise en charge des [modèles de lancement Amazon EC2](#). Il s'agit d'un nouveau type de ressource que le groupe Auto Scaling de votre environnement peut utiliser pour lancer des instances Amazon EC2 ; il nécessite de nouvelles autorisations. La plupart des clients ne devraient pas être affectés, car les environnements peuvent toujours utiliser la ressource héritée, lancer des configurations, si votre stratégie utilisateur ne dispose pas des autorisations requises. Toutefois, si vous essayez d'utiliser une nouvelle fonctionnalité qui nécessite des modèles de lancement Amazon EC2 et que vous disposez d'une stratégie personnalisée, la création ou la mise à jour de votre environnement peut échouer. Dans ce cas, assurez-vous que votre stratégie personnalisée dispose des autorisations suivantes.

### Autorisations requises pour les modèles de lancement Amazon EC2

- `EC2:CreateLaunchTemplate`
- `EC2:CreateLaunchTemplateVersions`
- `EC2>DeleteLaunchTemplate`
- `EC2>DeleteLaunchTemplateVersions`
- `EC2:DescribeLaunchTemplate`
- `EC2:DescribeLaunchTemplateVersions`

Une stratégie IAM contient des déclarations de stratégie décrivant les autorisations que vous souhaitez accorder. Lorsque vous créez une déclaration de stratégie pour Elastic Beanstalk, vous devez comprendre comment utiliser les quatre parties suivantes d'une déclaration de stratégie :

- Effect indique s'il faut autoriser ou refuser les actions spécifiées dans la déclaration.

- Action indique les [opérations d'API](#) que vous souhaitez contrôler. Par exemple, utilisez `elasticbeanstalk:CreateEnvironment` pour spécifier l'opération `CreateEnvironment`. Certaines opérations, telles que la création d'un environnement, nécessitent des autorisations supplémentaires pour effectuer ces actions. Pour de plus amples informations, veuillez consulter [Ressources et conditions pour les actions Elastic Beanstalk \(p. 952\)](#).

#### Note

Pour utiliser l'opération d'API `UpdateTagsForResource`, spécifiez l'une des deux actions virtuelles suivantes (ou les deux) au lieu du nom de l'opération d'API :

`elasticbeanstalk:AddTags`

Contrôle l'autorisation d'appeler `UpdateTagsForResource` et de transmettre la liste des balises à ajouter dans le paramètre `TagsToAdd`.

`elasticbeanstalk:RemoveTags`

Contrôle l'autorisation d'appeler `UpdateTagsForResource` et de transmettre une liste de clés de balises à supprimer dans le paramètre `TagsToRemove`.

- Resource spécifie les ressources auxquelles vous souhaitez contrôler l'accès. Pour spécifier des ressources Elastic Beanstalk, répertoriez [l'Amazon Resource Name \(p. 951\)](#) (ARN) de chaque ressource.
- (Facultatif) Condition indique les restrictions concernant l'autorisation accordée dans la déclaration. Pour de plus amples informations, veuillez consulter [Ressources et conditions pour les actions Elastic Beanstalk \(p. 952\)](#).

Les sections suivantes illustrent quelques cas dans lesquels vous pouvez envisager d'utiliser une stratégie d'utilisateur personnalisée.

## Activation de la création d'un environnement Elastic Beanstalk limité

L'exemple de stratégie suivant permet à un utilisateur appliquant cette stratégie d'appeler l'action `CreateEnvironment` afin de créer un environnement dont le nom commence par `Test`, avec l'application et la version d'application spécifiées.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "CreateEnvironmentPerm",  
            "Action": [  
                "elasticbeanstalk:CreateEnvironment"  
            ],  
            "Effect": "Allow",  
            "Resource": [  
                "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My First Elastic Beanstalk Application/Test*"  
            ],  
            "Condition": {  
                "StringEquals": {  
                    "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My First Elastic Beanstalk Application"],  
                    "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My First Elastic Beanstalk Application/First Release"]  
                }  
            }  
        },  
        {  
            "Sid": "AllNonResourceCalls",  
            "Action": [  
            ]  
        }  
    ]  
}
```

```
        "elasticbeanstalk:CheckDNSAvailability",
        "elasticbeanstalk>CreateStorageLocation"
    ],
    "Effect": "Allow",
    "Resource": [
        "*"
    ]
}
]
```

La stratégie ci-dessus montre comment accorder un accès limité aux opérations Elastic Beanstalk. Pour pouvoir lancer réellement un environnement, l'utilisateur doit avoir l'autorisation de créer les ressources AWS qui alimentent également l'environnement. Par exemple, la stratégie suivante accorde l'accès à l'ensemble de ressources par défaut pour un environnement de serveur web :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:*",
                "ecs:*",
                "elasticloadbalancing:*",
                "autoscaling:*",
                "cloudwatch:*",
                "s3:*",
                "sns:*",
                "cloudformation:*",
                "sqs:*
```

## Activation de l'accès aux journaux Elastic Beanstalk stockés dans Amazon S3

La stratégie de l'exemple suivant permet à un utilisateur d'extraire des journaux Elastic Beanstalk, de les afficher dans Amazon S3 et de les récupérer.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3>DeleteObject",
                "s3>GetObjectAcl",
                "s3>PutObjectAcl"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:s3:::elasticbeanstalk-*"
        }
    ]
}
```

### Note

Pour limiter ces autorisations aux chemins d'accès aux journaux, utilisez le format de ressource suivant.

```
"arn:aws:s3:::elasticbeanstalk-us-east-2-123456789012/resources/environments/logs/*"
```

## Activation de la gestion d'une application Elastic Beanstalk spécifique

L'exemple de stratégie suivant permet à un utilisateur de gérer des environnements et d'autres ressources dans une application Elastic Beanstalk spécifique. La stratégie refuse les actions Elastic Beanstalk sur les ressources des autres applications, et refuse également la création et la suppression des applications Elastic Beanstalk.

### Note

La stratégie ne refuse pas l'accès aux ressources par le biais d'autres services. Elle représente un moyen efficace pour répartir les responsabilités de la gestion des applications Elastic Beanstalk entre les différents utilisateurs, et non un moyen de sécuriser les ressources sous-jacentes.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk>CreateApplication",
        "elasticbeanstalk>DeleteApplication"
      ],
      "Resource": [
        "*"
      ]
    },
    {
      "Effect": "Deny",
      "Action": [
        "elasticbeanstalk>CreateApplicationVersion",
        "elasticbeanstalk>CreateConfigurationTemplate",
        "elasticbeanstalk>CreateEnvironment",
        "elasticbeanstalk>DeleteApplicationVersion",
        "elasticbeanstalk>DeleteConfigurationTemplate",
        "elasticbeanstalk>DeleteEnvironmentConfiguration",
        "elasticbeanstalk>DescribeApplicationVersions",
        "elasticbeanstalk>DescribeConfigurationOptions",
        "elasticbeanstalk>DescribeConfigurationSettings",
        "elasticbeanstalk>DescribeEnvironmentResources",
        "elasticbeanstalk>DescribeEnvironments",
        "elasticbeanstalk>DescribeEvents",
        "elasticbeanstalk>DeleteEnvironmentConfiguration",
        "elasticbeanstalk>RebuildEnvironment",
        "elasticbeanstalk>RequestEnvironmentInfo",
        "elasticbeanstalk>RestartAppServer",
        "elasticbeanstalk>RetrieveEnvironmentInfo",
        "elasticbeanstalk>SwapEnvironmentCNAMEs",
        "elasticbeanstalk>TerminateEnvironment",
        "elasticbeanstalk>UpdateApplicationVersion",
        "elasticbeanstalk>UpdateConfigurationTemplate",
        "elasticbeanstalk>UpdateEnvironment",
        "elasticbeanstalk>RetrieveEnvironmentInfo",
        "elasticbeanstalk>ValidateConfigurationSettings"
      ],
      "Resource": [
        "*"
      ],
      "Condition": {
        "StringNotEquals": {
          "elasticbeanstalk>InApplication": [
            "arn:aws:s3:::elasticbeanstalk-us-east-2-123456789012/resources/environments/logs/*"
          ]
        }
      }
    }
  ]
}
```

```
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/myapplication"
    ]
}
}
```

## Format Amazon Resource Name (ARN) pour Elastic Beanstalk

Vous pouvez spécifier une ressource pour une stratégie IAM via l'Amazon Resource Name (ARN) de cette ressource. Pour Elastic Beanstalk, l'ARN a le format suivant.

```
arn:aws:elasticbeanstalk:region:account-id:resource-type/resource-path
```

Où :

- **region** est la région dans laquelle se situe la ressource (par exemple, **us-west-2**).
- **account-id** est l'ID du compte AWS, sans trait d'union (par exemple, **123456789012**).
- **resource-type** identifie le type de la ressource Elastic Beanstalk, par exemple, **environment**. Consultez la liste de tous les types de ressource Elastic Beanstalk dans le tableau ci-dessous.
- **resource-path** est la partie identifiant la ressource spécifique. Une ressource Elastic Beanstalk comporte un chemin qui identifie cette ressource de façon unique. Consultez le tableau ci-dessous afin de connaître le format du chemin de ressource pour chaque type de ressource. Par exemple, un environnement est toujours associé à une application. Le chemin de ressource pour l'environnement **myEnvironment** de l'application **myApp** se présente comme suit :

```
myApp/myEnvironment
```

Elastic Beanstalk comporte plusieurs types de ressources, que vous pouvez spécifier dans une stratégie. Le tableau suivant affiche le format ARN de chacun d'eux, accompagné d'un exemple.

Type de ressource	Format de l'ARN
application	<b>arn:aws:elasticbeanstalk:<b>region</b>:<b>account-id</b>:application/<b>application-name</b></b>  Exemple : <b>arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App</b>
applicationversion	<b>arn:aws:elasticbeanstalk:<b>region</b>:<b>account-id</b>:applicationversion/<b>application-name/version-label</b></b>  Exemple : <b>arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version</b>
configurationtemplate	<b>arn:aws:elasticbeanstalk:<b>region</b>:<b>account-id</b>:configurationtemplate/<b>application-name/template-name</b></b>  Exemple : <b>arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template</b>

Type de ressource	Format de l'ARN
environment	<code>arn:aws:elasticbeanstalk:<i>region:account-id:environment</i>/<i>application-name/environment-name</i></code>  Exemple : <code>arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/MyEnvironment</code>
platform	<code>arn:aws:elasticbeanstalk:<i>region:account-id:platform-platform-name/platform-version</i></code>  Exemple : <code>arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/MyPlatform/1.0</code>
solutionstack	<code>arn:aws:elasticbeanstalk:<i>region::solutionstack/solutionstack-name</i></code>  Exemple : <code>arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7</code>

Une application spécifique contient toujours un environnement, une version d'application et un modèle de configuration. Vous remarquerez que ces ressources comportent toutes un nom d'application dans leur chemin de ressource, ce qui permet de les identifier de façon unique par leur nom de ressource et l'application qu'elles contiennent. Même si les piles de solutions sont utilisées par les environnements et les modèles de configuration, elles ne sont pas spécifiques à une application ou un compte AWS, et leur ARN ne comporte pas l'application ni le compte AWS.

## Ressources et conditions pour les actions Elastic Beanstalk

Cette section décrit les ressources et les conditions que vous pouvez utiliser dans les déclarations de stratégie pour accorder des autorisations qui permettent la réalisation d'actions Elastic Beanstalk sur des ressources Elastic Beanstalk spécifiques.

Ces conditions vous permettent de spécifier des autorisations pour les ressources que l'action doit réaliser. Par exemple, quand vous pouvez appeler l'action `CreateEnvironment`, vous devez également spécifier la version d'application à déployer, ainsi que l'application qui contient le nom de cette application. Lorsque vous définissez des autorisations pour l'action `CreateEnvironment`, vous spécifiez l'application et la version de l'application sur lesquelles vous souhaitez que l'action agissent en utilisant les conditions `InApplication` et `FromApplicationVersion`.

En outre, vous pouvez spécifier la configuration de l'environnement avec une pile de solutions (`FromSolutionStack`) ou un modèle de configuration (`FromConfigurationTemplate`). La déclaration de stratégie suivante permet à l'action `CreateEnvironment` de créer un environnement avec le nom `myenv` (spécifié par `Resource`) dans l'application `My App` (spécifiée par la condition `InApplication`) à l'aide de la version d'application `My Version` (`FromApplicationVersion`) avec une configuration `32bit Amazon Linux running Tomcat 7` (`FromSolutionStack`):

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:CreateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/myenv"
    }
  ]
}
```

```

    "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"
    ],
    "Condition": {
        "StringEquals": {
            "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
            "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:applicationversion/My App/My Version"],
            "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-
east-2:solutionstack/32bit Amazon Linux running Tomcat 7"]
        }
    }
]
}

```

### Note

La plupart des clés de condition mentionnées dans cette rubrique sont spécifiques à Elastic Beanstalk, et leurs noms contiennent le préfixe `elasticbeanstalk:`. Par souci de concision, nous omettons ce préfixe dans les noms de clé de condition lorsque nous les mentionnons dans les sections suivantes. Par exemple, nous mentionnons `InApplication` au lieu du nom complet `elasticbeanstalk:InApplication`.

En revanche, nous mentionnons quelques clés de condition utilisées dans les services AWS, et nous incluons leur préfixe `aws:` pour mettre en avant l'exception.

Les exemples de stratégies présentent toujours les noms complets de clé de condition, y compris le préfixe.

### Sections

- [Informations sur les stratégies pour les actions Elastic Beanstalk \(p. 953\)](#)
- [Clés de condition pour les actions Elastic Beanstalk \(p. 975\)](#)

## Informations sur les stratégies pour les actions Elastic Beanstalk

Le tableau suivant répertorie toutes les actions Elastic Beanstalk, la ressource sur laquelle agit chaque action, ainsi que les informations contextuelles supplémentaires qui peuvent être fournies à l'aide de conditions.

Informations de stratégie pour les actions Elastic Beanstalk, y compris des ressources, des conditions, des exemples et des dépendances

Ressource	Conditions	Exemple de déclaration
Action: <a href="#">AbortEnvironmentUpdate</a>		
application environment	<a href="#">aws:ResourceTag/ key-name</a> (Facultatif)  <a href="#">aws:TagKeys</a> (Facultatif)	La stratégie suivante permet à un utilisateur d'annuler des opérations de mise à jour d'environnement sur des environnements dans une application nommée <code>My App</code> .  <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:AbortEnvironmentUpdate"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"       ]     }   ] }</pre>

Ressource	Conditions	Exemple de déclaration
		<pre>"arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App" ] } ]</pre>
<b>Action: <a href="#">CheckDNSAvailability</a></b>		
"*"	N/A	<pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:CheckDNSAvailability"       ],       "Effect": "Allow",       "Resource": "*"     }   ] }</pre>
<b>Action: <a href="#">ComposeEnvironments</a></b>		
application	aws:ResourceTag/ <b>key-name</b> (Facultatif)	La stratégie suivante permet à un utilisateur de composer des environnements qui appartiennent à une application nommée My App.
	aws:TagKeys (Facultatif)	<pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:ComposeEnvironments"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App"       ]     }   ] }</pre>
<b>Action: <a href="#">CreateApplication</a></b>		

Ressource	Conditions	Exemple de déclaration
application	<p><code>aws:RequestTag/ <i>key-name</i></code> (Facultatif)</p> <p><code>aws:TagKeys</code> (Facultatif)</p>	<p>Cet exemple permet à l'action <code>CreateApplication</code> de créer des applications dont les noms commencent par <b>DivA</b>:</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:CreateApplication"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/DivA*"       ]     }   ] }</pre>
Action: <a href="#">CreateApplicationVersion</a>		
applicationversion	<p><code>InApplication</code></p> <p><code>aws:RequestTag/ <i>key-name</i></code> (Facultatif)</p> <p><code>aws:TagKeys</code> (Facultatif)</p>	<p>Cet exemple permet à l'action <code>CreateApplicationVersion</code> de créer des versions d'applications avec n'importe quel nom (*) dans l'application <b>My App</b>:</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:CreateApplicationVersion"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/*"       ],       "Condition": {         "StringEquals": {           "elasticbeanstalk:InApplication": [             "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"           ]         }       }     }   ] }</pre>
Action: <a href="#">CreateConfigurationTemplate</a>		

Ressource	Conditions	Exemple de déclaration
configurationtemplate	<pre>InApplication FromApplication FromApplicationVersion FromConfigurationTemplate FromEnvironment FromSolutionStack aws:RequestTag/ key-name (Facultatif) aws:TagKeys (Facultatif)</pre>	<p>La stratégie suivante autorise l'action <code>CreateConfigurationTemplate</code> à créer des modèles de configuration dont le nom commence par <b>My Template</b> (<code>My Template*</code>) dans l'application <b>My App</b>:</p> <pre>"Version": "2012-10-17", "Statement": [ {     "Action": [         "elasticbeanstalk:CreateConfigurationTemplate"     ],     "Effect": "Allow",     "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template*"     ],     "Condition": {         "StringEquals": {             "elasticbeanstalk:InApplication": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"             ],             "elasticbeanstalk:FromSolutionStack": [                 "arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7"             ]         }     } }]</pre>

Action: [CreateEnvironment](#)

Ressource	Conditions	Exemple de déclaration
environment	InApplication  FromApplicationVersion  FromConfigurationTemplate  FromSolutionStack  aws:RequestTag/ <i>key-name</i> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>CreateEnvironment</code> à créer un environnement dont le nom est <code>myenv</code> dans l'application <code>My App</code> et à l'aide de la pile de solutions <code>32bit Amazon Linux running Tomcat 7</code>:</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:CreateEnvironment"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ],                     "elasticbeanstalk:FromApplicationVersion": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"                     ],                     "elasticbeanstalk:FromSolutionStack": [                         "arn:aws:elasticbeanstalk:us-east-2::solutionstack/32bit Amazon Linux running Tomcat 7"                     ]                 }             }         ]     ] }</pre>
Action: <code>CreatePlatformVersion</code>	platform  aws:RequestTag/ <i>key-name</i> (Facultatif)  aws:TagKeys (Facultatif)	<p>Cet exemple autorise l'action <code>CreatePlatformVersion</code> à créer des versions de plateforme ciblant la région <code>us-east-2</code>, dont le nom commence par <code>us-east-2_</code>:</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:CreatePlatformVersion"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_*"             ]         }     ] }</pre>

Ressource	Conditions	Exemple de déclaration
<b>Action: <a href="#">CreateStorageLocation</a></b>		
"*"	N/A	<pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:CreateStorageLocation"       ],       "Effect": "Allow",       "Resource": "*"     }   ] }</pre>
<b>Action: <a href="#">DeleteApplication</a></b>		
application	aws:ResourceTag/ <i>key-name</i> (Facultatif)	La stratégie suivante autorise l'action <code>DeleteApplication</code> à supprimer l'application <b>My App</b> :
	aws:TagKeys (Facultatif)	<pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:DeleteApplication"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"       ]     }   ] }</pre>
<b>Action: <a href="#">DeleteApplicationVersion</a></b>		

Ressource	Conditions	Exemple de déclaration
applicationversion	application aws:ResourceTag/ <b>key-name</b> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action DeleteApplicationVersion à supprimer une version de l'application dont le nom est <b>My Version</b> dans l'application <b>My App</b> :</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:DeleteApplicationVersion"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         ]     } }</pre>
Action: <a href="#">DeleteConfigurationTemplate</a>		
configurationtemplate	application (Facultatif)  aws:ResourceTag/ <b>key-name</b> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action DeleteConfigurationTemplate à supprimer un modèle de configuration dont le nom est <b>My Template</b> dans l'application <b>My App</b>. Spécifier le nom de l'application en tant que condition est facultatif.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:DeleteConfigurationTemplate"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template"             ]         }     ] }</pre>
Action: <a href="#">DeleteEnvironmentConfiguration</a>		

Ressource	Conditions	Exemple de déclaration
environment	InApplication (Facultatif)	<p>La stratégie suivante autorise l'action <code>DeleteEnvironmentConfiguration</code> à supprimer une configuration de l'ébauche pour l'environnement <b>myenv</b> dans l'application <b>My App</b>. Spécifier le nom de l'application en tant que condition est facultatif.</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:DeleteEnvironmentConfiguration"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"       ]     }   ] }</pre>
Action: <a href="#">DeletePlatformVersion</a>		
platform	<a href="#">aws:ResourceTag/<i>key-name</i></a> (Facultatif)	<p>La stratégie suivante autorise l'action <code>DeletePlatformVersion</code> à supprimer les versions de plateforme ciblant la région <b>us-east-2</b>, dont le nom commence par <b>us-east-2_</b>:</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:DeletePlatformVersion"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_"       ]     }   ] }</pre>
Action: <a href="#">DescribeApplications</a>		

Ressource	Conditions	Exemple de déclaration
application	<p><code>aws:ResourceTag/ <i>key-name</i> (Facultatif)</code></p> <p><code>aws:TagKeys (Facultatif)</code></p>	<p>La stratégie suivante autorise l'action <code>DescribeApplications</code> à décrire l'application My App.</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:DescribeApplications"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"       ]     }   ] }</pre>
Action: <a href="#">DescribeApplicationVersions</a>		
applicationversion	<p><code>ArnApplication (Facultatif)</code></p> <p><code>aws:ResourceTag/ <i>key-name</i> (Facultatif)</code></p> <p><code>aws:TagKeys (Facultatif)</code></p>	<p>La stratégie suivante autorise l'action <code>DescribeApplicationVersions</code> à décrire la version de l'application <b>My Version</b> dans l'application <b>My App</b>. Spécifier le nom de l'application en tant que condition est facultatif.</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:DescribeApplicationVersions"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"       ]     }   ] }</pre>
Action: <a href="#">DescribeConfigurationOptions</a>		

Ressource	Conditions	Exemple de déclaration
environment configurationtemplate solutionstack	InApplication (Facultatif)  aws:ResourceTag/ <i>key-name</i> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>DescribeConfigurationOptions</code> à décrire les options de configuration pour l'environnement <code>myenv</code> dans l'application <code>My App</code>. Spécifier le nom de l'application en tant que condition est facultatif.</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action":         "elasticbeanstalk:DescribeConfigurationOptions",       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"       ]     }   ] }</pre>
Action: <a href="#">DescribeConfigurationSettings</a>		
environment configurationtemplate	InApplication (Facultatif)  aws:ResourceTag/ <i>key-name</i> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>DescribeConfigurationSettings</code> à décrire les paramètres de configuration pour l'environnement <code>myenv</code> dans l'application <code>My App</code>. Spécifier le nom de l'application en tant que condition est facultatif.</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action":         "elasticbeanstalk:DescribeConfigurationSettings",       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"       ]     }   ] }</pre>
Action: <a href="#">DescribeEnvironmentHealth</a>		

Ressource	Conditions	Exemple de déclaration
environment	<p><code>aws:ResourceTag/ <i>key-name</i></code> (Facultatif)</p> <p><code>aws:TagKeys</code> (Facultatif)</p>	<p>La stratégie suivante permet l'utilisation de <code>DescribeEnvironmentHealth</code> pour récupérer des informations d'intégrité pour un environnement nommé <code>myenv</code>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action":                 "elasticbeanstalk:DescribeEnvironmentHealth",             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us- east-2:123456789012:environment/My App/myenv"             ]         }     ] }</pre>
Action: <a href="#">DescribeEnvironmentResources</a>		
environment	<p><code>InApplication</code> (Facultatif)</p> <p><code>aws:ResourceTag/ <i>key-name</i></code> (Facultatif)</p> <p><code>aws:TagKeys</code> (Facultatif)</p>	<p>La stratégie suivante autorise l'action <code>DescribeEnvironmentResources</code> à renvoyer une liste de ressources AWS pour l'environnement <code>myenv</code> dans l'application <code>My App</code>. Spécifier le nom de l'application en tant que condition est facultatif.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action":                 "elasticbeanstalk:DescribeEnvironmentResources",             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us- east-2:123456789012:environment/My App/myenv"             ]         }     ] }</pre>
Action: <a href="#">DescribeEnvironments</a>		

Ressource	Conditions	Exemple de déclaration
environment	InApplication (Facultatif)  aws:ResourceTag/ <i>key-name</i> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>DescribeEnvironments</code> à décrire les environnements <code>myenv</code> et <code>myotherenv</code> dans l'application <code>My App</code>. Spécifier le nom de l'application en tant que condition est facultatif.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": "elasticbeanstalk:DescribeEnvironments",             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv",                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App2/myotherenv"             ]         }     ] }</pre>
<b>Action: <code>DescribeEvents</code></b>		
application	InApplication	
applicationversion	aws:ResourceTag/ <i>key-name</i>	
configurationtemp	aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>DescribeEvents</code> à répertorier les descriptions de l'événement pour l'environnement <code>myenv</code> et la version de l'application <code>My Version</code> dans l'application <code>My App</code>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": "elasticbeanstalk:DescribeEvents",             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv",                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         }     ] }</pre>
environment	aws:TagKeys (Facultatif)	
<b>Action: <code>DescribeInstancesHealth</code></b>		

Ressource	Conditions	Exemple de déclaration
environment	N/A	<p>La stratégie suivante permet l'utilisation de <code>DescribeInstancesHealth</code> pour récupérer des informations d'intégrité pour des instances dans un environnement nommé <code>myenv</code>.</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action":         "elasticbeanstalk:DescribeInstancesHealth",       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us- east-2:123456789012:environment/My App/myenv"       ]     }   ] }</pre>
Action: <a href="#">DescribePlatformVersion</a>		
platform	<p><code>aws:ResourceTag/ <i>key-name</i></code> (Facultatif)</p> <p><code>aws:TagKeys</code> (Facultatif)</p>	<p>La stratégie suivante autorise l'action <code>DescribePlatformVersion</code> pour décrire les versions de plateforme ciblant la région <code>us-east-2</code>, dont le nom commence par <code>us-east-2_</code>:</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:DescribePlatformVersion"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us- east-2:123456789012:platform/us-east-2_"       ]     }   ] }</pre>
Action: <a href="#">ListAvailableSolutionStacks</a>		

Ressource	Conditions	Exemple de déclaration
solutionstack	N/A	<p>La stratégie suivante autorise l'action <code>ListAvailableSolutionStacks</code> à renvoyer uniquement la pile de solutions <b>32bit Amazon Linux running Tomcat 7</b>.</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk&gt;ListAvailableSolutionStacks"       ],       "Effect": "Allow",       "Resource": "arn:aws:elasticbeanstalk:us-east-2:solutionstack/32bit Amazon Linux running Tomcat 7"     }   ] }</pre>
Action: <a href="#">ListPlatformVersions</a>		
platform	<p><code>aws:RequestTag/ <i>key-name</i></code> (Facultatif)</p> <p><code>aws:TagKeys</code> (Facultatif)</p>	<p>Cet exemple autorise l'action <code>CreatePlatformVersion</code> à créer des versions de plateforme ciblant la région <code>us-east-2</code>, dont le nom commence par <b>us-east-2_</b>:</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk&gt;ListPlatformVersions"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:platform/us-east-2_"       ]     }   ] }</pre>
Action: <a href="#">ListTagsForResource</a>		

Ressource	Conditions	Exemple de déclaration
application applicationversion configurationtemplate environment platform	aws:ResourceTag/ <i>key-name</i> (Facultatif) aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>ListTagsForResource</code> à afficher la liste des balises des environnements existants uniquement s'ils possèdent une balise nommée <code>stage</code> et dont la valeur est <code>test</code> :</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk&gt;ListTagsForResource"             ],             "Effect": "Allow",             "Resource": "*",             "Condition": {                 "StringEquals": {                     "aws:ResourceTag/stage": ["test"]                 }             }         }     ] }</pre>
Action: <a href="#">RebuildEnvironment</a>		
environment	InApplication aws:ResourceTag/ <i>key-name</i> (Facultatif) aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>RebuildEnvironment</code> à recréer l'environnement <code>myenv</code> dans l'application <code>My App</code>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk&gt;RebuildEnvironment"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         }     ] }</pre>
Action: <a href="#">RequestEnvironmentInfo</a>		

Ressource	Conditions	Exemple de déclaration
environment	InApplication  aws:ResourceTag/ <b>key-name</b> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action RequestEnvironmentInfo à compiler des informations sur l'environnement <b>myenv</b> dans l'application <b>My App</b>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:RequestEnvironmentInfo"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         ]     } }</pre>
Action: <a href="#">RestartAppServer</a>		
environment	InApplication	<p>La stratégie suivante autorise l'action RestartAppServer à redémarrer le serveur conteneur d'application pour l'environnement <b>myenv</b> dans l'application <b>My App</b>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:RestartAppServer"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         ]     } }</pre>
Action: <a href="#">RetrieveEnvironmentInfo</a>		

Ressource	Conditions	Exemple de déclaration
environment	InApplication  aws:ResourceTag/ <b>key-name</b> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>RetrieveEnvironmentInfo</code> à récupérer les informations compilées pour l'environnement <code>myenv</code> dans l'application <code>My App</code>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:RetrieveEnvironmentInfo"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         ]     } }</pre>
Action: <a href="#">SwapEnvironmentCNAMEs</a>		
environment	InApplication (Facultatif)  FromEnvironment (Facultatif)	<p>La stratégie suivante autorise l'action <code>SwapEnvironmentCNAMEs</code> à échanger les CNAME pour les environnements <code>mysrcenv</code> et <code>mydestenv</code>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:SwapEnvironmentCNAMEs"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mysrcenv",                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mydestenv"             ]         }     ] }</pre>
Action: <a href="#">TerminateEnvironment</a>		

Ressource	Conditions	Exemple de déclaration
environment	InApplication  aws:ResourceTag/ <i>key-name</i> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action TerminateEnvironment à supprimer l'environnement <b>myenv</b> dans l'application <b>My App</b>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:TerminateEnvironment"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         ]     } }</pre>
Action : <a href="#">UpdateApplication</a>		
application	aws:ResourceTag/ <i>key-name</i> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action UpdateApplication à mettre à jour les propriétés de l'application <b>My App</b>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:UpdateApplication"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"             ]         }     ] }</pre>
Action : <a href="#">UpdateApplicationResourceLifecycle</a>		

Ressource	Conditions	Exemple de déclaration
application	<code>aws:ResourceTag/ key-name</code> (Facultatif)  <code>aws:TagKeys</code> (Facultatif)	<p>La stratégie suivante autorise l'action <code>UpdateApplicationResourceLifecycle</code> à mettre à jour les paramètres du cycle de vie de l'application <b>My App</b>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:UpdateApplicationResourceLifecycle"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"             ]         }     ] }</pre>
Action: <a href="#">UpdateApplicationVersion</a>		
applicationversion	<code>InApplication</code> <code>aws:ResourceTag/ key-name</code> (Facultatif)  <code>aws:TagKeys</code> (Facultatif)	<p>La stratégie suivante autorise l'action <code>UpdateApplicationVersion</code> à mettre à jour les propriétés de la version de l'application <b>My Version</b> dans l'application <b>My App</b>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:UpdateApplicationVersion"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         ]     ] }</pre>
Action: <a href="#">UpdateConfigurationTemplate</a>		

Ressource	Conditions	Exemple de déclaration
configurationtemplate	<b>InApplication</b> aws:ResourceTag <b>key-name</b> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>UpdateConfigurationTemplate</code> à mettre à jour les propriétés ou les options du modèle de configuration <b>My Template</b> dans l'application <b>My App</b>.</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:UpdateConfigurationTemplate"       ],       "Effect": "Allow",       "Resource": [         "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template"       ],       "Condition": {         "StringEquals": {           "elasticbeanstalk:InApplication": [             "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"           ]         }       }     ]   } }</pre>
Action: <a href="#">UpdateEnvironment</a>		

Ressource	Conditions	Exemple de déclaration
environment	InApplication  FromApplicationVersion  FromConfigurationTemplate  aws:ResourceTag/ <i>key-name</i> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action <code>UpdateEnvironment</code> à mettre à jour l'environnement <code>myenv</code> dans l'application <code>My App</code> en déployant la version de l'application <code>My Version</code>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:UpdateEnvironment"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ],                     "elasticbeanstalk:FromApplicationVersion": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"                     ]                 }             }         ]     } }</pre>
Action: <a href="#">UpdateTagsForResource</a> – AddTags		

Ressource	Conditions	Exemple de déclaration
application applicationversion configurationtemplate environment platform	aws:ResourceTag/ <i>key-name</i> (Facultatif) aws:RequestTag/ <i>key-name</i> (Facultatif) aws:TagKeys (Facultatif)	<p>L'action AddTags est l'une des deux actions virtuelles associées à l'API <a href="#">UpdateTagsForResource</a>.</p> <p>La stratégie suivante autorise l'action AddTags à modifier des balises des environnements existants uniquement s'ils possèdent une balise nommée stage et dont la valeur est test :</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:AddTags"       ],       "Effect": "Allow",       "Resource": "*",       "Condition": {         "StringEquals": {           "aws:ResourceTag/stage": ["test"]         }       }     }   ] }</pre>
Action: <a href="#">UpdateTagsForResource</a> – RemoveTags		
application applicationversion configurationtemplate environment platform	aws:ResourceTag/ <i>key-name</i> (Facultatif) aws:RequestTag/ <i>key-name</i> (Facultatif) aws:TagKeys (Facultatif)	<p>L'action RemoveTags est l'une des deux actions virtuelles associées à l'API <a href="#">UpdateTagsForResource</a>.</p> <p>La stratégie suivante empêche l'action RemoveTags de demander la suppression d'une balise nommée stage à partir des environnements existants :</p> <pre>{   "Version": "2012-10-17",   "Statement": [     {       "Action": [         "elasticbeanstalk:RemoveTags"       ],       "Effect": "Deny",       "Resource": "*",       "Condition": {         "ForAnyValue:StringEquals": {           "aws:TagKeys": ["stage"]         }       }     }   ] }</pre>
Action: <a href="#">ValidateConfigurationSettings</a>		

Ressource	Conditions	Exemple de déclaration
template environment	InApplication  aws:ResourceTag/ <b>key-name</b> (Facultatif)  aws:TagKeys (Facultatif)	<p>La stratégie suivante autorise l'action ValidateConfigurationSettings à valider les paramètres de configuration par rapport à l'environnement <b>myenv</b> dans l'application <b>My App</b>.</p> <pre>{     "Version": "2012-10-17",     "Statement": [         {             "Action": [                 "elasticbeanstalk:ValidateConfigurationSettings"             ],             "Effect": "Allow",             "Resource": [                 "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"             ],             "Condition": {                 "StringEquals": {                     "elasticbeanstalk:InApplication": [                         "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"                     ]                 }             }         ]     } }</pre>

## Clés de condition pour les actions Elastic Beanstalk

Les clés vous permettent de spécifier des conditions qui expriment les dépendances, limitent les autorisations ou spécifient les contraintes sur les paramètres d'entrée pour une action. Elastic Beanstalk prend en charge les clés suivantes.

### InApplication

Spécifie l'application qui contient la ressource sur laquelle l'action agit.

L'exemple suivant autorise l'action UpdateApplicationVersion à mettre à jour les propriétés de la version de l'application **My Version**. La condition InApplication spécifie **My App** comme le conteneur pour **My Version**.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "elasticbeanstalk:UpdateApplicationVersion"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"
            ],
            "Condition": {
                "StringEquals": {
```

```
        "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]
    }
}
]
```

#### FromApplicationVersion

Spécifie une version de l'application comme une dépendance ou une contrainte sur un paramètre d'entrée.

L'exemple suivant autorise l'action `UpdateEnvironment` à mettre à jour l'environnement `myenv` dans l'application `My App`. La condition `FromApplicationVersion` contraint le paramètre `VersionLabel` à autoriser uniquement la version de l'application `My Version` pour mettre à jour l'environnement.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:UpdateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"],
          "elasticbeanstalk:FromApplicationVersion": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/My App/My Version"]
        }
      }
    ]
  }
}
```

#### FromConfigurationTemplate

Spécifie un modèle de configuration comme une dépendance ou une contrainte sur un paramètre d'entrée.

L'exemple suivant autorise l'action `UpdateEnvironment` à mettre à jour l'environnement `myenv` dans l'application `My App`. La condition `FromConfigurationTemplate` contraint le paramètre `TemplateName` à autoriser uniquement le modèle de configuration `My Template` pour mettre à jour l'environnement.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "elasticbeanstalk:UpdateEnvironment"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/myenv"
      ],
      "Condition": {
        "StringEquals": {
          "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"]
        }
      }
    ]
  }
}
```

```

    "Condition": {
        "StringEquals": {
            "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:application/My App"],
            "elasticbeanstalk:FromConfigurationTemplate": ["arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/My Template"]
        }
    }
}

```

#### FromEnvironment

Spécifie un environnement comme une dépendance ou une contrainte sur un paramètre d'entrée.

L'exemple suivant autorise l'action `SwapEnvironmentCNAMES` à échanger les CNAME dans **My App** pour tous les environnements dont les noms commencent par `mysrcenv` et `mydestenv`, mais pas les environnements dont les noms commencent par `mysrcenvPROD*` et `mydestenvPROD*`.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "elasticbeanstalk:SwapEnvironmentCNAMES"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mysrcenv*",
                "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mydestenv*"
            ],
            "Condition": {
                "StringNotLike": {
                    "elasticbeanstalk:FromEnvironment": [
                        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mysrcenvPROD*",
                        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/My App/mydestenvPROD*"
                    ]
                }
            }
        ]
    }
}

```

#### FromSolutionStack

Spécifie une pile de solutions comme une dépendance ou une contrainte sur un paramètre d'entrée.

La stratégie suivante autorise l'action `CreateConfigurationTemplate` pour créer des modèles de configuration dont le nom commence par **My Template** (`My Template*`) dans l'application **My App**. La condition `FromSolutionStack` restreint le paramètre `solutionstack` pour autoriser uniquement la pile de solutions **32bit Amazon Linux running Tomcat 7** en tant que valeur d'entrée pour ce paramètre.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "elasticbeanstalk:CreateConfigurationTemplate"
            ]
        }
    ]
}

```

```
        ],
        "Effect": "Allow",
        "Resource": [
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:configurationtemplate/My App/
My Template*"
        ],
        "Condition": {
            "StringEquals": {
                "elasticbeanstalk:InApplication": ["arn:aws:elasticbeanstalk:us-
east-2:123456789012:application/My App"],
                "elasticbeanstalk:FromSolutionStack": ["arn:aws:elasticbeanstalk:us-
east-2::solutionstack/32bit Amazon Linux running Tomcat 7"]
            }
        }
    ]
}
```

`aws:ResourceTag/key-name, aws:RequestTag/key-name, aws:TagKeys`

Spécifiez des conditions basées sur la balise. Pour plus d'informations, consultez [Utilisation de balises pour contrôler l'accès aux ressources Elastic Beanstalk \(p. 978\)](#).

## Utilisation de balises pour contrôler l'accès aux ressources Elastic Beanstalk

Les conditions dans les instructions de stratégie d'utilisateur AWS Identity and Access Management (IAM) font partie de la syntaxe que vous utilisez pour spécifier des autorisations pour les ressources dont les actions Elastic Beanstalk ont besoin pour être menées à bien. Pour de plus amples informations sur la spécification de conditions de déclaration de stratégie, veuillez consulter [Ressources et conditions pour les actions Elastic Beanstalk \(p. 952\)](#). L'utilisation des balises dans les conditions est un moyen de contrôler l'accès aux ressources et demandes. Pour de plus amples informations sur le balisage des ressources Elastic Beanstalk, veuillez consulter [Balisage des ressources d'application Elastic Beanstalk \(p. 423\)](#). Cette rubrique explique le contrôle d'accès basé sur les balises.

Lorsque vous concevez des stratégies IAM, vous pouvez définir des autorisations granulaires en accordant l'accès à des ressources spécifiques. Au fur et à mesure que le nombre de ressources que vous gérez s'accroît, cette tâche devient plus difficile. Le balisage des ressources et l'utilisation de balises dans les déclarations de stratégie peuvent rendre cette tâche plus facile. Vous accordez l'accès en bloc à toute ressource avec une balise spécifique. Puis, vous appliquez cette balise à plusieurs reprises aux ressources correspondantes, lors de la création ou ultérieurement.

Les balises peuvent être attachées à la ressource ou transmises dans la demande aux services qui prennent en charge le balisage. Dans Elastic Beanstalk, les ressources et certaines actions peuvent comporter des balises. Lorsque vous créez une stratégie IAM, vous pouvez utiliser des clés de condition de balise pour contrôler :

- quels utilisateurs peuvent effectuer des actions sur un environnement, en fonction des balises qu'il possède déjà ;
- quelles balises peuvent être transmises dans une demande d'action ;
- si des clés de balise spécifiques peuvent être utilisées dans une demande.

Pour connaître la syntaxe complète et la sémantique des clés de condition de balise, consultez [Contrôle de l'accès à l'aide de balises](#) dans le Guide de l'utilisateur IAM .

Les exemples suivants montrent comment spécifier des conditions de balises dans les stratégies pour les utilisateurs Elastic Beanstalk.

Example 1 : Limiter les actions en fonction des balises dans la demande

La stratégie utilisateur gérée Elastic Beanstalk AdministratorAccess-AWSElasticBeanstalk fournit aux utilisateurs les autorisations illimitées nécessaires pour effectuer une action Elastic Beanstalk sur une ressource gérée par Elastic Beanstalk.

La stratégie suivante refuse aux utilisateurs non autorisés l'autorisation de créer des environnements de production Elastic Beanstalk. Pour ce faire, elle refuse l'action `CreateEnvironment` si la demande spécifie une balise nommée `stage` avec une valeur `gamma` ou `prod`. En outre, la stratégie empêche ces utilisateurs non autorisés de modifier la phase d'environnements de production, en n'autorisant pas les actions de modification de balise visant à inclure ces mêmes valeurs de balise ou à supprimer entièrement la balise `stage`. L'administrateur d'un client peut attacher cette stratégie IAM aux utilisateurs IAM non autorisés et à la stratégie d'utilisateur gérée.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Deny",  
            "Action": [  
                "elasticbeanstalk:CreateEnvironment",  
                "elasticbeanstalk:AddTags"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "aws:RequestTag/stage": ["gamma", "prod"]  
                }  
            }  
        },  
        {  
            "Effect": "Deny",  
            "Action": [  
                "elasticbeanstalk:RemoveTags"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "ForAnyValue:StringEquals": {  
                    "aws:TagKeys": ["stage"]  
                }  
            }  
        }  
    ]  
}
```

Example 2 : Limiter les actions en fonction des balises de ressource

La stratégie utilisateur gérée Elastic Beanstalk AdministratorAccess-AWSElasticBeanstalk fournit aux utilisateurs les autorisations illimitées nécessaires pour effectuer une action Elastic Beanstalk sur une ressource gérée par Elastic Beanstalk.

La stratégie suivante refuse aux utilisateurs non autorisés l'autorisation d'effectuer des actions sur les environnements de production Elastic Beanstalk. Pour ce faire, elle refuse des actions spécifiques si l'environnement possède une balise nommée `stage` avec une valeur `gamma` ou `prod`. L'administrateur d'un client peut attacher cette stratégie IAM aux utilisateurs IAM non autorisés et à la stratégie d'utilisateur gérée.

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Deny",
```

```
"Action": [
    "elasticbeanstalk:AddTags",
    "elasticbeanstalk:RemoveTags",
    "elasticbeanstalk:DescribeEnvironments",
    "elasticbeanstalk:TerminateEnvironment",
    "elasticbeanstalk:UpdateEnvironment",
    "elasticbeanstalk>ListTagsForResource"
],
"Resource": "*",
"Condition": {
    "StringEquals": {
        "aws:ResourceTag/stage": ["gamma", "prod"]
    }
}
]
```

#### Example 3 : Limiter les actions en fonction des balises dans la demande

La stratégie suivante accorde aux utilisateurs l'autorisation de créer des applications de développement Elastic Beanstalk.

Pour ce faire, il autorise les actions `CreateApplication` et `AddTags` si la demande spécifie une balise nommée `stage` avec la valeur `development`. La condition `aws:TagKeys` garantit que l'utilisateur ne peut pas ajouter d'autres clés de balise. En particulier, elle garantit la sensibilité à la casse de la clé de balise `stage`. Notez que cette stratégie est utile pour les utilisateurs Elastic Beanstalk qui n'ont pas de stratégie utilisateur gérée `AdministratorAccess-AWSElasticBeanstalk` attachée. La stratégie gérée fournit aux utilisateurs les autorisations complètes nécessaires pour effectuer une action Elastic Beanstalk sur une ressource gérée par Elastic Beanstalk.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "elasticbeanstalk>CreateApplication",
                "elasticbeanstalk>AddTags"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/stage": "development"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": ["stage"]
                }
            }
        }
    ]
}
```

#### Example 4 : Limiter les actions en fonction des balises de ressource

La stratégie suivante accorde aux utilisateurs l'autorisation d'effectuer des actions et d'obtenir des informations sur les applications de développement Elastic Beanstalk.

Pour ce faire, il autorise les actions spécifiques si l'application a une balise nommée `stage` avec la valeur `development`. La condition `aws:TagKeys` garantit que l'utilisateur ne peut pas ajouter d'autres clés de balise. En particulier, elle garantit la sensibilité à la casse de la clé de balise `stage`. Notez que cette stratégie est utile pour les utilisateurs Elastic Beanstalk qui n'ont pas de stratégie utilisateur

gérée AdministratorAccess-AWSElasticBeanstalk attachée. La stratégie gérée fournit aux utilisateurs les autorisations complètes nécessaires pour effectuer une action Elastic Beanstalk sur une ressource gérée par Elastic Beanstalk.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "elasticbeanstalk:UpdateApplication",  
                "elasticbeanstalk:DeleteApplication",  
                "elasticbeanstalk:DescribeApplications"  
            ],  
            "Resource": "*",  
            "Condition": {  
                "StringEquals": {  
                    "aws:ResourceTag/stage": "development"  
                },  
                "ForAllValues:StringEquals": {  
                    "aws:TagKeys": ["stage"]  
                }  
            }  
        }  
    ]  
}
```

## Exemples de stratégies basées sur des stratégies gérées

Cette section montre comment contrôler l'accès utilisateur à AWS Elastic Beanstalk et inclut des exemples de stratégies qui fournissent l'accès requis pour les scénarios courants. Ces stratégies sont dérivées des stratégies gérées par Elastic Beanstalk. Pour plus d'informations sur l'attachement de stratégies gérées à des groupes et des utilisateurs, consultez [Gestion des stratégies utilisateur Elastic Beanstalk \(p. 946\)](#).

Dans ce scénario, Example Corp. est une société de logiciels avec trois équipes chargées du site web d'entreprise : les administrateurs qui gèrent l'infrastructure, les développeurs qui construisent le logiciel du site web et une équipe d'assurance qualité qui teste le site web. Pour aider à gérer les autorisations pour ses ressources Elastic Beanstalk, Example Corp. crée trois groupes auxquels appartiennent les membres de chaque équipe respective : administrateurs, développeurs et testeurs. Example Corp. souhaite que le groupe des administrateurs dispose d'un accès complet à toutes les applications, tous les environnements et leurs ressources sous-jacentes afin qu'ils puissent créer, dépanner et supprimer toutes les ressources Elastic Beanstalk. Les développeurs ont besoin d'autorisations pour afficher toutes les ressources Elastic Beanstalk ainsi que créer et déployer des versions d'application. Les développeurs ne doivent pas pouvoir créer de nouvelles applications ou de nouveaux environnements ou encore interrompre des environnements en cours d'exécution. Les testeurs doivent consulter toutes les ressources Elastic Beanstalk pour surveiller et tester les applications. Les testeurs ne devraient pas être en mesure d'apporter des modifications aux ressources Elastic Beanstalk.

Les exemples de stratégies suivants fournissent les autorisations requises pour chaque groupe.

### Exemple 1 : Groupe d'administrateurs – Toutes les API Elastic Beanstalk et les API des services connexes

La stratégie suivante fournit aux utilisateurs des autorisations pour toutes les actions requises pour utiliser Elastic Beanstalk. Cette stratégie permet également à Elastic Beanstalk de mettre en service et de gérer des ressources en votre nom dans les services suivants. Elastic Beanstalk s'appuie sur ces services supplémentaires pour fournir des ressources sous-jacentes lors de la création d'un environnement.

- Amazon Elastic Compute Cloud
  - Elastic Load Balancing
  - Auto Scaling
  - Amazon CloudWatch
  - Amazon Simple Storage Service
  - Amazon Simple Notification Service
  - Amazon Relational Database Service
  - AWS CloudFormation

Notez que cette stratégie est un exemple. Elle offre un large ensemble d'autorisations pour les services AWS qu'Elastic Beanstalk utilise pour gérer les applications et les environnements. Par exemple, `ec2:*` permet à un utilisateur AWS Identity and Access Management (IAM) d'effectuer n'importe quelle action sur n'importe quelle ressource Amazon EC2 du compte AWS. Ces autorisations ne sont pas limitées aux ressources que vous utilisez avec Elastic Beanstalk. La bonne pratique consiste à accorder aux personnes autorisées uniquement les autorisations dont elles ont besoin pour réaliser leur travail.

```
{  
    "Version" : "2012-10-17",  
    "Statement" : [  
        {  
            "Effect" : "Allow",  
            "Action" : [  
                "elasticbeanstalk:*",  
                "ec2:*",  
                "elasticloadbalancing:*",  
                "autoscaling:*",  
                "cloudwatch:",  
                "s3:*",  
                "sns:*",  
                "rds:*",  
                "cloudformation:*"  
            ],  
            "Resource" : "*"  
        }  
    ]  
}
```

**Exemple 2 : Groupe de développeurs – Toutes les opérations à l'exception des opérations nécessitant des privilèges élevés**

La stratégie suivante refuse l'autorisation de créer des applications et des environnements, mais autorise toutes les autres actions Elastic Beanstalk.

Notez que cette stratégie est un exemple. Elle offre un large ensemble d'autorisations aux produits AWS qu'Elastic Beanstalk utilise pour gérer des applications et des environnements. Par exemple, `ec2:*` permet à un utilisateur IAM d'effectuer n'importe quelle action sur n'importe quelle ressource Amazon EC2 du compte AWS. Ces autorisations ne sont pas limitées aux ressources que vous utilisez avec Elastic Beanstalk. La bonne pratique consiste à accorder aux personnes autorisées uniquement les autorisations dont elles ont besoin pour réaliser leur travail.

```
{  
    "Version" : "2012-10-17",  
    "Statement" : [  
        {  
            "Action" : [  
                "elasticbeanstalk:CreateApplication",
```

```
"elasticbeanstalk>CreateEnvironment",
"elasticbeanstalk>DeleteApplication",
"elasticbeanstalk>RebuildEnvironment",
"elasticbeanstalk>SwapEnvironmentCNAMES",
"elasticbeanstalk>TerminateEnvironment"],
"Effect" : "Deny",
"Resource" : "*"
},
{
"Action" : [
"elasticbeanstalk:*",
"ec2:*",
"elasticloadbalancing:*",
"autoscaling:*",
"cloudwatch:*",
"s3:*",
"sns:*",
"rds:*",
"cloudformation:*"],
"Effect" : "Allow",
"Resource" : "*"
}
]
```

## Exemple 3 : Testeurs – Affichage uniquement

La stratégie suivante autorise l'accès en lecture seule à toutes les applications, toutes les versions d'applications, tous les événements et tous les environnements. Elle ne permet d'effectuer aucune action.

```
{
"Version" : "2012-10-17",
"Statement" : [
{
"Effect" : "Allow",
"Action" : [
"elasticbeanstalk:Check*",
"elasticbeanstalk:Describe*",
"elasticbeanstalk>List*",
"elasticbeanstalk:RequestEnvironmentInfo",
"elasticbeanstalk:RetrieveEnvironmentInfo",
"ec2:Describe*",
"elasticloadbalancing:Describe*",
"autoscaling:Describe*",
"cloudwatch:Describe*",
"cloudwatch>List*",
"cloudwatch:Get*",
"s3:Get*",
"s3>List*",
"sns:Get*",
"sns>List*",
"rds:Describe*",
"cloudformation:Describe*",
"cloudformation:Get*",
"cloudformation>List*",
"cloudformation:Validate*",
"cloudformation:Estimate*"
],
"Resource" : "*"
}
]
```

## Exemples de stratégies basées sur des autorisations de ressource

Cette section décrit un cas d'utilisation pour contrôler les autorisations utilisateur pour les actions Elastic Beanstalk qui accèdent à des ressources Elastic Beanstalk spécifiques. Nous suivrons les exemples de stratégies qui prennent en charge le cas d'utilisation. Pour de plus amples informations sur les stratégies des ressources Elastic Beanstalk, veuillez consulter [Création d'une stratégie utilisateur personnalisée \(p. 947\)](#). Pour plus d'informations sur l'attachement de stratégies à des utilisateurs et des groupes, consultez [Gestion des stratégies IAM](#) dans Utilisation d'AWS Identity and Access Management.

Dans notre cas d'utilisation, Example Corp. est une petite agence de conseils développant des applications pour les deux clients différents. John est le gestionnaire de développement chargé du développement des deux applications Elastic Beanstalk : app1 et app2. John fait du développement et quelques tests sur les deux applications, et lui seul peut mettre à jour l'environnement de production pour les deux applications. Voici les autorisations dont il a besoin pour app1 et app2 :

- Afficher des applications, des versions d'applications, des environnements et des modèles de configuration
- Créer des versions d'applications et les déployer dans l'environnement intermédiaire
- Mettre à jour l'environnement de production
- Créer et résilier des environnements

Jill est testeur qui a besoin d'un accès pour afficher les ressources suivantes afin de surveiller et de tester les deux applications : applications, versions d'applications, environnements et modèles de configuration. Cependant, elle ne devrait pas être en mesure d'apporter des modifications aux ressources Elastic Beanstalk.

Jack est le développeur pour app1 qui a besoin d'un accès pour afficher toutes les ressources pour app1 ainsi que de créer des versions d'applications pour app1 et de les déployer dans l'environnement intermédiaire.

Judy est l'administratrice du compte AWS pour Example Corp. Elle a créé des utilisateurs IAM pour John, Jill et Jack, et elle attache les stratégies suivantes à ces utilisateurs pour accorder les autorisations appropriées pour les applications app1 et app2.

### Exemple 1 : John – gestionnaire de développement pour app1, app2

Nous avons divisé la stratégie de John en trois stratégies distinctes afin qu'elles soient plus faciles à lire et à gérer. Ensemble, elles donnent à John les autorisations dont il a besoin pour effectuer des actions de développement, de test et de déploiement sur les deux applications.

La première stratégie spécifie les actions pour Auto Scaling, Amazon S3, Amazon EC2, CloudWatch, Amazon SNS, Elastic Load Balancing, Amazon RDS et AWS CloudFormation. Elastic Beanstalk s'appuie sur ces services supplémentaires pour fournir des ressources sous-jacentes lors de la création d'un environnement.

Notez que cette stratégie est un exemple. Elle offre un large ensemble d'autorisations aux produits AWS qu'Elastic Beanstalk utilise pour gérer des applications et des environnements. Par exemple, `ec2 : *` permet à un utilisateur IAM d'effectuer n'importe quelle action sur n'importe quelle ressource Amazon EC2 du compte AWS. Ces autorisations ne sont pas limitées aux ressources que vous utilisez avec Elastic Beanstalk. La bonne pratique consiste à accorder aux personnes autorisées uniquement les autorisations dont elles ont besoin pour réaliser leur travail.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:*",
                "ecs:*",
                "ecr:*",
                "elasticloadbalancing:*",
                "autoscaling:*",
                "cloudwatch:",
                "s3:*",
                "sns:*",
                "cloudformation:*",
                "dynamodb:*",
                "rds:*",
                "sns:*",
                "logs:*",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:PassRole",
                "iam>ListRolePolicies",
                "iam>ListAttachedRolePolicies",
                "iam>ListInstanceProfiles",
                "iam>ListRoles",
                "iam>ListServerCertificates",
                "acm:DescribeCertificate",
                "acm>ListCertificates",
                "codebuild>CreateProject",
                "codebuild>DeleteProject",
                "codebuild:BatchGetBuilds",
                "codebuild:StartBuild"
            ],
            "Resource": "*"
        }
    ]
}
```

La deuxième stratégie spécifie les actions Elastic Beanstalk que John est autorisé à effectuer sur les ressources app1 et app2. La déclaration AllCallsInApplications autorise toutes les actions Elastic Beanstalk ("elasticbeanstalk:") effectuées sur toutes les ressources au sein d'app1 et d'app2 (par exemple, elasticbeanstalk>CreateEnvironment). La déclaration AllCallsOnApplications autorise toutes les actions Elastic Beanstalk ("elasticbeanstalk:") sur les ressources d'application app1 et app2 (par exemple, elasticbeanstalk>DescribeApplications, elasticbeanstalk>UpdateApplication, etc.). La déclaration AllCallsOnSolutionStacks autorise toutes les actions Elastic Beanstalk ("elasticbeanstalk:") pour les ressources d'une pile de solutions (par exemple, elasticbeanstalk>ListAvailableSolutionStacks).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllCallsInApplications",
            "Action": [
                "elasticbeanstalk:"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {

```

```

        "elasticbeanstalk:InApplication": [
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
            "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
        ]
    }
},
{
    "Sid": "AllCallsOnApplications",
    "Action": [
        "elasticbeanstalk:*"
    ],
    "Effect": "Allow",
    "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
    ]
},
{
    "Sid": "AllCallsOnSolutionStacks",
    "Action": [
        "elasticbeanstalk:*"
    ],
    "Effect": "Allow",
    "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
    ]
}
]
}

```

La troisième stratégie spécifie les actions Elastic Beanstalk pour lesquelles la deuxième stratégie a besoin d'autorisations afin de réaliser ces actions Elastic Beanstalk. L'instruction AllNonResourceCalls autorise l'action elasticbeanstalk:CheckDNSAvailability, qui est obligatoire pour appeler elasticbeanstalk>CreateEnvironment et d'autres actions. Elle autorise également l'action elasticbeanstalk>CreateStorageLocation, qui est obligatoire pour elasticbeanstalk>CreateApplication, elasticbeanstalk>CreateEnvironment et d'autres actions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllNonResourceCalls",
            "Action": [
                "elasticbeanstalk:CheckDNSAvailability",
                "elasticbeanstalk>CreateStorageLocation"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        }
    ]
}
```

## Exemple 2 : Jill – testeur pour app1, app2

Nous avons divisé la stratégie de Jill en trois stratégies distinctes afin qu'elles soient plus faciles à lire et à gérer. Ensemble, elles donnent à Jill les autorisations dont elle a besoin pour effectuer les actions de tests et de suivi sur les deux applications.

La première stratégie spécifie les actions `Describe*`, `List*` et `Get*` sur Auto Scaling, Amazon S3, Amazon EC2, CloudWatch, Amazon SNS, Elastic Load Balancing, Amazon RDS et AWS CloudFormation (pour les types de conteneur non hérités) afin que les actions Elastic Beanstalk soient en mesure de récupérer les informations pertinentes sur les ressources sous-jacentes des applications app1 et app2.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:Describe*",
                "elasticloadbalancing:Describe*",
                "autoscaling:Describe*",
                "cloudwatch:Describe*",
                "cloudwatch>List*",
                "cloudwatch:Get*",
                "s3:Get*",
                "s3>List*",
                "sns:Get*",
                "sns>List*",
                "rds:Describe*",
                "cloudformation:Describe*",
                "cloudformation:Get*",
                "cloudformation>List*",
                "cloudformation:Validate*",
                "cloudformation:Estimate*"
            ],
            "Resource": "*"
        }
    ]
}
```

La deuxième stratégie spécifie les actions Elastic Beanstalk que Jill est autorisée à effectuer sur les ressources app1 et app2. L'instruction `AllReadCallsInApplications` lui permet d'appeler les actions `Describe*` et les actions d'informations d'environnement. L'instruction `AllReadCallsOnApplications` lui permet d'appeler les actions `DescribeApplications` et `DescribeEvents` sur les ressources d'application app1 et app2. L'instruction `AllReadCallsOnSolutionStacks` permet l'affichage d'actions qui impliquent des ressources d'une pile de solutions (`ListAvailableSolutionStacks`, `DescribeConfigurationOptions` et `ValidateConfigurationSettings`).

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllReadCallsInApplications",
            "Action": [
                "elasticbeanstalk:Describe*",
                "elasticbeanstalk:RequestEnvironmentInfo",
                "elasticbeanstalk:RetrieveEnvironmentInfo"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "elasticbeanstalk:InApplication": [
                        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
                        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
                    ]
                }
            }
        }
}
```

```

},
{
    "Sid": "AllReadCallsOnApplications",
    "Action": [
        "elasticbeanstalk:DescribeApplications",
        "elasticbeanstalk:DescribeEvents"
    ],
    "Effect": "Allow",
    "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1",
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app2"
    ]
},
{
    "Sid": "AllReadCallsOnSolutionStacks",
    "Action": [
        "elasticbeanstalk>ListAvailableSolutionStacks",
        "elasticbeanstalk:DescribeConfigurationOptions",
        "elasticbeanstalk:ValidateConfigurationSettings"
    ],
    "Effect": "Allow",
    "Resource": [
        "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
    ]
}
]
}

```

La troisième stratégie spécifie les actions Elastic Beanstalk pour lesquelles la deuxième stratégie a besoin d'autorisations afin de réaliser ces actions Elastic Beanstalk. L'instruction `AllNonResourceCalls` permet l'action `elasticbeanstalk:CheckDNSAvailability`, qui est obligatoire pour certaines actions d'affichage.

```

{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllNonResourceCalls",
            "Action": [
                "elasticbeanstalk:CheckDNSAvailability"
            ],
            "Effect": "Allow",
            "Resource": [
                "*"
            ]
        }
    ]
}

```

## Exemple 3 : Jack – développeur pour app1

Nous avons divisé la stratégie de Jack en trois stratégies distinctes afin qu'elles soient plus faciles à lire et à gérer. Ensemble, elles donnent à Jack les autorisations dont il a besoin pour effectuer des tests, des contrôles et des actions de déploiement sur la ressource app1.

La première stratégie spécifie les actions sur Auto Scaling, Amazon S3, Amazon EC2, CloudWatch, Amazon SNS, Elastic Load Balancing, Amazon RDS et AWS CloudFormation (pour les types de conteneur non hérités) afin que les actions Elastic Beanstalk soient en mesure d'afficher et d'utiliser les ressources sous-jacentes de l'application app1. Pour afficher la liste des types de conteneurs non hérités pris en charge, consultez [the section called “Pourquoi certaines versions de plate-forme sont-elles marquées héritées ?” \(p. 507\)](#)

Notez que cette stratégie est un exemple. Elle offre un large ensemble d'autorisations aux produits AWS qu'Elastic Beanstalk utilise pour gérer des applications et des environnements. Par exemple, `ec2:*` permet à un utilisateur IAM d'effectuer n'importe quelle action sur n'importe quelle ressource Amazon EC2 du compte AWS. Ces autorisations ne sont pas limitées aux ressources que vous utilisez avec Elastic Beanstalk. La bonne pratique consiste à accorder aux personnes autorisées uniquement les autorisations dont elles ont besoin pour réaliser leur travail.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "ec2:*",  
                "elasticloadbalancing:*",  
                "autoscaling:*",  
                "cloudwatch:*",  
                "s3:*",  
                "sns:*",  
                "rds:*",  
                "cloudformation:*"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

La deuxième stratégie spécifie les actions Elastic Beanstalk que Jack est autorisé à effectuer sur la ressource app1.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "AllReadCallsAndAllVersionCallsInApplications",  
            "Action": [  
                "elasticbeanstalk:Describe*",  
                "elasticbeanstalk:RequestEnvironmentInfo",  
                "elasticbeanstalk:RetrieveEnvironmentInfo",  
                "elasticbeanstalk>CreateApplicationVersion",  
                "elasticbeanstalk>DeleteApplicationVersion",  
                "elasticbeanstalk:UpdateApplicationVersion"  
            ],  
            "Effect": "Allow",  
            "Resource": [  
                "*"  
            ],  
            "Condition": {  
                "StringEquals": {  
                    "elasticbeanstalk:InApplication": [  
                        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1"  
                    ]  
                }  
            }  
        },  
        {  
            "Sid": "AllReadCallsOnApplications",  
            "Action": [  
                "elasticbeanstalk:DescribeApplications",  
                "elasticbeanstalk:DescribeEvents"  
            ],  
            "Effect": "Allow",  
            "Resource": [  
                "*"  
            ]  
        }  
    ]  
}
```

```

        "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1"
    ],
},
{
    "Sid":"UpdateEnvironmentInApplications",
    "Action":[
        "elasticbeanstalk:UpdateEnvironment"
    ],
    "Effect":"Allow",
    "Resource":[
        "arn:aws:elasticbeanstalk:us-east-2:123456789012:environment/app1/app1-
staging*"
    ],
    "Condition":{
        "StringEquals":{
            "elasticbeanstalk:InApplication":[
                "arn:aws:elasticbeanstalk:us-east-2:123456789012:application/app1"
            ]
        },
        "StringLike":{
            "elasticbeanstalk:FromApplicationVersion":[
                "arn:aws:elasticbeanstalk:us-east-2:123456789012:applicationversion/app1/
*"
            ]
        }
    }
},
{
    "Sid":"AllReadCallsOnSolutionStacks",
    "Action":[
        "elasticbeanstalk>ListAvailableSolutionStacks",
        "elasticbeanstalk:DescribeConfigurationOptions",
        "elasticbeanstalk:ValidateConfigurationSettings"
    ],
    "Effect":"Allow",
    "Resource":[
        "arn:aws:elasticbeanstalk:us-east-2::solutionstack/*"
    ]
}
]
}

```

La troisième stratégie spécifie les actions Elastic Beanstalk pour lesquelles la deuxième stratégie a besoin d'autorisations afin de réaliser ces actions Elastic Beanstalk. L'instruction AllNonResourceCalls autorise l'action elasticbeanstalk:CheckDNSAvailability, qui est obligatoire pour appeler elasticbeanstalk>CreateEnvironment et d'autres actions. Elle autorise également l'action elasticbeanstalk>CreateStorageLocation, qui est obligatoire pour elasticbeanstalk>CreateEnvironment, et d'autres actions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid":"AllNonResourceCalls",
            "Action":[
                "elasticbeanstalk:CheckDNSAvailability",
                "elasticbeanstalk>CreateStorageLocation"
            ],
            "Effect":"Allow",
            "Resource":[
                "*"
            ]
        }
    ]
}
```

}

## Utilisation d'Elastic Beanstalk avec Amazon RDS

Vous pouvez utiliser Elastic Beanstalk avec Amazon Relational Database Service (Amazon RDS) pour configurer, exploiter et mettre à l'échelle une base de données relationnelle. Deux options sont disponibles pour commencer. Les voici.

- Créez une nouvelle base de données dans Amazon RDS.
- Commencez avec une base de données précédemment [créée par Elastic Beanstalk \(p. 617\)](#) et [découplée \(p. 622\)](#) par la suite d'un environnement Beanstalk. Pour de plus amples informations, veuillez consulter [the section called "Base de données" \(p. 617\)](#).

Vous pouvez utiliser l'une ou l'autre approche pour exécuter une instance de base de données dans Amazon RDS et configurer votre application pour qu'elle s'y connecte lors du lancement. Vous pouvez connecter plusieurs environnements à une base de données et effectuer également des mises à jour continues avec des déploiements bleu/vert.

### Note

Si vous n'avez encore jamais utilisé d'instance de base de données avec votre application, nous vous recommandons d'abord d'ajouter votre base de données à un environnement de test à l'aide de la console Elastic Beanstalk. Cette opération vous permet de vérifier que votre application peut lire les propriétés de l'environnement, créer une chaîne de connexion et se connecter à une instance de base de données sans le travail de configuration supplémentaire requis pour une base de données autonome. Pour de plus amples informations, veuillez consulter [Ajout d'une base de données à votre environnement Elastic Beanstalk \(p. 617\)](#).

Pour autoriser les instances Amazon EC2 de votre environnement à se connecter à une base de données extérieure, configurez un groupe de sécurité supplémentaire pour le groupe Auto Scaling associé à votre environnement. Vous pouvez attacher le même groupe de sécurité attaché à votre instance de base de données. Vous pouvez également utiliser un groupe de sécurité distinct. Si vous attachez un groupe de sécurité différent, vous devez configurer le groupe de sécurité attaché à votre base de données pour autoriser l'accès entrant à partir de ce groupe de sécurité.

### Note

Vous pouvez connecter votre environnement à une base de données en ajoutant une règle au groupe de sécurité attaché à votre base de données. Cette règle doit autoriser l'accès entrant à partir du groupe de sécurité généré automatiquement qu'Elastic Beanstalk attache au groupe Auto Scaling pour votre environnement. Toutefois, sachez qu'en créant cette règle, vous créez également une dépendance entre les deux groupes de sécurité. Par conséquent, lorsque vous essayez de résilier l'environnement, Elastic Beanstalk ne peut pas supprimer le groupe de sécurité de l'environnement, car le groupe de sécurité de la base de données en dépend.

Une fois que vous avez lancé votre instance de base de données et configuré les groupes de sécurité, vous pouvez transmettre les informations de connexion, telles que le point de terminaison et le mot de passe, à votre application via les propriétés d'environnement. Il s'agit du même mécanisme que celui utilisé en arrière-plan par Elastic Beanstalk lorsque vous exécutez une instance de base de données dans votre environnement.

Pour bénéficier d'une couche de sécurité supplémentaire, vous pouvez stocker vos informations de connexion dans Amazon S3 et configurer Elastic Beanstalk pour les récupérer au cours du déploiement. Avec les [fichiers de configuration \(.ebextensions\) \(p. 737\)](#), vous pouvez configurer les instances de votre environnement afin de récupérer en toute sécurité des fichiers depuis Amazon S3 lorsque vous déployez votre application.

## Rubriques

- [Lancement et connexion d'une instance Amazon RDS externe dans un VPC par défaut \(p. 992\)](#)
- [Lancement et connexion d'une instance Amazon RDS externe dans EC2-Classic \(p. 997\)](#)
- [Stockage de la chaîne de connexion dans Amazon S3 \(p. 1001\)](#)
- [Nettoyage d'une instance Amazon RDS externe \(p. 1003\)](#)

# Lancement et connexion d'une instance Amazon RDS externe dans un VPC par défaut

Pour utiliser une base de données externe avec une application en cours d'exécution dans Elastic Beanstalk, deux options sont mises à votre disposition. La première consiste à lancer une instance de base de données avec Amazon RDS. Toute instance que vous lancez avec Amazon RDS est totalement indépendante d'Elastic Beanstalk et de vos environnements Elastic Beanstalk. Cela signifie que vous pouvez utiliser n'importe quel type d'instance et n'importe quel moteur de base de données pris en charge par Amazon RDS, même ceux qui ne sont pas utilisés par Elastic Beanstalk.

Outre le lancement d'une nouvelle instance de base de données, vous pouvez également commencer avec une base de données précédemment [créeée par Elastic Beanstalk \(p. 617\)](#) et [découplée \(p. 622\)](#) par la suite d'un environnement Beanstalk. Pour de plus amples informations, veuillez consulter [the section called "Base de données" \(p. 617\)](#). Si vous choisissez cette option, vous ne devez pas terminer la procédure de lancement d'une nouvelle base de données. Toutefois, vous devez effectuer les procédures suivantes décrites au sein de cette rubrique.

Les procédures suivantes décrivent le processus pour un [VPC par défaut](#). Le processus est le même si vous utilisez un VPC personnalisé, à l'exception des obligations supplémentaires suivantes : votre environnement et l'instance DB doivent se situer dans le même sous-réseau ou dans des sous-réseaux qui sont autorisés à communiquer entre eux. Pour de plus amples informations sur la configuration d'un VPC personnalisé à utiliser avec Elastic Beanstalk, consultez [Utilisation d'Elastic Beanstalk avec Amazon VPC \(p. 1006\)](#).

### Note

- Si vous commencez avec une base de données créée par Elastic Beanstalk et découpée ensuite d'un environnement Beanstalk, vous pouvez ignorer les premières étapes et poursuivre avec les étapes regroupées sous Pour modifier les règles de trafic entrant sur le groupe de sécurité de votre instance RDS.
- Si vous envisagez d'utiliser la base de données que vous découpez pour un environnement de production, vérifiez que le type de stockage utilisé par la base de données est adapté à votre charge de travail. Pour de plus amples informations, consultez [Stockage d'instance de base de données](#) et [Modification d'une instance de base de données](#) dans le Guide de l'utilisateur Amazon RDS.

Pour lancer une instance DB RDS dans un VPC par défaut

1. Ouvrez la [console RDS](#).
2. Dans la panneau de navigation, choisissez Databases (Bases de données).
3. Choisissez Create database (Créer une base de données).
4. Choisissez Création standard.

### Important

Ne choisissez pas Création facile. Si vous choisissez cette option, vous ne pouvez pas configurer les paramètres nécessaires pour lancer cette base de données RDS.

5. Sous Configuration supplémentaire, pour Nom initial de la base de données, tapez **ebdb**.

6. Vérifiez les paramètres par défaut et ajustez ces paramètres en fonction de vos exigences spécifiques. Prêtez attention aux options suivantes :
  - DB instance class (Classe d'instance de base de données) : choisissez une taille d'instance avec un niveau approprié de puissance d'UC et de mémoire pour votre charge de travail.
  - Multi-AZ deployment (Déploiement multi-AZ) : pour une haute disponibilité, définissez cette option sur Create an Aurora Replica/Reader node in a different AZ (Créer un nœud de réplica/lecteur Aurora dans une autre zone de disponibilité).
  - Master username (Identifiant principal) et Master password (Mot de passe principal) : nom d'utilisateur et mot de passe de la base de données. Notez les valeurs de ces paramètres, car vous en aurez besoin par la suite.
7. Vérifiez les paramètres par défaut pour les autres options, puis cliquez sur Créer une base de données.

Ensuite, modifiez le groupe de sécurité attaché à votre instance de base de données pour autoriser le trafic entrant sur le port approprié. Il s'agit du même groupe de sécurité que celui que vous attacherez plus tard à votre environnement Elastic Beanstalk. Par conséquent, la règle que vous ajoutez accordera une autorisation d'accès entrant aux autres ressources du même groupe de sécurité.

Pour modifier les règles de trafic entrant sur le groupe de sécurité associé à votre instance RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.
4. Dans la section Connectivity (Connectivité), prenez note des Subnets (Sous-réseaux), des Security groups (Groupes de sécurité) et du Endpoint (Point de terminaison) affichés sur cette page. Ainsi, vous pourrez utiliser ces informations ultérieurement.
5. Sous Security (Sécurité), vous voyez le groupe de sécurité associé à l'instance de base de données. Ouvrez le lien pour afficher le groupe de sécurité dans la console Amazon EC2.

Connectivity	Monitoring	Logs & events	Configuration	Maintenance & backups	Tags
<b>Connectivity</b>					
<b>Endpoint &amp; port</b> <p>Endpoint ebdb... .us-east-1.rds.amazonaws.com</p> <p>Port 3306</p>	<b>Networking</b> <p>Availability zone us-east-1b</p> <p>VPC <a href="#">vpc-5732152e</a></p> <p>Subnet group default</p> <p>Subnets</p> <ul style="list-style-type: none"> <li><a href="#">subnet-778f5359</a></li> <li><a href="#">subnet-7cc75e73</a></li> <li><a href="#">subnet-68432522</a></li> <li><a href="#">subnet-8762b1db</a></li> <li><a href="#">subnet-a7578ac0</a></li> <li><a href="#">subnet-a0e4069e</a></li> </ul>	<b>Security</b> <p>VPC security groups <a href="#">rds-launch-wizard-4 (active)</a></p> <p>Public accessibility Yes</p> <p>Certificate authority <a href="#">rds-ca-2015</a></p> <p>Certificate authority date Mar 5th, 2020</p>			

6. Dans les détails du groupe de sécurité, choisissez l'onglet Inbound (Entrant).
7. Choisissez Modifier.
8. Choisissez Add Rule.
9. Pour Type, choisissez le moteur de base de données utilisé par votre application.
10. Pour Source, entrez **sg-** pour afficher la liste des groupes de sécurité disponibles. Choisissez le groupe de sécurité associé au groupe Auto Scaling utilisé avec votre environnement Elastic Beanstalk. Cela permet aux instances Amazon EC2 de l'environnement d'accéder à la base de données.

The screenshot shows the 'Edit inbound rules' page. At the top, there are four filter tabs: Type (MySQL/Aurora), Protocol (TCP), Port Range (3306), and Source (Custom). Below these, two rows of inbound rules are listed. The first row has a 'Type' of MySQL/Aurora, 'Protocol' of TCP, and 'Port Range' of 3306. The second row also has a 'Type' of MySQL/Aurora, 'Protocol' of TCP, and 'Port Range' of 3306. A large yellow box highlights the 'Source' input field, which contains 'sg-'. A dropdown menu is open over this field, showing a single item: 'sg-07ecc7ed5b2c0c099 - rds-launch-wizard-4'. A note at the bottom of the page states: 'NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This will cause traffic to be dropped for a very brief period of time until the new rule can be created.'

11. Choisissez Enregistrer.

Ensuite, ajoutez le groupe de sécurité de l'instance de base de données à votre environnement en cours d'exécution. Cette procédure conduit Elastic Beanstalk à reprovisionner toutes les instances de votre environnement avec le groupe de sécurité supplémentaire associé.

Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    - a. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
    - b. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

#### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

- c. Dans le panneau de navigation, choisissez Configuration.
- d. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).
- e. Sous EC2 security groups (Groupes de sécurité EC2), choisissez les groupes de sécurité à attacher aux instances, en plus du groupe de sécurité de l'instance créé par Elastic Beanstalk.
- f. Choisissez Apply.
- g. Lisez l'avertissement, puis choisissez Confirmer.
- Pour ajouter un groupe de sécurité à l'aide d'un [fichier de configuration \(p. 737\)](#), utilisez l'exemple de fichier [securitygroup-addeexisting.config](#).

Ensuite, transmettez les informations de connexion à votre environnement à l'aide des propriétés de l'environnement. Lorsque vous [ajoutez une instance de base de données à votre environnement \(p. 617\)](#) avec la console Elastic Beanstalk, Elastic Beanstalk utilise des propriétés d'environnement comme RDS\_HOSTNAME pour transmettre les informations de connexion à votre application. Vous pouvez utiliser les mêmes propriétés. De cette façon, vous utilisez le même code d'application avec les instances de base de données intégrées et les instances de base de données externes. Vous pouvez également choisir vos propres noms de propriété.

Pour configurer les propriétés d'environnement pour une instance de base de données Amazon RDS

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Dans la section Environment properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance DB RDS intégrée, utilisez les noms et valeurs suivants : Vous pouvez trouver toutes les valeurs, à l'exception de votre mot de passe, dans la [console RDS](#).

Nom de la propriété	Description	Valeur de la propriété
RDS_HOSTNAME	Nom d'hôte de l'instance DB.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Endpoint (Point de terminaison).
RDS_PORT	Port sur lequel l'instance de base de données accepte des connexions. La valeur par défaut varie selon les moteurs de base de données.	Sous l'onglet Connectivity & security (Connectivité et sécurité) de la console Amazon RDS : Port.
RDS_DB_NAME	Nom de la base de données, <b>ebdb</b> .	Sous l'onglet Configuration de la console Amazon RDS : DB Name (Nom de base de données).
RDS_USERNAME	Nom d'utilisateur que vous avez configuré pour votre base de données.	Sous l'onglet Configuration de la console Amazon RDS : Master username (Identifiant principal).
RDS_PASSWORD	Mot de passe que vous avez configuré pour votre base de données.	Non disponible pour référence dans la console Amazon RDS.

## Environment Properties

The following properties are passed into the application as environment variables. [Learn more.](#)

Property Name	Property Value
RDS_DB_NAME	ebdb
RDS_HOSTNAME	webapp-db.jxccb5mpan
RDS_PORT	5432
RDS_USERNAME	webapp-admin
RDS_PASSWORD	kUj5uKxmWDMYc403

[Cancel](#)

6. Choisissez Apply.

Si vous n'avez pas programmé votre application pour lire les propriétés d'environnement et créer une chaîne de connexion, consultez l'une des rubriques suivantes afin d'obtenir des instructions pour le langage de votre choix :

- Java SE – [Connexion à une base de données \(plateformes Java SE\) \(p. 138\)](#)
- Java avec Tomcat – [Connexion à une base de données \(plateformes Tomcat\) \(p. 138\)](#)
- Node.js – [Connexion à une base de données \(p. 289\)](#)
- .NET – [Connexion à une base de données \(p. 223\)](#)
- PHP – [Connexion à une base de données avec un PDO ou MySQLi \(p. 355\)](#)
- Python – [Connexion à une base de données \(p. 383\)](#)
- Ruby – [Connexion à une base de données \(p. 402\)](#)

Enfin, selon le moment auquel votre application lit les variables d'environnement, vous pouvez être amené à redémarrer le serveur d'application sur les instances de votre environnement.

Pour redémarrer les serveurs d'applications de votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.

2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Restart app server(s) (Redémarrer le(s) serveur(s) d'applications).

## Lancement et connexion d'une instance Amazon RDS externe dans EC2-Classic

Important

Amazon EC2-Classic atteindra la fin de sa prise en charge standard le 15 août 2022. Pour éviter les interruptions de vos charges de travail, nous vous recommandons de procéder à la migration d'Amazon EC2-Classic vers un VPC avant cette date. Nous vous demandons également de ne pas lancer de ressources AWS sur Amazon EC2-Classic à l'avenir et de plutôt utiliser Amazon VPC. Pour de plus amples informations, consultez [Migration d'EC2-Classic vers un VPC \(p. 653\)](#) et le billet de blog [EC2-Classic Networking is Retiring - Here's How to Prepare](#).

Si vous utilisez EC2-Classic (et non un VPC) avec AWS Elastic Beanstalk, la procédure change légèrement en raison des différences de fonctionnement des groupes de sécurité. Dans EC2-Classic, les instances de base de données ne peuvent pas utiliser de groupes de sécurité EC2. Elles disposent donc d'un groupe de sécurité de base de données qui fonctionne uniquement avec Amazon RDS.

Vous pouvez ajouter à un groupe de sécurité de base de données des règles qui autorisent l'accès entrant à partir de groupes de sécurité EC2. Toutefois, vous ne pouvez pas attacher un groupe de sécurité de base de données au groupe Auto Scaling associé à votre environnement. Pour éviter de créer une dépendance entre le groupe de sécurité de base de données et votre environnement, vous devez créer un troisième groupe de sécurité dans Amazon EC2. Vous devez ensuite ajouter une règle dans le groupe de sécurité de base de données pour accorder l'accès entrant au nouveau groupe de sécurité. Enfin, vous devez l'affecter au groupe Auto Scaling au sein de votre environnement Elastic Beanstalk.

Note

- Si vous commencez avec une base de données créée par Elastic Beanstalk et découpée ensuite d'un environnement Beanstalk, vous pouvez ignorer les premières étapes et poursuivre avec les étapes regroupées sous Pour créer un groupe de sécurité de liaison.
- Si vous envisagez d'utiliser la base de données que vous découpez pour un environnement de production, vérifiez que le type de stockage utilisé par la base de données est adapté à votre charge de travail. Pour de plus amples informations, consultez [Stockage d'instance de base de données](#) et [Modification d'une instance de base de données](#) dans le Guide de l'utilisateur Amazon RDS.

### Pour lancer une instance RDS dans EC2 classic (sans VPC)

1. Ouvrez la [console de gestion RDS](#).
2. Choisissez Create database (Créer une base de données).
3. Continuez dans l'assistant. Notez les valeurs que vous saisissez pour les options suivantes :
  - Master Username
  - Master Password
4. Lorsque vous atteignez Configurer les paramètres avancés, pour les paramètres Réseau et sécurité, choisissez les options suivantes :

- VPC – **Not in VPC**. Si cette option n'est pas disponible, cela signifie que votre compte ne prend peut-être pas en charge [EC2-Classic](#) ou que vous avez choisi un [type d'instance uniquement disponible dans le VPC](#).
  - Zone de disponibilité – **No Preference**
  - DB Security Group(s) (Security Group DB) – **Create new Security Group**
5. Configurez les autres options, puis sélectionnez Crée une base de données. Notez les valeurs que vous saisissez pour les options suivantes :
- Database Name
  - Database Port

Dans EC2-Classic, votre instance de base de données dispose d'un groupe de sécurité de base de données au lieu d'un groupe de sécurité VPC. Vous ne pouvez pas attacher de groupe de sécurité de base de données à votre environnement Elastic Beanstalk. En lieu et place, vous devez créer un autre groupe de sécurité que vous pouvez autoriser à accéder à l'instance de base de données et attacher à votre environnement. Nous le désignons par le terme groupe de sécurité de liaison et l'appelons **webapp-bridge**.

Pour créer un groupe de sécurité de liaison

1. Ouvrez la [console Amazon EC2](#).
2. Dans la barre de navigation, choisissez Security Groups (Groupes de sécurité) sous Network & Security (Réseau et sécurité).
3. Sélectionnez Crée un groupe de sécurité.
4. Pour Security group name (Nom du groupe de sécurité), saisissez **webapp-bridge**.
5. Pour Description, saisissez **Provide access to DB instance from Elastic Beanstalk environment instances..**
6. Pour VPC, laissez la valeur sélectionnée par défaut.
7. Sélectionnez Create (Créer).

Ensuite, modifiez le groupe de sécurité associé à votre instance DB pour autoriser le trafic entrant provenant du groupe de sécurité de liaison.

Pour modifier les règles d'entrée sur le groupe de sécurité pour votre instance RDS

1. Ouvrez la [console Amazon RDS](#).
2. Choisissez Bases de données.
3. Choisissez le nom de votre instance de base de données pour en afficher les détails.
4. Dans la section Connectivity (Connectivité), sous Security (Sécurité), le groupe de sécurité associé à l'instance de base de données s'affiche. Ouvrez le lien pour afficher le groupe de sécurité dans la console Amazon EC2.
5. Dans les détails du groupe de sécurité, définissez l'option Connection Type (Type de connexion) sur EC2 Security Group (Groupe de sécurité EC2).
6. Dans le champ EC2 Security Group Name (Nom du groupe de sécurité EC2), indiquez le nom du groupe de sécurité de liaison que vous avez créé.
7. Choisissez Authorize.

Ensuite, ajoutez le groupe de sécurité de liaison à votre environnement en cours d'exécution. Cette procédure nécessite que le groupe de sécurité supplémentaire associé soit de nouveau fourni à toutes les instances de votre environnement.

## Pour ajouter un groupe de sécurité à votre environnement

- Effectuez l'une des actions suivantes :
  - Pour ajouter un groupe de sécurité à l'aide de la console Elastic Beanstalk
    - a. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
    - b. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

- c. Dans le panneau de navigation, choisissez Configuration.
- d. Dans la catégorie de configuration Instances, choisissez Edit (Modifier).
- e. Sous EC2 security groups (Groupes de sécurité EC2), choisissez les groupes de sécurité à attacher aux instances, en plus du groupe de sécurité de l'instance créé par Elastic Beanstalk.
- f. Choisissez Apply.
- g. Lisez l'avertissement, puis choisissez Confirmer.
- Pour ajouter un groupe de sécurité à l'aide d'un [fichier de configuration \(p. 737\)](#), utilisez l'exemple de fichier [securitygroup-addexisting.config](#).

Ensuite, transmettez les informations de connexion à votre environnement à l'aide des propriétés de l'environnement. Lorsque vous [ajoutez une instance de base de données à votre environnement \(p. 617\)](#) avec la console Elastic Beanstalk, Elastic Beanstalk utilise des propriétés d'environnement comme RDS\_HOSTNAME pour transmettre les informations de connexion à votre application. Vous pouvez utiliser les mêmes propriétés afin d'utiliser le même code d'application avec les instances de base de données intégrées et les instances de base de données externes. Vous pouvez également choisir vos propres noms de propriété.

## Pour configurer les propriétés de l'environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

### Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Dans le panneau de navigation, choisissez Configuration.
4. Dans la catégorie de configuration Software (Logiciels), choisissez Edit (Modifier).
5. Dans la section Environment Properties (Propriétés de l'environnement), définissez les variables lues par votre application pour créer une chaîne de connexion. Pour assurer la compatibilité avec les environnements disposant d'une instance RDS intégrée, utilisez les paramètres suivants :
  - RDS\_DB\_NAME : valeur du champ DB Name (Nom de base de données) présente dans la console Amazon RDS.
  - RDS\_USERNAME : valeur du champ Master Username (Identifiant principal) que vous entrez lorsque vous ajoutez la base de données à votre environnement.
  - RDS\_PASSWORD : valeur du champ Master Password (Mot de passe principal) que vous entrez lorsque vous ajoutez la base de données à votre environnement.

- RDS\_HOSTNAME : valeur du champ Endpoint (Point de terminaison) de l'instance de base de données présente dans la console Amazon RDS.
- RDS\_PORT : valeur du champ Port présente dans la console Amazon RDS.

Environment Properties	
Property Name	Property Value
RDS_DB_NAME	ebdb
RDS_HOSTNAME	webapp-db.jxzcb5mpan
RDS_PORT	5432
RDS_USERNAME	webapp-admin
RDS_PASSWORD	kUj5uKxmWDMYc403

[Cancel](#)

6. Choisissez Apply (Appliquer)

Si vous n'avez pas encore programmé votre application pour lire les propriétés d'environnement et créer une chaîne de connexion, consultez l'une des rubriques suivantes afin d'obtenir des instructions pour le langage de votre choix :

- Java SE – [Connexion à une base de données \(plateformes Java SE\) \(p. 138\)](#)
- Java avec Tomcat – [Connexion à une base de données \(plateformes Tomcat\) \(p. 138\)](#)
- Node.js – [Connexion à une base de données \(p. 289\)](#)
- .NET – [Connexion à une base de données \(p. 223\)](#)
- PHP – [Connexion à une base de données avec un PDO ou MySQLi \(p. 355\)](#)
- Python – [Connexion à une base de données \(p. 383\)](#)
- Ruby – [Connexion à une base de données \(p. 402\)](#)

Enfin, selon le moment auquel votre application lit les variables d'environnement, vous pouvez être amené à redémarrer le serveur d'application sur les instances de votre environnement.

Pour redémarrer les serveurs d'applications de votre environnement

1. Ouvrez la [console Elastic Beanstalk](#) et, dans la liste Regions (Régions), sélectionnez votre région AWS.
2. Dans le panneau de navigation, choisissez Environments (Environnements), puis choisissez le nom de votre environnement dans la liste.

Note

Si vous avez plusieurs environnements, utilisez la barre de recherche pour filtrer la liste des environnements.

3. Choisissez Environment actions (Actions d'environnement), puis Restart app server(s) (Redémarrer le(s) serveur(s) d'applications).

## Stockage de la chaîne de connexion dans Amazon S3

Une option de substitution, bien que non optimale, consiste à fournir les informations de connexion à votre application avec des propriétés d'environnement. Cela permet de conserver vos mots de passe en dehors de votre code. Cependant, les propriétés d'environnement sont visibles dans la [console de gestion de l'environnement \(p. 429\)](#) et peuvent être consultées par n'importe quel utilisateur autorisé à [décrire les paramètres de configuration](#) sur votre environnement. Selon la plateforme, les propriétés d'environnement peuvent également s'afficher dans les [journals d'instance \(p. 884\)](#).

Pour éviter ce problème, nous vous recommandons de verrouiller vos informations de connexion en les stockant dans un compartiment Amazon S3. La procédure principale est la suivante :

- Chargez un fichier contenant votre chaîne de connexion dans un compartiment Amazon S3.
- Accordez au profil d'instance EC2 l'autorisation de lire le fichier.
- Configurez votre application pour télécharger le fichier au cours du déploiement.
- Lisez le fichier dans votre code d'application.

Commencez par créer un compartiment afin de stocker le fichier contenant votre chaîne de connexion. Pour cet exemple, nous utiliserons un fichier JSON qui dispose d'une seule paire clé-valeur. La valeur correspond à une chaîne de connexion JDBC pour une instance de base de données PostgreSQL dans Amazon RDS :

`beanstalk-database.json`

```
{  
    "connection": "jdbc:postgresql://mydb.b5uacpxznijm.us-west-2.rds.amazonaws.com:5432/ebdb?  
user=username&password=mypassword"  
}
```

Les parties de l'URL en surbrillance correspondent au point de terminaison, au port, au nom de base de données, au nom d'utilisateur et au mot de passe de la base de données.

Pour créer un compartiment et charger un fichier

1. Ouvrez la [console Amazon S3](#).
2. Choisissez Créer un compartiment.
3. Indiquez une valeur pour les champs Bucket Name (Nom du compartiment) et Region (Région).
4. Sélectionnez Créer.
5. Ouvrez le compartiment, puis choisissez Charger.

6. Suivez les instructions pour charger le fichier.

Par défaut, votre compte est propriétaire du fichier et est autorisé à le gérer. Toutefois, les utilisateurs et les rôles IAM ne disposent de cette autorisation que si vous leur en accordez explicitement l'accès. Accordez une autorisation aux instances de votre environnement Elastic Beanstalk en ajoutant une stratégie au [profil d'instance \(p. 22\)](#).

Le profil d'instance par défaut se nomme `aws-elasticbeanstalk-ec2-role`. Si vous n'êtes pas certain du nom de votre profil d'instance, vous le trouverez sur la page Configuration de la [console de gestion de l'environnement \(p. 538\)](#).

Pour ajouter des autorisations au profil d'instance

1. Ouvrez la [console IAM](#).
2. Sélectionnez Roles.
3. Choisissez `aws-elasticbeanstalk-ec2-role`.
4. Choisissez Add inline policy.
5. Ajoutez une stratégie permettant à l'instance de récupérer le fichier.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Sid": "database",  
            "Action": [  
                "s3:GetObject"  
            ],  
            "Effect": "Allow",  
            "Resource": [  
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET-123456789012/beanstalk-database.json"  
            ]  
        }  
    ]  
}
```

Remplacez les noms de compartiment et d'objet par les noms de votre compartiment et de votre objet.

Ensuite, ajoutez un [fichier de configuration \(p. 737\)](#) à votre code source, demandant à Elastic Beanstalk de télécharger le fichier à partir d'Amazon S3 au cours du déploiement.

`~/my-app/.ebextensions/database.config`

```
Resources:  
  AWSEBAutoScalingGroup:  
    Metadata:  
      AWS::CloudFormation::Authentication:  
        S3Auth:  
          type: "s3"  
          buckets: ["DOC-EXAMPLE-BUCKET-123456789012"]  
          roleName: "aws-elasticbeanstalk-ec2-role"  
  
  files:  
    "/tmp/beanstalk-database.json" :  
      mode: "000644"  
      owner: root  
      group: root  
      authentication: "S3Auth"
```

source: <https://s3-us-west-2.amazonaws.com/DOC-EXAMPLE-BUCKET-123456789012/beanstalk-database.json>

Ce fichier de configuration effectue deux actions. La clé `Resources` ajoute une méthode d'authentification aux métadonnées du groupe Auto Scaling pour l'environnement. Elastic Beanstalk peut utiliser cette méthode d'authentification pour accéder à Amazon S3. La clé `files` permet à Elastic Beanstalk de télécharger le fichier à partir d'Amazon S3 et de le stocker localement dans `/tmp/` pendant le déploiement.

Déployez votre application avec le fichier de configuration dans le dossier `.ebextensions` à la racine de votre code source. Si vous avez correctement configuré les autorisations, le déploiement est effectué et le fichier est téléchargé dans toutes les instances de votre environnement. Dans le cas contraire, le déploiement échoue.

Enfin, ajoutez du code à votre application afin de lire le fichier JSON et d'utiliser la chaîne de connexion pour vous connecter à la base de données.

## Nettoyage d'une instance Amazon RDS externe

Lorsque vous connectez une instance Amazon RDS externe à votre environnement Elastic Beanstalk, l'instance de base de données ne dépend pas du cycle de vie de votre environnement et n'est par conséquent pas supprimée lorsque vous résiliez votre environnement. Afin de vous assurer que les informations personnelles que vous pouvez avoir stockées dans l'instance de base de données ne sont pas inutilement conservées, supprimez les enregistrements dont vous n'avez plus besoin. Vous pouvez également supprimer l'instance de base de données.

## Utilisation d'Elastic Beanstalk avec Amazon S3

Amazon Simple Storage Service (Amazon S3) assure un stockage de données hautement durable et tolérant aux pannes.

Elastic Beanstalk crée un compartiment Amazon S3 nommé `elasticbeanstalk-region-account-id` pour chaque région dans laquelle vous créez des environnements. Elastic Beanstalk utilise ce compartiment pour stocker les objets (des fichiers de configuration temporaires, par exemple) requis pour le bon fonctionnement de votre application.

Elastic Beanstalk n'active pas le chiffrement par défaut pour le compartiment Amazon S3 qu'il crée. Cela signifie que, par défaut, les objets sont stockés non chiffrés dans le compartiment (et sont accessibles uniquement pour les utilisateurs autorisés). Certaines applications exigent que tous les objets soient chiffrés lorsqu'ils sont stockés sur un disque dur, dans une base de données, etc. (c'est également ce que l'on nomme chiffrement au repos). Si vous avez cette exigence, vous pouvez configurer les compartiments de votre compte pour que le chiffrement soit activé par défaut. Pour de plus amples informations, consultez [Chiffrement par défaut Amazon S3 pour les compartiments S3](#) dans le Guide de l'utilisateur Amazon Simple Storage Service.

## Contenu du compartiment Elastic Beanstalk Amazon S3

Le tableau ci-dessous répertorie certains objets stockés par Elastic Beanstalk dans votre compartiment Amazon S3 `elasticbeanstalk-*`. Le tableau indique également les objets qui doivent être supprimés manuellement. Pour éviter des frais de stockage inutiles et pour vous assurer qu'aucune information personnelle n'est conservée, veillez à supprimer manuellement ces objets lorsque vous n'en avez plus besoin.

Objet	Stocké à quel moment?	Supprimé à quel moment?
Versions de l'application (p. 409)	Lorsque vous créez un environnement ou déployez votre code d'application dans un environnement existant, Elastic Beanstalk stocke une version de l'application dans Amazon S3 et l'associe à l'environnement.	Pendant la suppression de l'application, et conformément au Cycle de vie des versions (p. 411).
Bundles source (p. 409)	Lorsque vous chargez une nouvelle version de l'application à l'aide de la console Elastic Beanstalk ou de l'interface de ligne de commande (CLI) EB, Elastic Beanstalk en stocke une copie dans Amazon S3, et la définit comme le bundle de fichiers source de votre environnement.	Manuellement. Lorsque vous supprimez une version d'application, vous pouvez choisir Delete versions from Amazon S3 (Supprimer les versions d'Amazon S3) pour supprimer également le bundle source connexe. Pour plus d'informations, consultez Gestion des versions d'application (p. 409).
Plateformes personnalisées	Lorsque vous créez une plateforme personnalisée, Elastic Beanstalk stocke temporairement les données associées dans Amazon S3.	Une fois que vous avez terminé la création de la plateforme personnalisée.
Les fichiers journaux (p. 884)	Vous pouvez demander à Elastic Beanstalk de récupérer les fichiers journaux d'instance (journaux de processus ou de groupe) et de les stocker dans Amazon S3. Vous pouvez également activer la rotation des journaux et configurer votre environnement de sorte à publier automatiquement les journaux dans Amazon S3 après leur rotation.	Journaux de processus et de groupe : 15 minutes après leur création.  Journaux ayant subi une rotation : Manuellement.
Configurations enregistrées (p. 779)	Manuellement.	Manuellement.

## Suppression d'objets dans le compartiment Elastic Beanstalk Amazon S3

Lorsque vous arrêtez un environnement ou supprimez une application, Elastic Beanstalk supprime la plupart des objets connexes d'Amazon S3. Pour réduire les coûts de stockage d'une application en cours d'exécution, supprimez systématiquement les objets dont votre application n'a pas besoin. En outre, prenez une attention particulière aux objets que vous devez supprimer manuellement, comme indiqué dans Contenu du compartiment Elastic Beanstalk Amazon S3 (p. 1003). Pour être sûr que des informations privées ne sont pas inutilement conservées, supprimez ces objets lorsque vous n'en avez plus besoin.

- Supprimez les versions d'application que vous ne pensez plus utiliser dans votre application. Lorsque vous supprimez une version de l'application, vous pouvez sélectionner Delete versions from Amazon S3 (Supprimer les versions d'Amazon S3) pour supprimer également le bundle de fichiers source connexe. Il s'agit d'une copie du code source et des fichiers de configuration de votre application, chargée par Elastic Beanstalk sur Amazon S3 lorsque vous avez déployé une application ou chargé une version de l'application. Pour savoir comment supprimer une version d'application, consultez Gestion des versions d'application (p. 409).
- Supprimez les journaux ayant fait l'objet d'une rotation, dont vous n'avez pas besoin. Vous pouvez aussi les télécharger ou les déplacer vers Amazon S3 Glacier pour procéder à des analyses complémentaires.

- Supprimez les configurations enregistrées que vous n'allez plus utiliser dans vos environnements.

## Suppression du compartiment Elastic Beanstalk Amazon S3

Elastic Beanstalk applique une stratégie de compartiment aux compartiments qu'il crée pour permettre aux environnements d'écrire dans le compartiment et d'empêcher toute suppression accidentelle. Si vous avez besoin de supprimer un compartiment créé par Elastic Beanstalk, commencez par supprimer la stratégie de compartiment de la section Permissions (Autorisations) des propriétés de compartiment dans la console Amazon S3.

### Warning

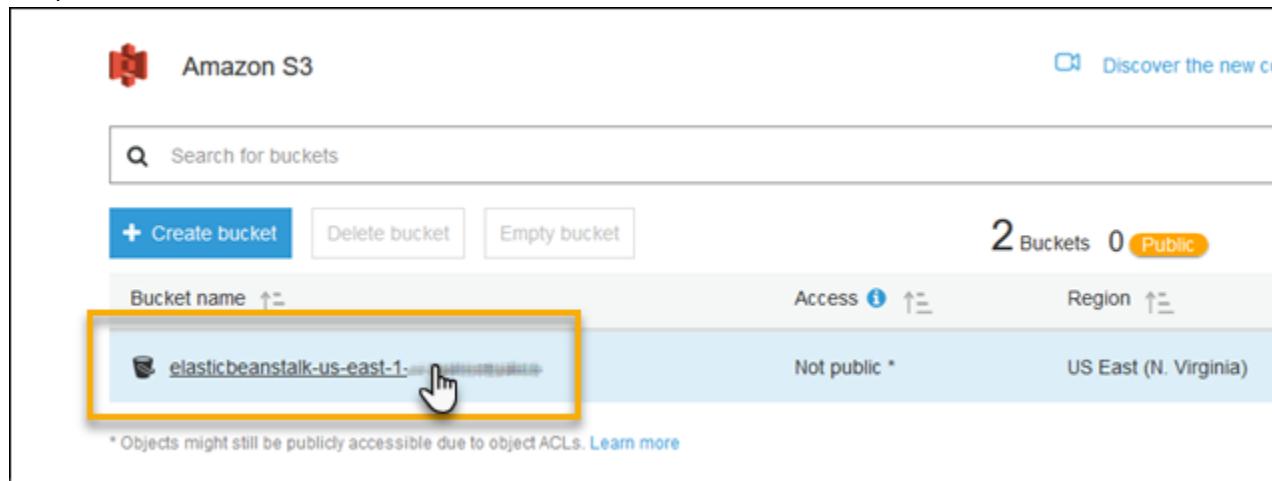
Si vous supprimez un compartiment créé par Elastic Beanstalk dans votre compte, et que vous avez toujours des applications existantes et des environnements en cours d'exécution dans la région correspondante, vos applications risquent de ne plus fonctionner correctement. Par exemple :

- Lorsqu'un environnement monte en puissance, Elastic Beanstalk doit être en mesure d'identifier la version d'application de l'environnement dans le compartiment Amazon S3 et de l'utiliser pour lancer de nouvelles instances Amazon EC2.
- Lorsque vous créez une plateforme personnalisée, Elastic Beanstalk utilise le stockage Amazon S3 temporaire au cours du processus de création.

Nous vous recommandons de supprimer certains objets inutiles de votre compartiment Elastic Beanstalk Amazon S3, au lieu de supprimer l'ensemble du compartiment.

Pour supprimer un compartiment de stockage Elastic Beanstalk (console)

1. Ouvrez la [console Amazon S3](#).
2. Ouvrez la page du compartiment de stockage Elastic Beanstalk en choisissant le nom du compartiment.



3. Choisissez l'onglet Autorisations.
4. Choisissez Stratégie de compartiment.
5. Sélectionnez Supprimer.

The screenshot shows the AWS S3 Bucket Policy editor. At the top, there are tabs for Overview, Properties, Permissions, and Management. Below these are sub-tabs for Access Control List, Bucket Policy, and CORS configuration. The Bucket Policy tab is selected. A text area titled "Bucket policy editor ARN: arn:aws:s3:::elasticbeanstalk-us-east-1" contains a JSON policy:

```
1  {
2    "Version": "2008-10-17",
3    "Statement": [
4      {
5        "Sid": "eb-ad51e",
6        "Effect": "Allow",
7        "Principal": "
```

6. Revenez à la page principale de la console Amazon S3, puis sélectionnez le compartiment de stockage Elastic Beanstalk en cliquant sur sa ligne n'importe où sauf sur le nom du compartiment.

The screenshot shows the AWS S3 buckets list. At the top, there are buttons for Create bucket, Delete bucket, and Empty bucket. It displays 2 Buckets and 0 Public buckets. A table lists the buckets:

Bucket name	Access	Region
elasticbeanstalk-us-east-1	Not public *	US East (N. Virginia)

\* Objects might still be publicly accessible due to object ACLs. [Learn more](#)

7. Choisissez Supprimer le compartiment.
8. Tapez le nom du compartiment, puis choisissez Confirmer.

## Utilisation d'Elastic Beanstalk avec Amazon VPC

Vous pouvez utiliser Amazon [Virtual Private Cloud](#) (Amazon VPC) pour créer un réseau sécurisé pour votre application Elastic Beanstalk et les ressources AWS connexes. Lorsque vous créez votre environnement, vous choisissez les VPC, sous-réseaux et groupes de sécurité utilisés pour vos instances d'application et votre équilibrer de charge. Vous pouvez utiliser la configuration de VPC de votre choix, à condition qu'elle respecte les conditions suivantes.

### Exigences du VPC

- Accès Internet – Les instances peuvent avoir accès à Internet via l'une des méthodes suivantes.
  - Sous-réseau public – Les instances ont une adresse IP publique et utilisent une passerelle Internet pour accéder à Internet.

- Sous-réseau privé – Les instances utilisent un périphérique NAT pour accéder à Internet.

#### Note

Si vous configurez des [points de terminaison d'un VPC \(p. 1021\)](#) dans votre VPC pour vous connecter à la fois aux services `elasticbeanstalk` et `elasticbeanstalk-healthd`, l'accès Internet est facultatif et est uniquement requis si votre application en a spécifiquement besoin. Sans points de terminaison de VPC, votre VPC doit avoir accès à Internet.

Le VPC par défaut qu'Elastic Beanstalk met en place pour vous fournir un accès Internet.

Elastic Beanstalk ne prend pas en charge les paramètres de proxy comme `HTTPS_PROXY` pour la configuration d'un proxy web.

- NTP – Les instances dans votre environnement Elastic Beanstalk utilisent le protocole NTP (Network Time) pour synchroniser l'horloge système. Si les instances ne peuvent pas communiquer sur le port UDP 123, l'horloge peut se désynchroniser, ce qui entraîne des problèmes avec les rapports d'intégrité d'Elastic Beanstalk. Assurez-vous que vos groupes de sécurité VPC et ACL réseau autorisent le trafic UDP entrant et sortant sur le port 123 pour éviter ces problèmes.

Le référentiel [elastic-beanstalk-samples](#) fournit des modèles AWS CloudFormation que vous pouvez utiliser pour créer un VPC destiné à vos environnements Elastic Beanstalk.

Pour créer des ressources avec un modèle AWS CloudFormation

1. Clonez le référentiel d'exemples ou téléchargez un modèle à l'aide des liens que vous trouvez dans le fichier [README](#).
2. Ouvrez la [console AWS CloudFormation](#).
3. Choisissez Créer une pile.
4. Choisissez Télécharger un modèle sur Amazon S3.
5. Choisissez Charger le fichier et chargez le fichier de modèle depuis votre ordinateur local.
6. Choisissez Suivant et suivez les instructions pour créer une pile avec les ressources dans le modèle.

Lorsque la création de la pile est terminée, vérifiez l'onglet Outputs (Sorties) pour trouver l'ID du VPC et les ID de sous-réseau. Utilisez ces ressources pour configurer le VPC dans la [catégorie de configuration Réseau \(p. 457\)](#) de l'assistant de création d'un environnement.

#### Rubriques

- [VPC public \(p. 1007\)](#)
- [VPC public/privé \(p. 1008\)](#)
- [VPC privé \(p. 1008\)](#)
- [Exemple : Lancement d'une application Elastic Beanstalk dans un VPC avec des hôtes bastion \(p. 1010\)](#)
- [Exemple : Lancement d'un Elastic Beanstalk dans un VPC avec Amazon RDS \(p. 1015\)](#)
- [Utilisation d'Elastic Beanstalk avec les points de terminaison VPC \(p. 1021\)](#)

## VPC public

AWS CloudFormation modèle – [vpc-public.yaml](#)

#### Paramètres (charge équilibrée)

- Visibilité de l'équilibrEUR de charge – Public
- Sous-réseaux de l'équilibrEUR de charge – Public pour les deux

- Adresse IP publique de l'instance – Activée
- Sous-réseaux de l'instance – Public pour les deux
- Groupes de sécurité de l'instance – Ajouter le groupe de sécurité par défaut

#### Paramètres (une seule instance)

- Sous-réseaux de l'instance – Public pour un des deux
- Groupes de sécurité de l'instance – Ajouter le groupe de sécurité par défaut

Une conception élémentaire de VPC public uniquement inclut un ou plusieurs sous-réseaux publics, une passerelle Internet et un groupe de sécurité par défaut qui autorise le trafic entre les différentes ressources du VPC. Lorsque vous créez un environnement dans le VPC, Elastic Beanstalk crée des ressources supplémentaires qui varient en fonction du type d'environnement.

#### Ressources VPC

- Instance unique – Elastic Beanstalk crée un groupe de sécurité pour l'instance d'application qui autorise le trafic sur le port 80 depuis Internet, et attribue une adresse IP Elastic à l'instance pour qu'elle soit associée à une adresse IP publique. Le nom de domaine de l'environnement est résolu avec l'adresse IP publique de l'instance.
- Équilibrage de charge – Elastic Beanstalk crée un groupe de sécurité pour l'équilibrEUR de charge qui autorise le trafic sur le port 80 depuis Internet, et un groupe de sécurité pour les instances d'application qui autorisent le trafic depuis le groupe de sécurité de l'équilibrEUR de charge. Le nom de domaine de l'environnement est résolu avec le nom de domaine public de l'équilibrEUR de charge.

Cette option est similaire à la façon dont Elastic Beanstalk gère la mise en réseau lorsque vous utilisez le VPC par défaut. La sécurité d'un sous-réseau public dépend des groupes de sécurité d'instance et d'équilibrEUR de charge créés par Elastic Beanstalk. C'est là la configuration la moins coûteuse, car aucun passerelle NAT n'est requise.

## VPC public/privé

AWS CloudFormation modèle – [vpc-privatepublic.yaml](#)

#### Paramètres (charge équilibrée)

- Visibilité de l'équilibrEUR de charge – Public
- Sous-réseaux de l'équilibrEUR de charge – Public pour les deux
- Adresse IP publique de l'instance – Désactivée
- Sous-réseaux de l'instance – Privé pour les deux
- Groupes de sécurité de l'instance – Ajouter le groupe de sécurité par défaut

Pour une sécurité optimale, ajoutez des sous-réseaux privés à votre VPC pour créer une configuration publique-privée. Cette configuration nécessite un équilibrEUR de charge et une passerelle NAT dans les sous-réseaux publics, et vous permet d'exécuter vos instances d'application, votre base de données et les autres ressources dans des sous-réseaux privés. Les instances des sous-réseaux privés peuvent uniquement communiquer avec Internet via l'équilibrEUR de charge et la passerelle NAT.

## VPC privé

AWS CloudFormation modèle – [vpc-private.yaml](#)

### Paramètres (charge équilibrée)

- Visibilité de l'équilibrer de charge – Privé
- Sous-réseaux de l'équilibrer de charge – Les deux sous-réseaux privés
- Adresse IP publique de l'instance – Désactivée
- Sous-réseaux de l'instance – Privé pour les deux
- Groupes de sécurité de l'instance – Ajouter le groupe de sécurité par défaut

Pour les applications internes qui ne doivent pas avoir d'accès depuis Internet, vous pouvez exécuter toutes les applications dans des sous-réseaux privés et configurer l'équilibrer de charge de façon à ce qu'il soit interne (remplacez la valeur de Load balancer visibility (Visibilité de l'équilibrer de charge) par Internal (Interne). Ce modèle crée un VPC sans sous-réseaux publics et sans passerelle Internet. Utilisez cette configuration pour les applications qui doivent être accessibles à partir du même VPC ou d'un VPN associé uniquement.

## Exécution d'un environnement Elastic Beanstalk dans un VPC privé

Lorsque vous créez votre environnement Elastic Beanstalk dans un VPC privé, l'environnement n'a pas accès à Internet. Votre application peut avoir besoin d'accéder au service Elastic Beanstalk ou à d'autres services. Votre environnement peut utiliser des rapports d'intégrité améliorés et, dans ce cas, les instances d'environnement envoient des informations d'intégrité au service d'intégrité amélioré. Et le code Elastic Beanstalk sur les instances d'environnement envoie du trafic vers d'autres services AWS, et un autre trafic vers des points de terminaison non AWS (par exemple, pour télécharger des packages de dépendance pour votre application). Voici quelques étapes que vous devrez peut-être emprunter dans ce cas, pour vous assurer que votre environnement fonctionne correctement.

- Configurez des points de terminaison d'un VPC pour Elastic Beanstalk – Elastic Beanstalk et son service d'intégrité amélioré prennent en charge les points de terminaison d'un VPC, ce qui garantit que le trafic vers ces services reste à l'intérieur du réseau Amazon et ne nécessite pas d'accès Internet. Pour de plus amples informations, veuillez consulter [the section called "Points de terminaison d'un VPC" \(p. 1021\)](#).
- Configurer les points de terminaison d'un VPC pour des services supplémentaires – les instances Elastic Beanstalk envoient du trafic vers plusieurs autres services AWS en votre nom : Amazon Simple Storage Service (Amazon S3), Amazon Simple Queue Service (Amazon SQS), AWS CloudFormation et Amazon CloudWatch Logs. Vous devez également configurer les points de terminaison de VPC pour ces services. Pour de plus amples informations sur les points de terminaison VPC, y compris les liens par service, veuillez consulter les [points de terminaison VPC](#) dans le Guide de l'utilisateur Amazon VPC.

### Note

Certains services AWS, y compris Elastic Beanstalk, prennent en charge les terminaux VPC dans un nombre limité de régions AWS. Lorsque vous concevez votre solution de VPC privé, vérifiez qu'Elastic Beanstalk et les autres services dépendants mentionnés ici prennent en charge les points de terminaison d'un VPC dans la région AWS que vous choisissez.

- Fournir une image Docker privée – Dans un environnement [Docker \(p. 46\)](#), le code sur les instances de l'environnement peut essayer d'extraire votre image Docker configurée à partir d'Internet lors de la création de l'environnement et échouer. Pour éviter cet échec, [créez une image Docker personnalisée \(p. 57\)](#) sur votre environnement ou utilisez une image Docker stockée dans [Amazon Elastic Container Registry](#) (Amazon ECR) et [configurez un point de terminaison VPC pour le service Amazon ECR](#).
- Activer les noms DNS – Le code Elastic Beanstalk sur les instances d'environnement envoie du trafic à tous les services AWS à l'aide de leurs points de terminaison publics. Pour vous assurer que ce trafic passe bien, choisissez l'option Enable DNS name (Activer le nom DNS) lorsque vous configurez tous les points de terminaison d'un VPC d'interface. Cela ajoute une entrée DNS dans votre VPC qui mappe le point de terminaison du service public au point de terminaison de VPC d'interface.

## Important

Si votre VPC n'est pas privé et dispose d'un accès Internet public, et si Enable DNS name (Activer le nom DNS) est désactivé pour n'importe quel point de terminaison de VPC, le trafic vers le service correspondant passe par l'Internet public. Ce n'est probablement pas votre intention. Ce problème est facile à détecter avec un VPC privé, car il empêche ce trafic de passer et vous recevez des erreurs. Cependant, avec un VPC public, vous n'obtenez aucune indication.

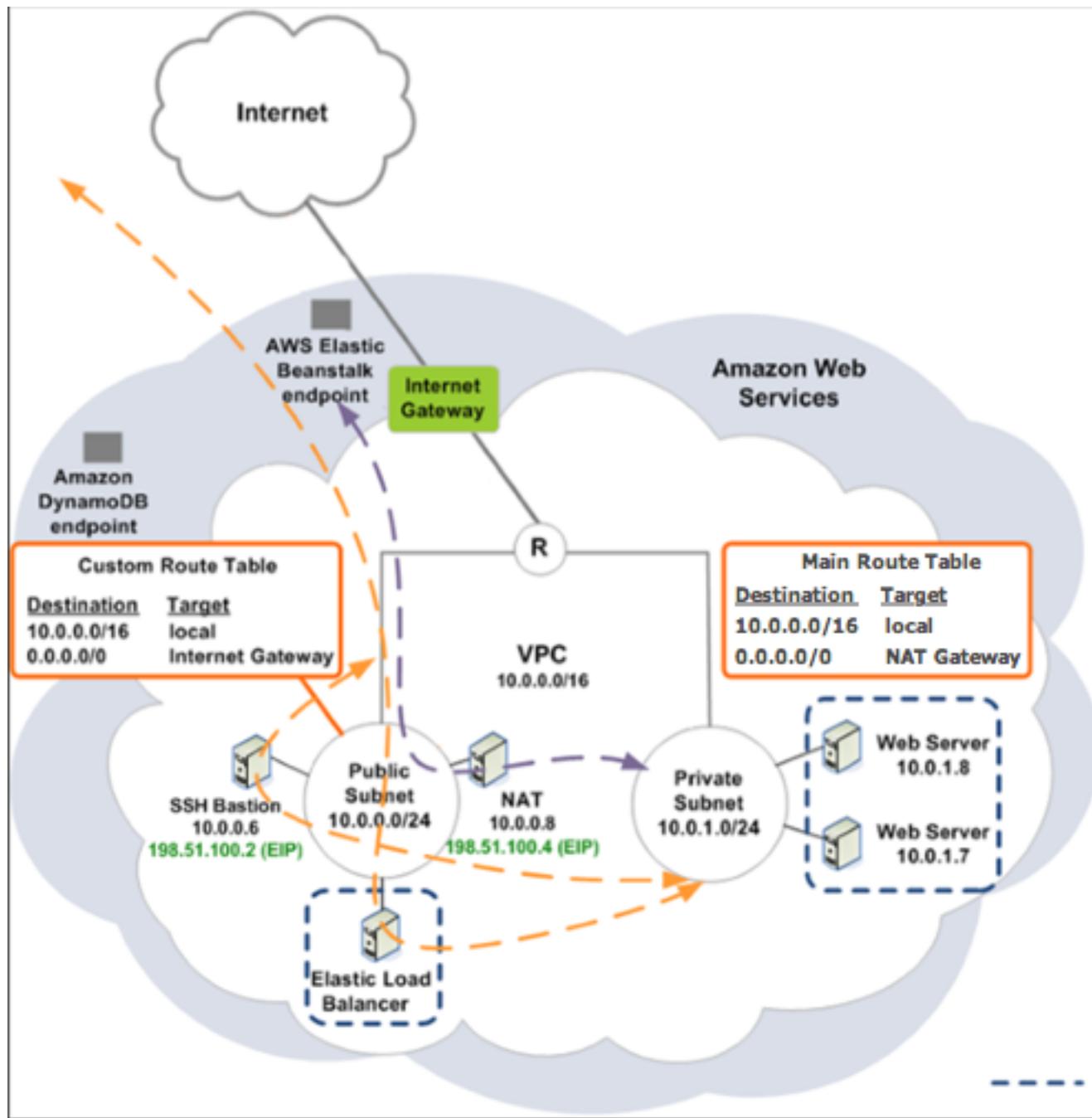
- Inclure les dépendances de l'application – Si votre application a des dépendances telles que des packages d'exécution de langage, elle peut essayer de les télécharger et de les installer à partir d'Internet lors de la création de l'environnement et échouer. Pour éviter cet échec, incluez tous les packages de dépendance dans le bundle source de votre application.
- Utiliser une version de plateforme actuelle – Assurez-vous que votre environnement utilise une version de plateforme qui a été publiée le 24 février 2020 ou plus tard. Plus précisément, utilisez une version de plateforme qui a été publiée avec ou après l'une de ces deux mises à jour : [Linux Update 2020-02-28](#), [Windows Update 2020-02-24](#).

## Note

La raison pour laquelle il vous faut une version de plateforme mise à jour est que les anciennes versions présentaient un problème qui empêcherait les entrées DNS créées par l'option Enable DNS name (Activer le nom DNS) de fonctionner correctement pour Amazon SQS.

## Exemple : Lancement d'une application Elastic Beanstalk dans un VPC avec des hôtes bastion

Si vos instances Amazon EC2 se trouvent à l'intérieur d'un sous-réseau privé, vous ne serez pas en mesure de vous y connecter à distance. Pour vous connecter à vos instances, vous pouvez configurer des serveurs bastions dans le sous-réseau public pour servir de proxy. Par exemple, vous pouvez définir des achemineurs de port SSH ou des passerelles RDP dans le sous-réseau public pour définir des serveurs proxy du trafic à destination de vos serveurs de base de données et en provenance de votre propre réseau. Cette section fournit un exemple sur la manière de créer d'un VPC avec un sous-réseau privé et public. Les instances sont trouvent à l'intérieur du sous-réseau privé et l'hôte bastion, la passerelle NAT ainsi que l'équilibriseur de charge se trouvent à l'intérieur du sous-réseau public. Votre infrastructure ressemblera au schéma suivant.



Pour déployer une application Elastic Beanstalk à l'intérieur d'un VPC à l'aide d'un hôte bastion, suivez les étapes décrites dans les sous-sections suivantes.

#### Étapes

- Création d'un VPC avec des sous-réseaux public et privé (p. 1012)
- Crédit et configuration du groupe de sécurité de l'hôte bastion (p. 1012)
- Mise à jour du groupe de sécurité de l'instance (p. 1014)
- Création d'un hôte bastion (p. 1014)

## Création d'un VPC avec des sous-réseaux public et privé

Réalisez toutes les procédures décrites dans [VPC public/privé \(p. 1008\)](#). Lors du déploiement de l'application, vous devez spécifier une paire de clés Amazon EC2 pour les instances afin que vous puissiez vous y connecter à distance. Pour plus d'informations sur la façon de spécifier la paire de clés d'instance, consultez [Instances Amazon EC2 de votre environnement Elastic Beanstalk \(p. 538\)](#).

## Création et configuration du groupe de sécurité de l'hôte bastion

Créez un groupe de sécurité pour l'hôte bastion et ajoutez des règles qui autorisent le trafic SSH entrant à partir d'Internet et le trafic SSH sortant vers le sous-réseau privé qui contient les instances Amazon EC2.

Pour créer le groupe de sécurité de l'hôte bastion

1. Ouvrez la console Amazon VPC à l'adresse <https://console.aws.amazon.com/vpc/>.
2. Dans le panneau de navigation, sélectionnez Groupes de sécurité.
3. Sélectionnez Créer un groupe de sécurité.
4. Dans la boîte de dialogue Créez un groupe de sécurité, entrez les informations suivantes et choisissez Oui, créer.

Balise Nom (facultatif)

Entrez une étiquette de nom pour le groupe de sécurité.

Nom du groupe

Entrez le nom du groupe de sécurité.

Description

Entrez une description pour le groupe de sécurité.

VPC

Sélectionnez votre VPC.

Le groupe de sécurité est créé et apparaît sur la page Groupes de sécurité. Notez qu'il possède un ID (par exemple, ..., sg-xxxxxx). Vous devrez peut-être activer la colonne ID du groupe en cliquant sur Afficher/Masquer dans l'angle supérieur droit de la page.

Pour configurer le groupe de sécurité de l'hôte bastion

1. Dans la liste des groupes de sécurité, cochez la case correspondant au groupe de sécurité que vous venez de créer pour votre hôte bastion.
2. Sous l'onglet Règles entrantes, choisissez Modifier.
3. Si nécessaire, sélectionnez Add another rule (Ajouter une autre règle).
4. Si votre hôte bastion est une instance Linux, sous Type, sélectionnez SSH.

Si votre hôte bastion est une instance Windows, sous Type, sélectionnez RDP.

5. Entrez la plage CIDR source souhaitée dans le champ Source et choisissez Save (Enregistrer).

The screenshot shows the AWS VPC Dashboard. On the left, there's a sidebar with links like VPC Dashboard, Virtual Private Cloud, Your VPCs, Subnets, Route Tables, Internet Gateways, DHCP Option Sets, Elastic IPs, Security, Network ACLs, and Security Groups (which is selected). The main area has tabs for Create Security Group, Delete, Filter: All Security Groups, Search Security Groups..., Summary, Inbound Rules (which is selected), Outbound Rules, and Tags. There's a table for Inbound Rules with columns Type, Protocol, Port Range, Source, and Remove. A row is being edited for SSH, TCP (6), port 22, and source 0.0.0.0/0. Buttons for Cancel and Save are at the bottom of the table.

6. Sous l'onglet Outbound Rules (Règles sortantes), choisissez Edit (Modifier).
7. Si nécessaire, sélectionnez Add another rule (Ajouter une autre règle).
8. Sous Type, sélectionnez le type que vous avez spécifiée pour la règle entrante.
9. Dans le champ Source, entrez la plage d'adresses CIDR du sous-réseau des hôtes dans le sous-réseau privé du VPC.

Pour la trouver :

- a. Ouvrez la console Amazon VPC à l'adresse <https://console.aws.amazon.com/vpc/>.
- b. Dans le panneau de navigation, choisissez Sous-réseaux.
- c. Notez la valeur sous IPv4 CIDR pour chaque Availability Zone (Zone de disponibilité) dans laquelle vous avez des hôtes avec lesquels vous souhaitez que l'hôte bastion établisse une passerelle.

#### Note

Si vous avez des hôtes dans plusieurs zones de disponibilité, créez une règle sortante pour chacun de ces zones de disponibilité.

Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4
	subnet-9de72a91	available	vpc-7523e317	172.31.0.0/20	4091
	subnet-04575af9	available	vpc-7523e317	172.31.16.0/20	4090
	subnet-3d33734c	available	vpc-7523e317	172.31.32.0/20	4091
	subnet-caa74a67	available	vpc-7523e317	172.31.48.0/20	4091
	subnet-9dfcbf9f5	available	vpc-7523e317	172.31.64.0/20	4091

10. Choisissez Enregistrer.

## Mise à jour du groupe de sécurité de l'instance

Par défaut, le groupe de sécurité que vous avez créé pour vos instances ne permet pas le trafic entrant. Même si Elastic Beanstalk modifiera le groupe par défaut pour les instances pour autoriser le trafic SSH, vous devez modifier votre groupe de sécurité d'instance personnalisée pour autoriser le trafic RDP si vos instances sont des instances Windows.

Pour mettre à jour le groupe de sécurité de l'instance pour RDP

1. Dans la liste des groupes de sécurité, cochez la case correspondant au groupe de sécurité de l'instance.
2. Dans l'onglet Entrant, choisissez Modifier.
3. Si nécessaire, sélectionnez Add another rule (Ajouter une autre règle).
4. Entrez les valeurs suivantes, puis sélectionnez Save (Enregistrer).

Type

RDP

Protocole

TCP

Plage de ports

3389

Source

Entrez l'ID du groupe de sécurité de l'hôte bastion (par exemple, sg-8a6f71e8) et choisissez Save (Enregistrer).

## Création d'un hôte bastion

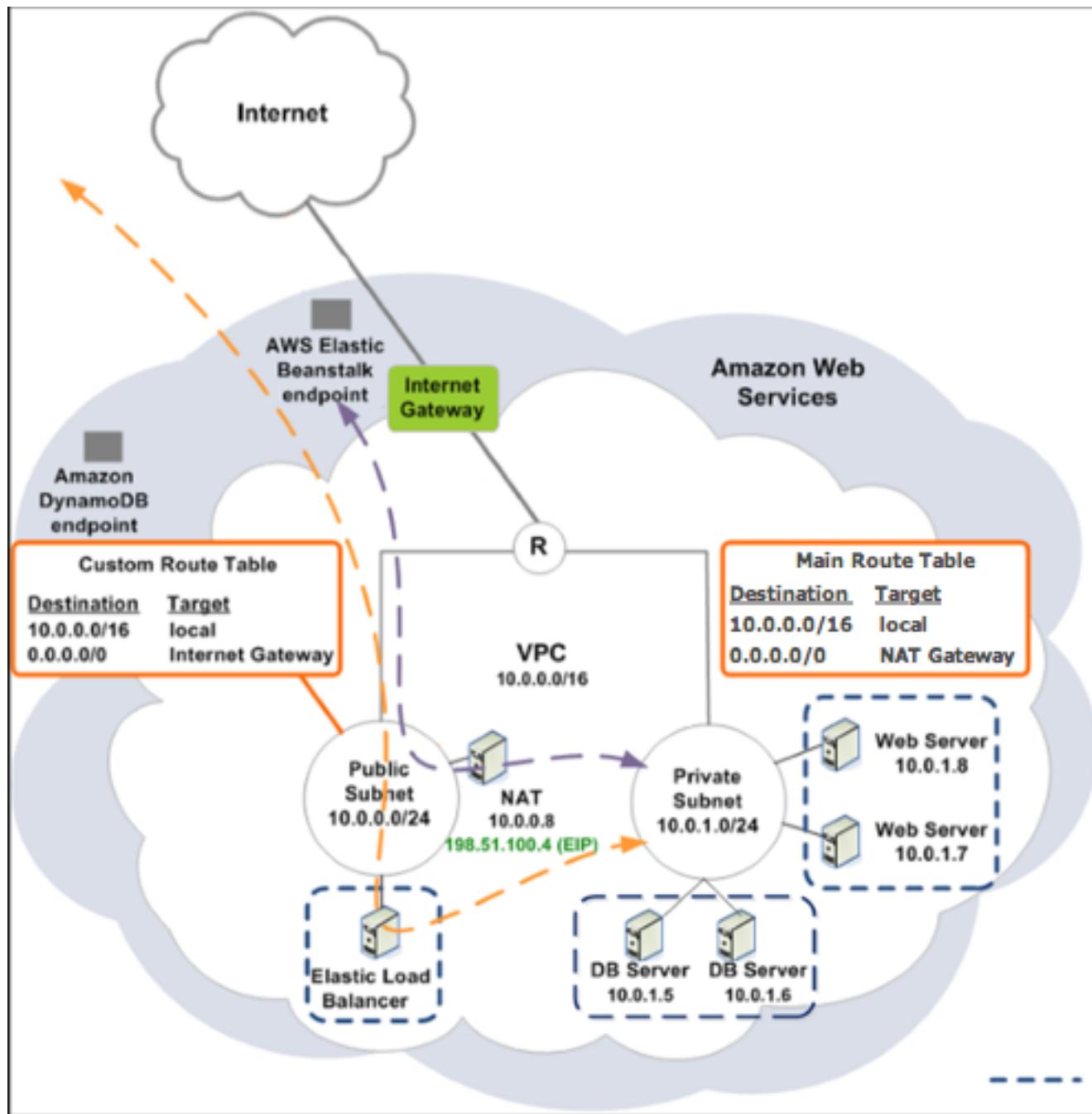
Pour créer un hôte bastion, vous lancez une instance Amazon EC2 dans votre sous-réseau public qui servira d'hôte bastion.

Pour de plus amples informations sur la configuration d'un hôte bastion pour les instances Windows du sous-réseau privé, veuillez consulter [Contrôle de l'accès réseau aux instances EC2 à l'aide d'un serveur Bastion](#).

Pour de plus amples informations sur la configuration d'un hôte bastion pour les instances Linux sur le sous-réseau privé, veuillez consulter l'article de blog [Connexion sécurisée aux instances Linux exécutées dans un VPC privé Amazon](#).

## Exemple : Lancement d'un Elastic Beanstalk dans un VPC avec Amazon RDS

Cette section vous guide à travers les tâches de déploiement d'une application Elastic Beanstalk avec Amazon RDS dans un VPC à l'aide d'une passerelle NAT. Votre infrastructure ressemblera au schéma suivant.



#### Note

Si vous n'avez pas utilisé une instance de base de données avec votre application auparavant, essayez [d'en ajouter une à un environnement de test \(p. 617\)](#) et [d'établir une connexion à une instance de base de données externe \(p. 991\)](#) avant d'ajouter une configuration VPC au mélange.

## Création d'un VPC avec des sous-réseaux public et privé

Vous pouvez utiliser la [console Amazon VPC](#) pour créer un VPC.

### Pour créer un VPC

1. Connectez-vous à la [console Amazon VPC](#).
2. Dans le panneau de navigation, choisissez Tableau de bord du VPC. Ensuite, choisissez Créeer VPC.
3. Choisissez VPC avec des sous-réseaux publics et privés, puis Sélectionner.

**Step 1: Select a VPC Configuration**

VPC with a Single Public Subnet	In addition to containing a public subnet, this configuration adds a private subnet whose instances are not addressable from the Internet. Instances in the private subnet can establish outbound connections to the Internet via the public subnet using Network Address Translation (NAT).  Creates:  A /16 network with two /24 subnets. Public subnet instances use Elastic IPs to access the Internet. Private subnet instances access the Internet via Network Address Translation (NAT). (Hourly charges for NAT devices apply.)
<b>VPC with Public and Private Subnets</b>	<b>Select</b>
VPC with Public and Private Subnets and Hardware VPN Access	
VPC with a Private Subnet Only and Hardware VPN Access	

[Cancel and Exit](#)

4. Votre équilibrer de charge Elastic Load Balancing et vos instances Amazon EC2 doivent se trouver dans la même zone de disponibilité pour pouvoir communiquer ensemble. Choisissez la même zone de disponibilité dans chaque liste Zone de disponibilité.

**Step 2: VPC with Public and Private Subnets**

IPv4 CIDR block: <input type="text" value="10.0.0.0/16"/> (65536 IP addresses available)	IPv6 CIDR block: <input checked="" type="radio"/> No IPv6 CIDR Block <input type="radio"/> Amazon provided IPv6 CIDR block
VPC name: <input type="text"/>	
Public subnet's IPv4 CIDR: <input type="text" value="10.0.0.0/24"/> (251 IP addresses available)	Availability Zone: <input type="text" value="No Preference"/>
Public subnet name: <input type="text" value="Public subnet"/>	Private subnet's IPv4 CIDR: <input type="text" value="10.0.1.0/24"/> (251 IP addresses available)
	Availability Zone: <input type="text" value="No Preference"/>
Private subnet name: <input type="text" value="Private subnet"/>	You can add more subnets after AWS creates the VPC.
Specify the details of your NAT gateway (NAT gateway rates apply).	
Elastic IP Allocation ID: <input type="text"/>	<a href="#">Use a NAT instance instead</a>
Service endpoints	
<a href="#">Add Endpoint</a>	
Enable DNS hostnames: <input checked="" type="radio"/> Yes <input type="radio"/> No	
Hardware tenancy: <input type="text" value="Default"/>	
Enable ClassicLink: <input type="radio"/> Yes <input checked="" type="radio"/> No	

[Cancel and Exit](#) [Back](#) [Create VPC](#)

5. Choisissez une adresse IP Elastic pour votre passerelle NAT.
6. Sélectionnez Create VPC (Créer un VPC).

L'Assistant commence à créer le VPC, les sous-réseaux et la passerelle Internet. Il met également à jour la table de routage principale et crée une table de routage personnalisée. Enfin, il crée une passerelle NAT dans le sous-réseau public.

#### Note

Vous pouvez choisir de lancer une instance NAT dans le sous-réseau public au lieu d'une passerelle NAT. Pour plus amples informations, veuillez consulter [Scénario 2 : VPC avec des sous-réseaux publics et privés \(NAT\)](#) dans le Guide de l'utilisateur Amazon VPC.

7. Une fois que le VPC est créé avec succès, vous obtenez un ID de VPC. Vous avez besoin de cette valeur pour l'étape suivante. Pour voir l'ID de votre VPC, choisissez Vos VPC dans le volet gauche de la [console Amazon VPC](#).

Name	VPC ID	State	IPv4 CIDR	DHCP options set	Route table	Network ACL
vpc-f56cff91	available	172.31.0.0/16	dopt-6e7bda0b	rtb-4f0f472b	acl-ca059fae	

## Création d'un groupe de sous-réseaux DB

Un groupe de sous-réseaux DB pour un VPC est un ensemble de sous-réseaux (généralement privés) que vous pouvez désigner pour vos instances backend DB RDS. Chaque groupe de sous-réseau DB doit posséder au moins un sous-réseau pour chaque zone de disponibilité dans une région AWS donnée. Pour de plus amples informations, veuillez consulter [Création d'un sous-réseau dans votre VPC](#).

### Création d'un groupe de sous-réseaux DB

- Ouvrez la [console Amazon RDS](#) .
- Dans le volet de navigation, choisissez Subnet groups (Groupes de sous-réseaux).
- Choisissez Create DB Subnet Group (Créer groupe de sous-réseaux de base de données).
- Choisissez Name (Nom), puis saisissez le nom de votre groupe de sous-réseaux DB.
- Choisissez Description, puis décrivez votre groupe de sous-réseaux DB.
- Pour VPC, choisissez l'ID du VPC que vous avez créé précédemment.
- Dans Add subnets (Ajouter des sous-réseaux), choisissez Add all the subnets related to this VPC (Ajouter tous les sous-réseaux associés à ce VPC).

Availability zone	Subnet ID	CIDR block	Action
us-east-2c	subnet-da3408ae	10.0.1.0/24	<button>Remove</button>
us-east-2c	subnet-db3408af	10.0.0.0/24	<button>Remove</button>
us-east-2b	subnet-4f195024	10.0.2.0/24	<button>Remove</button>
us-east-2a	subnet-fe064f95	10.0.3.0/24	<button>Remove</button>

**Cancel** **Create**

- Lorsque vous avez terminé, sélectionnez Create.

Votre nouveau groupe de sous-réseaux DB s'affiche dans la liste des groupes de sous-réseaux de base de données de la console Amazon RDS. Vous pouvez le choisir pour afficher des détails, tels que tous les sous-réseaux associés à ce groupe, dans le volet des détails en bas de la page.

## Déploiement sur Elastic Beanstalk

Une fois que vous avez configuré votre VPC, vous pouvez y créer votre environnement et déployer votre application sur Elastic Beanstalk. Vous pouvez le faire à l'aide de la console Elastic Beanstalk ou utiliser les boîtes à outils AWS, AWS CLI, la CLI EB ou l'API Elastic Beanstalk. Si vous utilisez la console Elastic Beanstalk, il vous suffit de télécharger votre fichier .war ou .zip et de sélectionner les paramètres VPC dans l'assistant. Elastic Beanstalk crée ensuite votre environnement dans votre VPC et déploie votre application. Vous pouvez également utiliser les boîtes à outils AWS, AWS CLI, la CLI EB ou l'API Elastic Beanstalk pour déployer votre application. Pour ce faire, vous devez définir vos paramètres VPC dans un fichier de configuration et déployer ce fichier avec votre groupe source. Cette rubrique fournit des instructions pour les deux méthodes.

### Déploiement avec la console Elastic Beanstalk

La console Elastic Beanstalk vous guide à travers la création de votre nouvel environnement dans votre VPC. Vous devez fournir un fichier .war (pour les applications Java) ou un fichier .zip (pour toutes les autres applications). Sur la page Configuration VPC de l'assistant d'environnement Elastic Beanstalk, vous devez effectuer les sélections suivantes :

#### VPC

Sélectionnez votre VPC.

#### Groupe de sécurité VPC

Sélectionnez le groupe de sécurité d'instance que vous avez créé précédemment.

#### Visibilité ELB

Sélectionnez `External` si votre équilibreur de charge doit être publiquement disponible, ou `Internal` si l'équilibreur de charge ne doit être disponible que dans votre VPC.

Sélectionnez les sous-réseaux pour votre équilibreur de charge et les instances EC2. Veillez à sélectionner le sous-réseau public pour l'équilibreur de charge et le sous-réseau privé pour vos instances Amazon EC2. Par défaut, l'assistant de création de VPC crée le sous-réseau public dans 10.0.0.0/24 et le sous-réseau privé dans 10.0.1.0/24.

Vous pouvez afficher les ID de vos sous-réseaux en cliquant sur Subnets (Sous-réseaux) dans la [console Amazon VPC](#).

Name	Subnet ID	State	VPC	IPv4 CIDR	Available IPv4	Availability Zone
	subnet-3ba3c75e	available	vpc-f56cff91	172.31.64.0/20	4091	us-east-1a
<b>selected</b>	<b>subnet-ec18feb4</b>	<b>available</b>	<b>vpc-f56cff91</b>	<b>172.31.16.0/20</b>	<b>4089</b>	<b>us-east-1d</b>

**subnet-ec18feb4**

**Summary**    **Route Table**    **Network ACL**    **Flow Logs**    **Tags**

Subnet ID: subnet-ec18feb4    Availability Zone: us-east-1d  
 IPv4 CIDR: 172.31.16.0/20    Route table: rtb-4f0f472b  
 IPv6 CIDR:    Network ACL: acl-ca059fae  
 State: available    Default subnet: yes  
 VPC: vpc-f56cff91    Auto-assign Public IP: yes  
 Available IPs: 4089    Auto-assign IPv6 address: no

## Déploiement avec les boîtes à outils AWS, la CLI EB, AWS CLI ou l'API

Lorsque vous déployez votre application sur Elastic Beanstalk à l'aide des boîtes à outils AWS, de la CLI EB, de AWS CLI ou de l'API, vous pouvez spécifier vos paramètres VPC dans un fichier et déployer ce dernier avec votre solution groupée de fichiers source. Pour plus d'informations, consultez [Personnalisation d'environnement avancée avec fichiers de configuration \(.ebextensions\) \(p. 737\)](#).

Lorsque vous mettez à jour les paramètres de l'option, vous devez spécifier au moins les éléments suivants :

- **VPCId** – Contient l'ID du VPC.
- **Subnets** (Sous-réseaux) – Contient l'ID du sous-réseau du groupe Auto Scaling. Dans cet exemple, il s'agit de l'ID du sous-réseau privé.
- **ELBSubnets** – Contient l'ID du sous-réseau pour l'équilibreur de charge. Dans cet exemple, il s'agit de l'ID du sous-réseau public.
- **SecurityGroups** – Contient l'ID des groupes de sécurité.
- **DBSubnets** – Contient l'ID des sous-réseaux de base de données.

### Note

Lorsque vous utilisez des sous-réseaux DB, vous devez créer des sous-réseaux supplémentaires dans votre VPC afin de couvrir toutes les zones de disponibilité dans la région AWS.

Le cas échéant, vous pouvez également spécifier les informations suivantes :

- **ELBScheme** – Spécifiez `internal` pour créer un équilibreur de charge interne dans votre VPC afin qu'aucun accès à votre application Elastic Beanstalk ne soit possible en dehors de votre VPC.

Voici un exemple des paramètres d'options que vous pouvez utiliser lors du déploiement de votre application Elastic Beanstalk à l'intérieur d'un VPC. Pour de plus amples informations sur les paramètres d'options VPC (y compris les exemples pour savoir comment les spécifier, les valeurs par défaut et les valeurs valides), veuillez consulter le tableau d'espace de noms `aws:ec2:vpc` dans [Options de configuration \(p. 658\)](#).

```
option_settings:
  - namespace: aws:autoscaling:launchconfiguration
```

```
option_name: EC2KeyName
value: ec2keypair

- namespace: aws:ec2:vpc
  option_name: VPCId
  value: vpc-170647c

- namespace: aws:ec2:vpc
  option_name: Subnets
  value: subnet-4f195024

- namespace: aws:ec2:vpc
  option_name: ELBSubnets
  value: subnet-fe064f95

- namespace: aws:ec2:vpc
  option_name: DBSubnets
  value: subnet-fg148g78

- namespace: aws:autoscaling:launchconfiguration
  option_name: InstanceType
  value: m1.small

- namespace: aws:autoscaling:launchconfiguration
  option_name: SecurityGroups
  value: sg-7f1ef110
```

#### Note

Lorsque vous utilisez des sous-réseaux DB, assurez-vous que vous avez des sous-réseaux dans votre VPC afin de couvrir toutes les zones de disponibilité dans la région AWS.

## Utilisation d'Elastic Beanstalk avec les points de terminaison VPC

Un point de terminaison de VPC permet une connexion privée entre votre VPC et les services AWS pris en charge ou les services de point de terminaison de VPC alimentés par AWS PrivateLink sans nécessiter une passerelle Internet, un périphérique NAT, une connexion VPN ou une connexion AWS Direct Connect.

Les instances de votre VPC ne requièrent pas d'adresses IP publiques pour communiquer avec les ressources du service. Le trafic entre votre VPC et les autres services ne quitte pas le réseau Amazon. Pour des informations complètes sur les points de terminaison d'un VPC, veuillez consulter [Points de terminaison d'un VPC](#) dans le Guide de l'utilisateur Amazon VPC.

AWS Elastic Beanstalk prend en charge AWS PrivateLink , qui fournit une connectivité privée au service Elastic Beanstalk et élimine l'exposition du trafic à l'Internet public. Pour permettre à votre application d'envoyer des demandes à Elastic Beanstalk à l'aide d' AWS PrivateLink , vous configurez un type de point de terminaison de VPC appelé point de terminaison d'un VPC d'interface (point de terminaison d'interface). Pour de plus amples informations, consultez [Points de terminaison VPC \( AWS PrivateLink \)](#) dans le Guide de l'utilisateur Amazon VPC.

#### Note

Elastic Beanstalk prend en charge AWS PrivateLink et les points de terminaison d'un VPC d'interface dans un nombre limité de régions AWS. Nous travaillons à étendre notre support à d'autres régions AWS dans un avenir proche.

## Configuration d'un point de terminaison VPC pour Elastic Beanstalk

Pour créer le point de terminaison d'un VPC d'interface pour le service Elastic Beanstalk dans votre VPC, suivez la procédure [Création d'un point de terminaison d'interface](#). Pour Service name (Nom du service), choisissez com.amazonaws.**region**.elasticbeanstalk.

Si votre VPC est configuré avec un accès Internet public, votre application peut toujours accéder à Elastic Beanstalk via Internet à l'aide du point de terminaison public elasticbeanstalk.**region**.amazonaws.com. Pour empêcher cela, assurez-vous que Enable DNS name (Activer le nom DNS) est activé lors de la création du point de terminaison (true par défaut). Cela ajoute une entrée DNS dans votre VPC qui mappe le point de terminaison du service public au point de terminaison de VPC d'interface.

## Configuration d'un point de terminaison de VPC pour l'intégrité améliorée

Si vous avez activé les [rapports d'intégrité améliorée \(p. 837\)](#) pour votre environnement, vous pouvez également configurer les informations d'intégrité améliorée à envoyer à AWS PrivateLink . Les informations d'intégrité améliorées sont envoyées par le healthd démon, un composant Elastic Beanstalk sur vos instances d'environnement, à un service d'intégrité amélioré Elastic Beanstalk distinct. Pour créer un point de terminaison d'un VPC d'interface pour ce service dans votre VPC, suivez la procédure [Création d'un point de terminaison d'interface](#). Pour Service name (Nom du service), choisissez com.amazonaws.**region**.elasticbeanstalk.

### Important

Le daemon healthd envoie des informations d'intégrité améliorée au point de terminaison public, elasticbeanstalk-health.**region**.amazonaws.com. Si votre VPC est configuré avec un accès Internet public et que Enable DNS name (Activer nom DNS) est désactivé pour le point de terminaison de VPC, des informations d'intégrité améliorée circulent sur l'Internet public. Ce n'est probablement pas votre intention lorsque vous configurez un point de terminaison d'un VPC d'intégrité améliorée. Assurez-vous que Enable DNS name (Activer nom DNS) est activé (true par défaut).

## Utilisation de points de terminaison d'un VPC dans un VPC privé

Un VPC privé, ou un sous-réseau privé dans un VPC, n'a pas d'accès Internet public. Vous pouvez exécuter votre environnement Elastic Beanstalk dans un [VPC privé \(p. 1008\)](#) et configurer des points de terminaison de VPC d'interface pour une sécurité renforcée. Dans ce cas, sachez que votre environnement peut essayer de se connecter à Internet pour d'autres raisons que celle de contacter le service Elastic Beanstalk. Pour en savoir plus sur l'exécution d'un environnement dans un VPC privé, consultez [the section called "Exécution d'un environnement Elastic Beanstalk dans un VPC privé" \(p. 1009\)](#).

## Utilisation des stratégies de point de terminaison pour contrôler l'accès avec des points de terminaison de VPC

Par défaut, un point de terminaison de VPC permet un accès complet au service auquel il est associé. Lorsque vous créez ou modifiez un point de terminaison, vous pouvez y attacher une stratégie de point de terminaison.

Une politique de point de terminaison est une politique de ressource AWS Identity and Access Management (IAM) qui contrôle l'accès du point de terminaison au service spécifié. La stratégie de point de terminaison est spécifique au point de terminaison. Elle est distincte des stratégies IAM d'utilisateur ou d'instance que votre environnement peut avoir et ne les remplace pas. Pour de plus amples informations

sur la création et l'utilisation des stratégies de point de terminaison de VPC, veuillez consulter [Contrôler l'accès aux services avec des points de terminaison de VPC](#) dans le Guide de l'utilisateur Amazon VPC.

Dans l'exemple suivant, on refuse à tous les utilisateurs l'autorisation de mettre fin à un environnement via le point de terminaison de VPC et on autorise un accès complet à toutes les autres actions.

```
{  
    "Statement": [  
        {  
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "elasticbeanstalk:TerminateEnvironment",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

#### Note

À ce moment-là, seul le service Elastic Beanstalk principal prend en charge l'attachement d'une stratégie de point de terminaison à son point de terminaison de VPC. Le service d'intégrité améliorée ne prend pas en charge les stratégies de point de terminaison.

# Configuration de votre machine de développement pour une utilisation avec Elastic Beanstalk

Cette page vous montre comment configurer votre machine locale pour le développement d'une application AWS Elastic Beanstalk. Elle couvre la structure des dossiers, le contrôle de la source et les outils CLI.

## Rubriques

- [Création d'un dossier de projet \(p. 1024\)](#)
- [Configuration du contrôle de code source \(p. 1024\)](#)
- [Configuration d'un référentiel distant \(p. 1025\)](#)
- [Installation de l'interface de ligne de commande EB \(p. 1025\)](#)
- [Installation de l'AWS CLI \(p. 1026\)](#)

## Création d'un dossier de projet

Créez un dossier pour votre projet. Vous pouvez stocker ce dossier dans n'importe quel emplacement sur votre disque local à condition d'être autorisé à le lire et à y écrire. La création d'un dossier dans votre dossier d'utilisateur est acceptable. Si vous prévoyez de travailler sur plusieurs applications, créez vos dossiers de projet au sein d'un autre dossier nommé `workspace` ou `projects`, par exemple, pour que tout soit correctement organisé :

```
workspace/
|-- my-first-app
`-- my-second-app
```

Le contenu de votre dossier de projet varie en fonction de l'infrastructure ou du conteneur web utilisé par votre application.

### Note

Évitez les dossiers et les chemins d'accès avec des apostrophes ('') ou des guillemets ("") dans le nom du dossier ou n'importe quel élément de chemin. Certaines commandes Elastic Beanstalk échouent lorsqu'elles sont exécutées au sein d'un dossier avec l'un de ces caractères dans le nom.

## Configuration du contrôle de code source

Configurez le contrôle de code source pour vous protéger contre la suppression accidentelle de fichiers ou de code dans votre dossier de projet, et pour pouvoir annuler les modifications qui entraînent l'interruption de votre projet.

Si vous ne disposez pas d'un système de contrôle de code source, envisagez d'utiliser Git, un outil gratuit et facile à utiliser qui s'intègre parfaitement dans l'interface de ligne de commande (CLI) Elastic Beanstalk. Visitez la [page d'accueil Git](#) pour installer Git.

Suivez les instructions figurant sur le site Web de Git pour installer et configurer Git, puis exécutez `git init` dans votre dossier de projet afin de configurer un référentiel local :

```
~/workspace/my-first-app$ git init
Initialized empty Git repository in /home/local/username/workspace/my-first-app/.git/
```

Lorsque vous ajoutez du contenu à votre dossier de projet et mettez à jour le contenu, validez les modifications dans votre référentiel Git :

```
~/workspace/my-first-app$ git add default.jsp
~/workspace/my-first-app$ git commit -m "add default JSP"
```

Chaque fois que vous effectuez un commit (validation), vous créez un instantané de votre projet que vous pouvez restaurer par la suite en cas de problème. Pour plus d'informations sur les commandes et les processus Git, consultez la [documentation Git](#).

## Configuration d'un référentiel distant

Que se passe-t-il en cas de plantage de votre disque dur ou si vous souhaitez travailler sur votre projet à partir d'un autre ordinateur ? Pour sauvegarder votre code source en ligne et y accéder depuis n'importe quel ordinateur, configurez un référentiel distant dans lequel vous pouvez transférer vos commits.

AWS CodeCommit vous permet de créer un référentiel privé dans le cloud AWS. CodeCommit est gratuit dans l'[offre gratuite AWS](#) pour un maximum de cinq utilisateurs AWS Identity and Access Management (IAM) de votre compte. Pour obtenir des informations sur la tarification, veuillez consulter la [tarification relative à AWS CodeCommit](#).

Pour obtenir des instructions sur la configuration, veuillez consulter le [Guide de l'utilisateur AWSCodeCommit](#).

Pour stocker votre code de projet en ligne, vous pouvez également utiliser GitHub, une autre solution très appréciée. Elle vous permet de créer gratuitement un référentiel en ligne public et elle prend également en charge les référentiels privés sur la base d'un forfait mensuel. Pour vous inscrire à GitHub, accédez à [github.com](#).

Une fois que vous avez créé un référentiel distant pour votre projet, associez-le à votre référentiel local via `git remote add` :

```
~/workspace/my-first-app$ git remote add origin ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/my-repo
```

## Installation de l'interface de ligne de commande EB

Utilisez l'[interface de ligne de commande EB \(p. 1027\)](#) pour gérer vos environnements Elastic Beanstalk et surveiller leur état via la ligne de commande. Pour obtenir des instructions d'installation, consultez [Installation de l'interface de ligne de commande EB \(p. 1028\)](#).

Par défaut, l'interface de ligne de commande EB regroupe tous les éléments dans votre dossier de projet et les télécharge dans Elastic Beanstalk sous la forme d'un bundle de fichiers source. Si vous utilisez conjointement Git et l'interface de ligne de commande EB, vous pouvez éviter que les fichiers de classes générés ne soient validés dans la source grâce à `.gitignore` et éviter que les fichiers sources ne soient déployés grâce à `.ebignore`.

Vous pouvez également [configurer l'interface de ligne de commande EB pour déployer un artefact de build \(p. 1039\)](#) (un fichier WAR ou ZIP) au lieu du contenu de votre dossier de projet.

## Installation de l'AWS CLI

L'AWS Command Line Interface (AWS CLI) est un client unifié pour les services AWS qui fournit des commandes pour toutes les opérations d'API publiques. Ces commandes étant d'un niveau inférieur à celles fournies par l'interface de ligne de commande EB, le nombre de commandes requises pour effectuer une opération avec l'AWS CLI est généralement plus élevé. En revanche, l'AWS Command Line Interface vous permet de travailler avec n'importe quelle application ou environnement exécuté dans votre compte, sans avoir à configurer un référentiel sur votre ordinateur local. Utilisez l'AWS CLI pour créer des scripts qui simplifient ou automatisent les tâches opérationnelles.

Pour plus d'informations sur les services pris en charge et pour télécharger le AWS Command Line Interface, reportez-vous à la section [AWS Command Line Interface](#).

# Utilisation de l'interface de ligne de commande Elastic Beanstalk (EB)

Cette interface de ligne de commande pour AWS Elastic Beanstalk fournit des commandes interactives qui simplifient la création, la mise à jour et la surveillance des environnements à partir d'un référentiel local. Utilisez l'interface de ligne de commande EB dans le cadre de cycles de tests et de développement quotidiens, à la place de la console Elastic Beanstalk.

## Note

La version actuelle de l'interface de ligne de commande EB inclut un ensemble de commandes de base différent de celui des versions antérieures à la version 3.0. Si vous utilisez une version antérieure, veuillez consulter [Migration vers l'interface de ligne de commande Elastic Beanstalk 3 et CodeCommit \(p. 1125\)](#) pour plus d'informations sur la migration.

Après avoir [installé l'interface de ligne de commande EB \(p. 1028\)](#) et configuré un répertoire de projet, vous pouvez créer des environnements via une seule commande :

```
~/my-app$ eb create my-env
```

Le code source de l'interface de ligne de commande EB est un projet open-source. Il réside dans le référentiel GitHub [aws/aws-elastic-beanstalk-cli](#). Vous pouvez participer en signalant les problèmes, en faisant des suggestions et en soumettant des demandes d'extraction. Nous apprécions vos contributions ! Pour un environnement où vous avez l'intention d'utiliser uniquement l'interface de ligne de commande EB en l'état, nous vous recommandons de l'installer à l'aide de l'un des scripts d'installation de l'interface de ligne de commande EB, comme décrit dans [the section called “Installation de l'interface de ligne de commande EB à l'aide de scripts” \(p. 1028\)](#).

Auparavant, Elastic Beanstalk prenait en charge une interface de ligne de commande distincte qui fournissait un accès direct aux opérations de l'API, appelée [interface en ligne de commande de l'API Elastic Beanstalk \(p. 1125\)](#). Elle a été remplacée par l'[AWS CLI \(p. 1026\)](#), qui offre la même fonctionnalité, mais pour les API des tous les services AWS.

Avec l'[AWS CLI](#), vous avez un accès direct à l'API Elastic Beanstalk. L'[AWS CLI](#) est parfaitement adaptée pour les scripts, mais elle est plus compliquée à utiliser à partir de la ligne de commande en raison du nombre de commandes que vous devez exécuter et du nombre de paramètres pour chaque commande. Par exemple, la création d'un environnement nécessite la série de commandes suivante :

```
~$ aws elasticbeanstalk check-dns-availability --cname-prefix my-cname
~$ aws elasticbeanstalk create-application-version --application-name my-application --version-label v1 --source-bundle S3Bucket=DOC-EXAMPLE-BUCKET,S3Key=php-proxy-sample.zip
~$ aws elasticbeanstalk create-environment --cname-prefix my-cname --application-name my-app --version-label v1 --environment-name my-env --solution-stack-name "64bit Amazon Linux 2015.03 v2.0.0 running Ruby 2.2 (Passenger Standalone)"
```

Pour plus d'informations sur l'installation de l'interface de ligne de commande EB, la configuration d'un référentiel et l'utilisation des environnements, consultez les rubriques suivantes.

## Rubriques

- [Installation de l'interface de ligne de commande EB \(p. 1028\)](#)
- [Configuration de l'interface de ligne de commande EB \(p. 1036\)](#)
- [Gestion des environnements Elastic Beanstalk avec l'interface de ligne de commande EB \(p. 1040\)](#)
- [Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild \(p. 1044\)](#)
- [Utilisation de l'interface de ligne de commande EB avec Git \(p. 1046\)](#)

- Utilisation de l'interface de ligne de commande EB avec AWS CodeCommit (p. 1048)
- Utilisation de l'interface de ligne de commande EB pour surveiller l'intégrité de l'environnement (p. 1053)
- Gestion de plusieurs environnements Elastic Beanstalk en tant que groupe avec l'interface de ligne de commande EB (p. 1058)
- Résolution de problèmes avec l'interface de ligne de commande EB (p. 1060)
- Guide de référence des commandes de l'interface de ligne de commande (CLI) EB (p. 1062)
- Interface de ligne de commande EB 2.6 (retirée) (p. 1124)
- Interface de ligne de commande de l'API Elastic Beanstalk (mise hors service) (p. 1125)

## Installation de l'interface de ligne de commande EB

L'interface de ligne de commande AWS Elastic Beanstalk (EB CLI) est un client de ligne de commande qui vous permet de créer, configurer et gérer des environnements Elastic Beanstalk. Pour plus d'informations sur l'interface de ligne de commande EB, veuillez consulter [Interface de ligne de commande EB \(p. 1027\)](#).

### Rubriques

- [Installation de l'interface de ligne de commande EB à l'aide de scripts \(p. 1028\)](#)
- [Installation manuelle de l'interface de ligne de commande EB \(p. 1028\)](#)

## Installation de l'interface de ligne de commande EB à l'aide de scripts

Le moyen le plus simple et recommandé d'installer l'interface de ligne de commande EB consiste à utiliser les [scripts de configuration de l'interface de ligne de commande EB](#) disponibles sur GitHub. Utilisez les scripts pour installer l'interface de ligne de commande EB sur Linux, macOS ou Windows. Les scripts installent l'interface de ligne de commande EB et ses dépendances, y compris Python et pip. Les scripts créent également un environnement virtuel pour l'interface de ligne de commande EB. Pour obtenir des instructions d'installation, veuillez consulter le référentiel [aws/aws-elastic-beanstalk-cli-setup](#) sur GitHub.

## Installation manuelle de l'interface de ligne de commande EB

Pour installer l'interface de ligne de commande EB, nous vous recommandons d'utiliser les [scripts d'installation de l'interface de ligne de commande EB](#). Si les scripts d'installation ne sont pas compatibles avec votre environnement de développement, installez manuellement l'interface de ligne de commande EB.

La méthode de distribution principale pour l'interface de ligne de commande EB sous Linux, Windows et macOS est pip. Il s'agit d'un gestionnaire de package pour Python qui fournit un moyen simple d'installer, de mettre à niveau et de supprimer des packages Python et leurs dépendances. Pour macOS, vous pouvez également obtenir la dernière version de l'interface de ligne de commande EB avec Homebrew.

## Notes de compatibilité

L'interface de ligne de commande EB est développée en Python et nécessite Python version 2.7, 3.4 ou ultérieure.

### Note

Depuis la version 2015.03, Amazon Linux est fourni avec Python 2.7 et pip.

Nous vous recommandons d'utiliser les [scripts d'installation de l'interface de ligne de commande EB](#) pour installer l'interface de ligne de commande EB et ses dépendances. Si vous devez installer manuellement l'interface de ligne de commande EB, il peut être difficile de gérer les conflits de dépendance dans votre environnement de développement.

L'interface de ligne de commande EB et l'[AWS Command Line Interface \(AWS CLI\)](#) partagent une dépendance sur le package Python `botocore`. En raison d'une nouveauté dans `botocore`, les différentes versions de ces deux outils d'interface de ligne de commande dépendent des différentes versions de `botocore`.

Les dernières versions des deux interfaces de ligne de commande sont compatibles. Si vous devez utiliser une version antérieure, consultez le tableau suivant pour connaître la version compatible à utiliser.

Version de l'interface de ligne de commande EB	Version compatible de l'AWS CLI
3.14.5 ou version antérieure	1.16.9 ou version antérieure
3.14.6 ou version ultérieure	1.16.11 ou version ultérieure

## Installer l'interface de ligne de commande EB

Si vous avez déjà `pip` et une version prise en charge de Python, utilisez la procédure suivante pour installer l'interface de ligne de commande EB.

Si vous n'avez ni Python ni `pip`, utilisez la procédure indiquée pour votre système d'exploitation :

- [Installation de Python, de pip et de l'interface de ligne de commande EB sous Linux \(p. 1031\)](#)
- [Installation de l'interface de ligne de commande EB sous macOS \(p. 1033\)](#)
- [Installation de Python, de pip et de l'interface de ligne de commande EB sous Windows \(p. 1034\)](#)

Pour installer l'interface de ligne de commande EB

1. Exécutez la commande suivante.

```
$ pip install awsebcli --upgrade --user
```

L'option `--upgrade` demande à `pip` de mettre à niveau toutes les exigences qui sont déjà installées. L'option `--user` demande à `pip` d'installer le programme dans un sous-répertoire de votre répertoire utilisateur pour éviter la modification des bibliothèques utilisées par votre système d'exploitation.

### Note

Si vous rencontrez des problèmes lorsque vous tentez d'installer l'interface de ligne de commande EB avec `pip`, vous pouvez [installer l'interface de ligne de commande EB dans un environnement virtuel \(p. 1035\)](#) pour isoler l'outil et ses dépendances, ou utiliser une version de Python autre que celle que vous utilisez normalement.

2. Ajoutez le chemin d'accès au fichier exécutable à votre variable `PATH` :

- Sous Linux et macOS :

Linux : `~/local/bin`

macOS : `~/Library/Python/3.7/bin`

Pour modifier votre variable `PATH` (Linux, Unix ou macOS) :

- a. Recherchez le script de profil de votre shell dans votre dossier utilisateur. Si vous n'êtes pas certain du shell utilisé, exéutez echo \$SHELL.

```
$ ls -a ~  
.. .bash_logout .bash_profile .bashrc Desktop Documents Downloads
```

- Bash : .bash\_profile, .profile ou .bash\_login.
  - Zsh : .zshrc
  - Tcsh : .tcshrc, .cshrc ou .login.
- b. Ajoutez une commande d'exportation à votre script de profil. L'exemple suivant ajoute le chemin d'accès représenté par **LOCAL\_PATH** à la variable PATH actuelle.

```
export PATH=$LOCAL_PATH:$PATH
```

- c. Chargez le script de profil décrit à la première étape dans votre session actuelle. L'exemple suivant charge le script de profil représenté par **PROFILE\_SCRIPT**.

```
$ source ~/PROFILE_SCRIPT
```

- Sous Windows :

Python 3.7 : %USERPROFILE%\AppData\Roaming\Python\Python37\Scripts

Versions antérieures de Python : %USERPROFILE%\AppData\Roaming\Python\Scripts

Pour modifier votre variable PATH (Windows) :

- a. Appuyez sur la touche Windows et entrez **environment variables**.
- b. Choisissez Modifier les variables d'environnement pour votre compte.
- c. Choisissez PATH, puis Modifier.
- d. Ajoutez des chemins d'accès dans le champ Valeur de la variable, en les séparant par des points virgules. Par exemple: **C:\item1\path;C:\item2\path**
- e. Choisissez OK deux fois pour appliquer les nouveaux paramètres.
- f. Fermez toutes les fenêtres d'invite de commande en cours d'exécution, puis rouvrez une fenêtre d'invite de commande.

3. Vérifiez que l'interface de ligne de commande EB est installée correctement en exécutant eb --version.

```
$ eb --version  
EB CLI 3.14.8 (Python 3.7)
```

L'interface de ligne de commande Elastic Beanstalk est mise à jour régulièrement pour ajouter une fonctionnalité prenant en charge les dernières fonctions Elastic Beanstalk. Pour effectuez une mise à jour vers la dernière version de l'interface de ligne de commande EB, exécutez à nouveau la commande d'installation.

```
$ pip install awsebcli --upgrade --user
```

Si vous devez désinstaller l'interface de ligne de commande EB, utilisez pip uninstall.

```
$ pip uninstall awsebcli
```

## Installation de Python, de pip et de l'interface de ligne de commande EB sous Linux

L'interface de ligne de commande EB nécessite Python 2.7, 3.4 ou version ultérieure. Si votre distribution n'est pas fournie avec Python, ou vous a été fournie avec une version antérieure, installez Python avant d'installer `pip` et l'interface de ligne de commande EB.

Pour installer Python 3.7 sous Linux

1. Déterminez si Python est déjà installé.

```
$ python --version
```

### Note

Si votre distribution Linux est fournie avec Python, vous devrez peut-être installer le package de développement Python afin d'obtenir les en-têtes et les bibliothèques nécessaires à la compilation des extensions et à l'installation de l'interface de ligne de commande EB. Utilisez votre gestionnaire de package pour installer le package de développement (généralement nommé `python-dev` ou `python-devel`).

2. Si Python 2.7 ou version ultérieure n'est pas installé, installez Python 3.7 à l'aide du gestionnaire de package de votre distribution. Le nom de la commande et du package varie :

- Sur les dérivés Debian, comme Ubuntu, utilisez APT :

```
$ sudo apt-get install python3.7
```

- Sur Red Hat et dérivés, utilisez yum.

```
$ sudo yum install python37
```

- Sur SUSE et dérivés, utilisez zypper.

```
$ sudo zypper install python3-3.7
```

3. Ouvrez une invite de commande ou un shell, et exécutez la commande suivante pour vérifier que Python est installé correctement.

```
$ python3 --version
Python 3.7.3
```

Installez `pip` en utilisant le script fourni par Python Packaging Authority, puis installez l'interface de ligne de commande EB.

Pour installer `pip` et l'interface de ligne de commande EB

1. Téléchargez le script d'installation depuis [pypa.io](https://bootstrap.pypa.io/get-pip.py) :

```
$ curl -O https://bootstrap.pypa.io/get-pip.py
```

Le script télécharge et installe la dernière version de `pip` et un autre package obligatoire nommé `setuptools`.

2. Exécutez le script avec Python.

```
$ python3 get-pip.py --user
Collecting pip
  Downloading pip-8.1.2-py2.py3-none-any.whl (1.2MB)
Collecting setuptools
  Downloading setuptools-26.1.1-py2.py3-none-any.whl (464kB)
Collecting wheel
  Downloading wheel-0.29.0-py2.py3-none-any.whl (66kB)
Installing collected packages: pip, setuptools, wheel
Successfully installed pip setuptools wheel
```

L'appel de la version 3 de Python directement en utilisant la commande `python3` à la place de `python` garantit que `pip` est installé au bon emplacement, même si une version système plus ancienne de Python est présente sur votre système.

3. Ajoutez le chemin d'accès au fichier exécutable à votre variable `PATH` : `~/local/bin`

Pour modifier votre variable `PATH` (Linux, Unix ou macOS) :

- a. Recherchez le script de profil de votre shell dans votre dossier utilisateur. Si vous n'êtes pas certain du shell utilisé, exécutez `echo $SHELL`.

```
$ ls -a ~
. . . .bash_logout .bash_profile .bashrc Desktop Documents Downloads
```

- Bash : `.bash_profile`, `.profile` ou `.bash_login`.
- Zsh : `.zshrc`
- Tcsh : `.tcshrc`, `.cshrc` ou `.login`.

- b. Ajoutez une commande d'exportation à votre script de profil. L'exemple suivant ajoute le chemin d'accès représenté par `LOCAL_PATH` à la variable `PATH` actuelle.

```
export PATH=$LOCAL_PATH:$PATH
```

- c. Chargez le script de profil décrit à la première étape dans votre session actuelle. L'exemple suivant charge le script de profil représenté par `PROFILE_SCRIPT`.

```
$ source ~/PROFILE_SCRIPT
```

4. Vérifiez que `pip` est installé correctement.

```
$ pip --version
pip 8.1.2 from ~/local/lib/python3.7/site-packages (python 3.7)
```

5. Utilisez `pip` pour installer l'interface de ligne de commande EB.

```
$ pip install awsebcli --upgrade --user
```

6. Vérifiez que l'interface de ligne de commande EB est installée correctement.

```
$ eb --version
EB CLI 3.14.8 (Python 3.7)
```

Pour effectuer une mise à niveau vers la dernière version, exécutez à nouveau la commande d'installation.

```
$ pip install awsebcli --upgrade --user
```

## Installation de l'interface de ligne de commande EB sous macOS

Si vous utilisez le gestionnaire de package Homebrew, vous pouvez installer l'interface de ligne de commande EB à l'aide de la commande `brew`. Vous pouvez également installer Python et `pip`, puis utiliser `pip` pour installer l'interface de ligne de commande EB.

### Installation de l'interface de ligne de commande EB avec Homebrew

La dernière version de l'interface de ligne de commande EB est généralement disponible sur Homebrew quelques jours après qu'elle s'est affichée dans `pip`.

Pour installer l'interface de ligne de commande EB avec **Homebrew**

1. Vérifiez que vous utilisez la dernière version de Homebrew.

```
$ brew update
```

2. Exécutez `brew install awsebcli`.

```
$ brew install awsebcli
```

3. Vérifiez que l'interface de ligne de commande EB est installée correctement.

```
$ eb --version
EB CLI 3.14.8 (Python 3.7)
```

## Installation de Python, de pip et de l'interface de ligne de commande EB sous macOS

Vous pouvez installer la dernière version de Python et de `pip`, puis les utiliser pour installer l'interface de ligne de commande EB.

Pour installer l'interface de ligne de commande EB sous macOS

1. Téléchargez et installez Python à partir de la [page de téléchargement de Python.org](#). Nous utilisons la version 3.7 à titre d'illustration.

Note

L'interface de ligne de commande EB nécessite Python 2 version 2.7 ou Python 3 versions 3.4 à 3.7.

2. Installez `pip` en utilisant le script fourni par Python Packaging Authority.

```
$ curl -O https://bootstrap.pypa.io/get-pip.py
$ python3 get-pip.py --user
```

3. Utilisez `pip` pour installer l'interface de ligne de commande EB.

```
$ pip3 install awsebcli --upgrade --user
```

4. Ajoutez le chemin d'accès au fichier exécutable à votre variable `PATH` : `~/Library/Python/3.7/bin`

Pour modifier votre variable `PATH` (Linux, Unix ou macOS) :

- a. Recherchez le script de profil de votre shell dans votre dossier utilisateur. Si vous n'êtes pas certain du shell utilisé, exécutez `echo $SHELL`.

```
$ ls -a ~  
. .. .bash_logout .bash_profile .bashrc Desktop Documents Downloads
```

- Bash : `.bash_profile`, `.profile` ou `.bash_login`.
  - Zsh : `.zshrc`
  - Tcsh : `.tcshrc`, `.cshrc` ou `.login`.
- b. Ajoutez une commande d'exportation à votre script de profil. L'exemple suivant ajoute le chemin d'accès représenté par `LOCAL_PATH` à la variable `PATH` actuelle.

```
export PATH=$LOCAL_PATH:$PATH
```

- c. Chargez le script de profil décrit à la première étape dans votre session actuelle. L'exemple suivant charge le script de profil représenté par `PROFILE_SCRIPT`.

```
$ source ~/PROFILE_SCRIPT
```

5. Vérifiez que l'interface de ligne de commande EB est installée correctement.

```
$ eb --version  
EB CLI 3.14.8 (Python 3.7)
```

Pour effectuer une mise à niveau vers la dernière version, exécutez à nouveau la commande d'installation.

```
$ pip3 install awsebcli --upgrade --user
```

## Installation de Python, de pip et de l'interface de ligne de commande EB sous Windows

La Python Software Foundation fournit des programmes d'installation pour Windows qui incluent `pip`.

Pour installer Python 3.7 et `pip` (Windows)

1. Téléchargez le programme d'installation du fichier exécutable Windows x86-64 de Python 3.7 à partir de la [page de téléchargements de Python.org](#).
2. Exédez le programme d'installation.
3. Choisissez Add Python 3.7 to PATH (Ajouter Python 3.7 à PATH).
4. Choisissez Install Now (Installer maintenant).

Le programme d'installation installe Python dans votre dossier d'utilisateur et ajoute ses répertoires exécutables à votre chemin d'utilisateur.

Pour installer l'AWS CLI avec `pip` (Windows)

1. Dans le menu Démarrer, ouvrez une fenêtre d'invite de commande.
2. Vérifiez que Python et `pip` sont tous deux installés correctement en utilisant les commandes suivantes.

```
C:\Windows\System32> python --version  
Python 3.7.3  
C:\Windows\System32> pip --version
```

```
pip 9.0.1 from c:\users\myname\appdata\local\programs\python\python37\lib\site-packages  
(python 3.7)
```

3. Installez l'interface de ligne de commande EB en utilisant pip.

```
C:\Windows\System32> pip install awsebcli --upgrade --user
```

4. Ajoutez le chemin d'accès au fichier exécutable (%USERPROFILE%\AppData\roaming\Python\Python37\scripts) à votre variable d'environnement PATH. L'emplacement peut varier selon que vous installez Python pour un utilisateur ou pour tous les utilisateurs.

Pour modifier votre variable PATH (Windows) :

- a. Appuyez sur la touche Windows et entrez **environment variables**.
  - b. Choisissez Modifier les variables d'environnement pour votre compte.
  - c. Choisissez PATH, puis Modifier.
  - d. Ajoutez des chemins d'accès dans le champ Valeur de la variable, en les séparant par des points virgules. Par exemple: **C:\item1\path;C:\item2\path**
  - e. Choisissez OK deux fois pour appliquer les nouveaux paramètres.
  - f. Fermez toutes les fenêtres d'invite de commande en cours d'exécution, puis rouvrez une fenêtre d'invite de commande.
5. Redémarrez un nouveau shell de commande pour que la nouvelle variable PATH prennent effet.
  6. Vérifiez que l'interface de ligne de commande EB est installée correctement.

```
C:\Windows\System32> eb --version  
EB CLI 3.14.8 (Python 3.7)
```

Pour effectuer une mise à niveau vers la dernière version, exécutez à nouveau la commande d'installation.

```
C:\Windows\System32> pip install awsebcli --upgrade --user
```

## Installation de l'interface de ligne de commande EB dans un environnement virtuel

Pour éviter les conflits entre exigences des versions et autres packages pip, vous pouvez installer l'interface de ligne de commande EB dans un environnement virtuel.

Pour installer l'interface de ligne de commande EB dans un environnement virtuel

1. Installez virtualenv avec pip.

```
$ pip install --user virtualenv
```

2. Créez un environnement virtuel.

```
$ virtualenv ~/eb-ve
```

L'option -p vous permet d'utiliser un exécutable Python autre que l'exécutable par défaut.

```
$ virtualenv -p /usr/bin/python3.7 ~/eb-ve
```

3. Activez l'environnement virtuel.

Linux, Unix ou macOS

```
$ source ~/eb-ve/bin/activate
```

Windows

```
$ %USERPROFILE%\eb-ve\Scripts\activate
```

4. Installation de l'interface de ligne de commande EB.

```
(eb-ve)$ pip install awsebcli --upgrade
```

5. Vérifiez que l'interface de ligne de commande EB est installée correctement.

```
$ eb --version
EB CLI 3.14.8 (Python 3.7)
```

La commande deactivate vous permet de quitter l'environnement virtuel. Exécutez à nouveau la commande d'activation chaque fois que vous démarrez une nouvelle session.

Pour effectuer une mise à niveau vers la dernière version, exécutez à nouveau la commande d'installation.

```
(eb-ve)$ pip install awsebcli --upgrade
```

## Configuration de l'interface de ligne de commande EB

Une fois [l'installation de l'interface de ligne de commande EB effectuée \(p. 1028\)](#), vous êtes prêt à configurer votre répertoire de projet et l'interface de ligne de commande EB en exécutant eb init.

L'exemple suivant illustre les étapes de configuration à suivre lorsque vous exécutez eb init pour la première fois dans un dossier de projet nommé eb.

Pour initialiser un projet d'interface de ligne de commande EB

1. Tout d'abord, l'interface de ligne de commande EB vous invite à sélectionner une région. Saisissez le numéro qui correspond à la région que vous souhaitez utiliser, puis appuyez sur Entrée.

```
~/eb $ eb init
Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : Europe (Ireland)
5) eu-central-1 : Europe (Frankfurt)
6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
...
(default is 3): 3
```

2. Ensuite, fournissez votre clé d'accès et votre clé secrète pour que l'interface de ligne de commande EB puisse gérer les ressources à votre place. Les clés d'accès sont créées dans la console AWS

Identity and Access Management. Si vous n'avez pas de clés, veuillez consulter [Comment obtenir les informations d'identification de sécurité ?](#) dans la Référence générale d'Amazon Web Services.

```
You have not yet set up your credentials or your credentials are incorrect.  
You must provide your credentials.  
(aws-access-id): AKIAJOUAASEXAMPLE  
(aws-secret-key): 5ZR1rtTM4ciIAvdaEXAMPLEDtm+PiPSzpoK
```

3. Dans Elastic Beanstalk, une application est une ressource qui contient un ensemble de versions d'application (source), d'environnements et de configurations enregistrées, qui sont associés à une application web unique. Chaque fois que vous déployez votre code source dans Elastic Beanstalk via l'interface de ligne de commande EB, une nouvelle version d'application est créée et ajoutée à la liste.

```
Select an application to use  
1) [ Create new Application ]  
(default is 1): 1
```

4. Le nom de l'application par défaut correspond au nom du dossier dans lequel vous exécutez eb init. Saisissez un nom décrivant votre projet.

```
Enter Application Name  
(default is "eb"): eb  
Application eb has been created.
```

5. Sélectionnez une plateforme qui correspond à l'infrastructure ou au langage dans lequel votre application web est développée. Si vous n'avez pas encore commencé à développer votre application, choisissez la plateforme qui vous intéresse. Vous saurez bientôt comment lancer un exemple d'application et vous pourrez modifier ce paramètre ultérieurement.

```
Select a platform.  
1) Node.js  
2) PHP  
3) Python  
4) Ruby  
5) Tomcat  
6) IIS  
7) Docker  
8) Multi-container Docker  
9) GlassFish  
10) Go  
11) Java  
(default is 1): 1
```

6. Choisissez yes (oui) pour attribuer une paire de clés SSH aux instances de votre environnement Elastic Beanstalk. Cela vous permet de vous connecter directement à des fins de dépannage.

```
Do you want to set up SSH for your instances?  
(y/n): y
```

7. Choisissez une paire de clés existante ou créez-en une. Pour utiliser eb init afin de créer une nouvelle paire de clés, l'outil ssh-keygen doit être installé sur votre ordinateur local et disponible à partir de la ligne de commande. L'interface de ligne de commande EB enregistre la nouvelle paire de clés avec Amazon EC2 et stocke la clé privée localement dans un dossier nommé .ssh, inclus dans votre répertoire utilisateur.

```
Select a keypair.  
1) [ Create new KeyPair ]  
(default is 1): 1
```

Votre interface de ligne de commande EB est maintenant configurée et prête à être utilisée. Veuillez consulter [Gestion des environnements Elastic Beanstalk avec l'interface de ligne de commande EB \(p. 1040\)](#) pour obtenir des instructions sur la création et l'utilisation d'un environnement Elastic Beanstalk.

#### Configuration avancée

- [Utilisation d'un fichier .ebignore pour ignorer des fichiers \(p. 1038\)](#)
- [Utilisation de profils nommés \(p. 1038\)](#)
- [Déploiement d'un artefact à la place du dossier de projet \(p. 1039\)](#)
- [Configuration des paramètres et des priorités \(p. 1039\)](#)
- [Métadonnées de l'instance \(p. 1039\)](#)

## Utilisation d'un fichier .ebignore pour ignorer des fichiers

Vous pouvez donner instruction à l'interface de ligne de commande EB d'ignorer certains fichiers du répertoire de votre projet en ajoutant le fichier `.ebignore` au répertoire. Ce fichier fonctionne comme un fichier `.gitignore`. Lorsque vous déployez le répertoire de votre projet dans Elastic Beanstalk et que vous créez une nouvelle version d'application, l'interface de ligne de commande EB n'inclut pas les fichiers spécifiés par `.ebignore` dans le bundle source qu'elle crée.

Si `.ebignore` n'est pas présent, mais qu'il existe `.gitignore`, l'interface de ligne de commande EB ignore les fichiers spécifiés dans `.gitignore`. Si `.ebignore` est présent, l'interface de ligne de commande EB ne lit pas `.gitignore`.

Quand `.ebignore` est présent, l'interface de ligne de commande EB n'utilise pas de commandes Git pour créer votre bundle de fichiers source. Autrement dit, l'interface de ligne de commande EB ignore les fichiers spécifiés dans `.ebignore` et inclut tous les autres fichiers. Plus particulièrement, elle comprend des fichiers source non validés.

#### Note

Dans Windows, si vous ajoutez `.ebignore`, l'interface de ligne de commande EB suit les liens symboliques et inclut le fichier lié lors de la création d'un bundle de fichiers source. Il s'agit d'un problème connu, qui sera corrigé lors d'une future mise à jour.

## Utilisation de profils nommés

Si vous stockez vos informations d'identification sous la forme d'un profil nommé dans un fichier `credentials` ou `config`, vous pouvez utiliser l'option [--profile \(p. 1123\)](#) pour spécifier explicitement un profil. Par exemple, la commande suivante crée une nouvelle application à l'aide du profil `user2`.

```
$ eb init --profile user2
```

Vous pouvez également modifier le profil par défaut en définissant la variable d'environnement `AWS_EB_PROFILE`. Lorsque cette variable est définie, l'interface de ligne de commande EB lit les informations d'identification à partir du profil spécifié au lieu de `default` ou `eb-cli`.

Linux, macOS ou Unix

```
$ export AWS_EB_PROFILE=user2
```

Windows

```
> set AWS_EB_PROFILE=user2
```

## Déploiement d'un artefact à la place du dossier de projet

Vous pouvez demander à l'interface de ligne de commande EB de déployer un fichier ZIP ou WAR que vous générez dans le cadre d'un processus de création distinct en ajoutant les lignes suivantes à `.elasticbeanstalk/config.yml` dans votre dossier de projet.

```
deploy:  
  artifact: path/to/buildartifact.zip
```

Si vous configurez l'interface de ligne de commande EB dans votre [référentiel Git \(p. 1046\)](#), et ne validez pas l'artefact dans la source, utilisez l'option `--staged` pour déployer la version la plus récente.

```
~/eb$ eb deploy --staged
```

## Configuration des paramètres et des priorités

L'interface de ligne de commande EB utilise une chaîne fournisseur pour rechercher les informations d'identification AWS dans un certain nombre d'emplacements différents, y compris les variables d'environnement système ou utilisateur et les fichiers de configuration locaux AWS.

L'interface de ligne de commande EB recherche les informations d'identification et les paramètres de configuration dans l'ordre suivant :

1. Options de ligne de commande – Spécifiez un profil nommé avec `--profile` pour remplacer les paramètres par défaut.
2. Variables d'environnement – `AWS_ACCESS_KEY_ID` et `AWS_SECRET_ACCESS_KEY`.
3. Fichier d'informations d'identification AWS – Situé dans `~/.aws/credentials` sous Linux et OS X, ou dans `C:\Users\USERNAME\.aws\credentials` sous Windows. Ce fichier peut contenir plusieurs profils nommés en plus d'un profil par défaut.
4. [Fichier de configuration de l'AWS CLI](#) – situé dans `~/.aws/config` sous Linux et OS X ou dans `C:\Users\USERNAME\.aws\config` sous Windows. Ce fichier peut contenir un profil par défaut, des [profils nommés](#), ainsi que des paramètres de configuration spécifiques à une AWS CLI pour chacun.
5. Fichier de configuration existant de l'interface de ligne de commande EB – Situé dans `~/.elasticbeanstalk/config` sous Linux et OS X ou dans `C:\Users\USERNAME\.elasticbeanstalk\config` sous Windows.
6. Informations d'identification du profil d'instance – Ces informations d'identification peuvent être utilisées sur les instances Amazon EC2 avec un rôle d'instance attribué et elles sont fournies via le service de métadonnées Amazon EC2. Le [profil d'instance \(p. 22\)](#) doit avoir l'autorisation d'utiliser Elastic Beanstalk.

Si le fichier d'informations d'identification contient un profil nommé « eb-cli », l'interface de ligne de commande EB le priviliege par rapport au profil par défaut. Si aucun profil n'est trouvé, ou si le profil trouvé n'est pas autorisé à utiliser –, l'interface de ligne de commande EB vous invite à saisir les clés.

## Métadonnées de l'instance

Pour utiliser l'interface de ligne de commande EB à partir d'une instance Amazon EC2, créez un rôle ayant accès aux ressources nécessaires et attribuez ce rôle à l'instance lorsqu'elle est lancée. Lancez l'instance et installez l'interface de ligne de commande EB avec `pip`.

```
~$ sudo pip install awsebcli
```

pip est pré-installé sur Amazon Linux.

L'interface de ligne de commande EB lit les informations d'identification à partir des métadonnées de l'instance. Pour de plus amples informations, veuillez consulter [Octroi d'autorisations d'accès aux ressources AWS pour les applications qui s'exécutent sur des instances Amazon EC2](#) dans le Guide de l'utilisateur IAM.

## Gestion des environnements Elastic Beanstalk avec l'interface de ligne de commande EB

Une fois que vous avez [installé l'interface de ligne de commande EB \(p. 1028\)](#) et [configuré le répertoire de votre projet \(p. 1036\)](#), vous pouvez créer un environnement Elastic Beanstalk via l'interface de ligne de commande EB, déployer les mises à jour de la source et de la configuration, et extraire les journaux et les événements.

### Note

Créer des environnements avec l'interface de ligne de commande EB nécessite un [rôle de service \(p. 21\)](#). Vous pouvez créer un rôle de service en créant un environnement dans la console Elastic Beanstalk. Si vous n'avez pas de rôle de service, l'interface de ligne de commande EB essaie d'en créer un lorsque vous exécutez `eb create`.

L'interface de ligne de commande EB renvoie un code de sortie zéro (0) pour toutes les commandes réussies, et un code de sortie autre que zéro lorsqu'il rencontre une erreur.

Les exemples suivants utilisent un dossier de projet vide nommé eb, qui a été initialisé avec l'interface de ligne de commande EB pour être utilisé avec un exemple d'application Docker.

### Commandes de base

- [Eb create \(p. 1040\)](#)
- [Eb status \(p. 1041\)](#)
- [Eb health \(p. 1041\)](#)
- [Eb events \(p. 1042\)](#)
- [Eb logs \(p. 1042\)](#)
- [Eb open \(p. 1042\)](#)
- [Eb deploy \(p. 1043\)](#)
- [Eb config \(p. 1043\)](#)
- [Eb terminate \(p. 1044\)](#)

## Eb create

Pour créer votre premier environnement, exécutez [eb create \(p. 1078\)](#) et suivez les instructions. Si le répertoire de votre projet comporte du code source, l'interface de ligne de commande EB l'intègre dans un bundle et le déploie dans votre environnement. Sinon, un exemple d'application est utilisé.

```
~/eb$ eb create
Enter Environment Name
```

```
(default is eb-dev): eb-dev
Enter DNS CNAME prefix
(default is eb-dev): eb-dev
WARNING: The current directory does not contain any source code. Elastic Beanstalk is
         launching the sample application instead.
Environment details for: elasticBeanstalkExa-env
  Application name: elastic-beanstalk-example
  Region: us-west-2
  Deployed Version: Sample Application
  Environment ID: e-j3pmc8tscn
  Platform: 64bit Amazon Linux 2015.03 v1.4.3 running Docker 1.6.2
  Tier: WebServer-Standard
  CNAME: eb-dev.elasticbeanstalk.com
  Updated: 2015-06-27 01:02:24.813000+00:00
  Printing Status:
INFO: createEnvironment is starting.
-- Events -- (safe to Ctrl+C) Use "eb abort" to cancel the command.
```

Plusieurs minutes peuvent s'écouler avant que votre environnement soit prêt. Pour revenir à la ligne de commande pendant la création de l'environnement, appuyez sur Ctrl+C.

## Eb status

Exécutez eb status pour afficher l'état actuel de votre environnement. Lorsque l'état correspond à ready, cela signifie que l'exemple d'application est disponible à l'adresse elasticbeanstalk.com et que l'environnement est prêt à être mis à jour.

```
~/eb$ eb status
Environment details for: elasticBeanstalkExa-env
  Application name: elastic-beanstalk-example
  Region: us-west-2
  Deployed Version: Sample Application
  Environment ID: e-gbzqc3jcr
  Platform: 64bit Amazon Linux 2015.03 v1.4.3 running Docker 1.6.2
  Tier: WebServer-Standard
  CNAME: elasticbeanstalkexa-env.elasticbeanstalk.com
  Updated: 2015-06-30 01:47:45.589000+00:00
  Status: Ready
  Health: Green
```

## Eb health

Utilisez la commande eb health pour afficher les [informations relatives à l'état \(p. 837\)](#) des instances de votre environnement ainsi que l'état de votre environnement global. Utilisez l'option --refresh pour afficher les informations relatives à l'état dans une vue interactive mise à jour toutes les 10 secondes.

```
~/eb$ eb health
api                                     Ok          2016-09-15 18:39:04
WebServer
  total      ok     warning   degraded   severe     info    pending   unknown
  3          3        0          0          0          0        0          0
  instance-id      status      cause
  Overall          Ok
  i-0ef05ec54918bf567  Ok
  i-001880c1187493460  Ok
  i-04703409d90d7c353  Ok

  instance-id      r/sec    %2xx    %3xx    %4xx    %5xx    p99      p90      p75
  p50            p10
```

Overall	8.6	100.0	0.0	0.0	0.0	0.083*	0.065	0.053
0.040	0.019							
i-0ef05ec54918bf567	2.9	29	0	0	0	0.069*	0.066	0.057
0.050	0.023							
i-001880c1187493460	2.9	29	0	0	0	0.087*	0.069	0.056
0.050	0.034							
i-04703409d90d7c353	2.8	28	0	0	0	0.051*	0.027	0.024
0.021	0.015							
instance-id	type	az	running	load 1	load 5	user%	nice%	
system%	idle% iowait%							
i-0ef05ec54918bf567	t2.micro	1c	23 mins	0.19	0.05	3.0	0.0	
0.3	96.7	0.0						
i-001880c1187493460	t2.micro	1a	23 mins	0.0	0.0	3.2	0.0	
0.3	96.5	0.0						
i-04703409d90d7c353	t2.micro	1b	1 day	0.0	0.0	3.6	0.0	
0.2	96.2	0.0						
instance-id	status	id	version		ago			
deployments								
i-0ef05ec54918bf567	Deployed	28	app-bc1b-160915_181041	20 mins				
i-001880c1187493460	Deployed	28	app-bc1b-160915_181041	20 mins				
i-04703409d90d7c353	Deployed	28	app-bc1b-160915_181041	27 mins				

## Eb events

Utilisez eb events pour afficher la liste des événements produits par Elastic Beanstalk.

```
~/eb$ eb events
2015-06-29 23:21:09    INFO    createEnvironment is starting.
2015-06-29 23:21:10    INFO    Using elasticbeanstalk-us-east-2-EXAMPLE as Amazon S3
storage bucket for environment data.
2015-06-29 23:21:23    INFO    Created load balancer named: awseb-e-g-AWSEBLoa-EXAMPLE
2015-06-29 23:21:42    INFO    Created security group named: awseb-e-gbzqc3jcra-stack-
AWSEBSecurityGroup-EXAMPLE
...
```

## Eb logs

Utilisez eb logs pour extraire les journaux d'une instance de votre environnement. Par défaut, eb logs extrait les journaux de la première instance lancée et les affiche dans la sortie standard. Vous pouvez spécifier un ID d'instance avec l'option --instance afin d'obtenir les journaux d'une instance spécifique.

L'option --all extrait les journaux de toutes les instances et les enregistre dans des sous-répertoires, sous .elasticbeanstalk/logs.

```
~/eb$ eb logs --all
Retrieving logs...
Logs were saved to /home/local/ANT/mwunderl/ebcli/environments/test/.elasticbeanstalk/
logs/150630_201410
Updated symlink at /home/local/ANT/mwunderl/ebcli/environments/test/.elasticbeanstalk/logs/
latest
```

## Eb open

Pour ouvrir le site Web de votre environnement dans un navigateur, utilisez eb open :

```
~/eb$ eb open
```

Dans un environnement en mode fenêtré, votre navigateur par défaut s'ouvre dans une nouvelle fenêtre.  
Dans un environnement en mode terminal, un navigateur en ligne de commande (par exemple, w3m) est utilisé s'il est disponible.

## Eb deploy

Une fois que l'environnement est opérationnel, vous pouvez le mettre à jour à l'aide de la commande eb deploy.

Cette commande fonctionne mieux avec du code source à intégrer dans un bundle et à déployer. Pour cet exemple, nous avons donc créé un Dockerfile dans le répertoire du projet, avec le contenu suivant :

~/eb/Dockerfile

```
FROM ubuntu:12.04

RUN apt-get update
RUN apt-get install -y nginx zip curl

RUN echo "daemon off;" >> /etc/nginx/nginx.conf
RUN curl -o /usr/share/nginx/www/master.zip -L https://codeload.github.com/gabrielecirulli/2048/zip/master
RUN cd /usr/share/nginx/www/ && unzip master.zip && mv 2048-master/* . && rm -rf 2048-master master.zip

EXPOSE 80

CMD ["/usr/sbin/nginx", "-c", "/etc/nginx/nginx.conf"]
```

Ce Dockerfile déploie une image d'Ubuntu 12.04 et installe le jeu 2048. Exécutez eb deploy pour charger l'application dans votre environnement :

```
~/eb$ eb deploy
Creating application version archive "app-150630_014338".
Uploading elastic-beanstalk-example/app-150630_014338.zip to S3. This may take a while.
Upload Complete.
INFO: Environment update is starting.
-- Events -- (safe to Ctrl+C) Use "eb abort" to cancel the command.
```

Lorsque vous exécutez eb deploy, l'interface de ligne de commande EB crée un bundle avec le contenu de votre répertoire de projet et le déploie dans votre environnement.

### Note

Si vous avez initialisé un référentiel git dans votre dossier de projet, l'interface de ligne de commande EB déploie toujours la dernière version validée, même si des modifications sont en attente. Validez vos modifications avant d'exécuter eb deploy pour les déployer dans votre environnement.

## Eb config

Examinez les options de configuration disponibles pour votre environnement en cours d'exécution avec la commande eb config :

```
~/eb$ eb config
ApplicationName: elastic-beanstalk-example
DateUpdated: 2015-06-30 02:12:03+00:00
EnvironmentName: elasticBeanstalkExa-env
```

```
SolutionStackName: 64bit Amazon Linux 2015.03 v1.4.3 running Docker 1.6.2
settings:
  AWSEBAutoScalingScaleDownPolicy.aws:autoscaling:trigger:
    LowerBreachScaleIncrement: '-1'
  AWSEBAutoScalingScaleUpPolicy.aws:autoscaling:trigger:
    UpperBreachScaleIncrement: '1'
  AWSEBCloudwatchAlarmHigh.aws:autoscaling:trigger:
    UpperThreshold: '6000000'
...
```

Cette commande remplit une liste des options de configuration disponibles dans un éditeur de texte. La plupart des options affichées ont une valeur `null`. Elles ne sont pas définies par défaut, mais elles peuvent être modifiées pour mettre à jour les ressources dans votre environnement. Pour de plus amples informations sur ces options, veuillez consulter [Options de configuration \(p. 658\)](#).

## Eb terminate

Si vous avez terminé d'utiliser l'environnement pour l'instant, utilisez la commande `eb terminate` pour le suspendre.

```
~/eb$ eb terminate
The environment "eb-dev" and all associated instances will be terminated.
To confirm, type the environment name: eb-dev
INFO: terminateEnvironment is starting.
INFO: Deleted CloudWatch alarm named: awseb-e-jc8t3pmSCN-stack-
AWSEBCloudwatchAlarmHigh-1XLMU7DNCBV6Y
INFO: Deleted CloudWatch alarm named: awseb-e-jc8t3pmSCN-stack-
AWSEBCloudwatchAlarmLow-8IVI04W2SCXS
INFO: Deleted Auto Scaling group policy named: arn:aws:autoscaling:us-
east-2:123456789012:scalingPolicy:1753d43e-ae87-4df6-
a405-11d31f4c8f97:autoScalingGroupName/awseb-e-jc8t3pmSCN-stack-
AWSEBAutoScalingGroup-90TTS2ZL4MXV:policyName/awseb-e-jc8t3pmSCN-stack-
AWSEBAutoScalingScaleUpPolicy-A070H1BMUQAJ
INFO: Deleted Auto Scaling group policy named: arn:aws:autoscaling:us-
east-2:123456789012:scalingPolicy:1fd24ea4-3d6f-4373-
affc-4912012092ba:autoScalingGroupName/awseb-e-jc8t3pmSCN-stack-
AWSEBAutoScalingGroup-90TTS2ZL4MXV:policyName/awseb-e-jc8t3pmSCN-stack-
AWSEBAutoScalingScaleDownPolicy-LSWFUMZ46H1V
INFO: Waiting for EC2 instances to terminate. This may take a few minutes.
-- Events -- (safe to Ctrl+C)
```

Pour obtenir la liste complète des commandes de l'interface de ligne de commande EB, veuillez consulter [Guide de référence des commandes de l'interface de ligne de commande \(CLI\) EB \(p. 1062\)](#).

## Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild

[AWS CodeBuild](#) compile votre code source, exécute des tests unitaires et produit des artefacts prêts à être déployés. Vous pouvez utiliser CodeBuild avec l'interface de ligne de commande EB pour automatiser la création de votre application à partir de son code source. La création de l'environnement et chaque déploiement ultérieur commencent par une étape de développement, suivie du déploiement de l'application qui en résulte.

### Note

Certaines régions n'offrent pas CodeBuild. L'intégration entre Elastic Beanstalk et CodeBuild ne fonctionne pas dans ces régions.

Pour de plus amples informations sur les services AWS proposés dans chaque région, veuillez consulter le [Tableau des régions](#).

## Création d'une application

Pour créer une application Elastic Beanstalk qui utilise CodeBuild

1. Incluez un fichier de spécification de génération CodeBuild, `buildspec.yml`, dans le dossier de votre application.
2. Ajoutez au fichier une entrée `eb_codebuild_settings` avec des options spécifiques à Elastic Beanstalk.
3. Exécutez [eb init \(p. 1093\)](#) dans le dossier.

Elastic Beanstalk étend le [format de fichier de spécification de construction CodeBuild](#) pour inclure les paramètres supplémentaires suivants :

```
eb_codebuild_settings:  
  CodeBuildServiceRole: role-name  
  ComputeType: size  
  Image: image  
  Timeout: minutes
```

### CodeBuildServiceRole

L'ARN ou le nom de la fonction du service AWS Identity and Access Management (IAM) que CodeBuild peut utiliser pour interagir avec les services AWS dépendants en votre nom. Cette valeur est obligatoire. Si vous l'omettez, toute commande eb deploy ou eb create suivante échoue.

Pour en savoir plus sur la création d'une fonction de service pour CodeBuild, veuillez consulter [Création d'un rôle de service CodeBuild](#) dans le Guide de l'utilisateur AWS CodeBuild.

### Note

Vous avez également besoin d'autorisations pour exécuter des actions dans CodeBuild lui-même. La stratégie utilisateur AdministratorAccessAWSBeanstalk gérée par Elastic Beanstalk inclut toutes les autorisations d'action CodeBuild requises. Si vous n'utilisez pas la stratégie gérée, veillez à autoriser les autorisations suivantes dans votre stratégie d'utilisateur.

```
"codebuild:CreateProject",  
 "codebuild:DeleteProject",  
 "codebuild:BatchGetBuilds",  
 "codebuild:StartBuild"
```

Pour plus d'informations, consultez [Gestion des stratégies utilisateur Elastic Beanstalk \(p. 946\)](#).

### ComputeType

La quantité de ressources utilisée par le conteneur Docker dans l'environnement de construction CodeBuild. Les valeurs valides sont BUILD\_GENERAL1\_SMALL, BUILD\_GENERAL1\_MEDIUM et BUILD\_GENERAL1\_LARGE.

### Image

Nom de l'image Docker Hub ou Amazon ECR utilisée par CodeBuild pour l'environnement de construction. Cette image Docker doit contenir tous les outils et bibliothèques d'exécution nécessaires

à la création de votre code et doit correspondre à la plateforme cible de votre application. CodeBuild gère et maintient un ensemble d'images spécifiquement destinées à être utilisées avec Elastic Beanstalk. Il est recommandé d'utiliser l'une d'elles. Pour de plus amples informations, veuillez consulter [Images Docker fournies par CodeBuild](#) dans le Guide de l'utilisateur AWS CodeBuild.

La valeur `Image` est facultative. Si vous l'omettez, la commande `eb init` tente de choisir une image qui correspondent le mieux à votre plateforme cible. De plus, si vous exécutez `eb init` en mode interactif et si le choix automatique d'image échoue, vous êtes invité à en choisir une. Au terme de l'initialisation, `eb init` écrit l'image choisie dans le fichier `buildspec.yml`.

#### Timeout

Durée d'exécution de la construction CodeBuild, en minutes, avant l'expiration. Ce champ est facultatif. Pour de plus amples informations sur les valeurs valides et la valeur par défaut, veuillez consulter [Créer un projet de build dans CodeBuild](#).

#### Note

Ce délai d'expiration contrôle la durée maximale d'une exécution CodeBuild et l'interface de ligne de commande EB la respecte également dans le cadre de sa première étape de création d'une version de l'application. Il est différent de la valeur que vous pouvez spécifier avec l'option `--timeout` des commandes [eb create \(p. 1078\)](#) ou [eb deploy \(p. 1089\)](#). Cette valeur contrôle la durée maximale pendant laquelle l'interface de ligne de commande EB attend la création ou la mise à jour de l'environnement.

## Génération et déploiement du code de votre application

Chaque fois que le code de votre application doit être déployé, l'interface de ligne de commande EB utilise CodeBuild pour exécuter une génération, puis déploie les artefacts de génération résultants dans votre environnement. Cela se produit lorsque vous créez un environnement Elastic Beanstalk pour votre application à l'aide de la commande [eb create \(p. 1078\)](#) et chaque fois que vous déployez par la suite des modifications du code dans l'environnement à l'aide de la commande [eb deploy \(p. 1089\)](#).

Si l'étape CodeBuild échoue, la création ou le déploiement de l'environnement ne démarre pas.

## Utilisation de l'interface de ligne de commande EB avec Git

L'interface de ligne de commande EB s'intègre dans Git. Cette section explique comment utiliser Git avec l'interface de ligne de commande EB.

Pour installer Git et initialiser votre référentiel Git

1. Téléchargez la version la plus récente de Git sur le site <http://git-scm.com>.
2. Initialisez votre référentiel Git en saisissant la chaîne suivante :

```
~/eb$ git init
```

L'interface de ligne de commande EB reconnaît désormais que votre application est configurée avec Git.

3. Si vous n'avez pas encore exécuté eb init, faites-le maintenant :

```
~/eb$ eb init
```

## Associer des environnements Elastic Beanstalk à des branches Git

Vous pouvez associer un environnement différent à chaque branche de votre code. Lorsque vous procédez à l'extraction d'une branche, les modifications sont déployées pour l'environnement associé. Vous pouvez par exemple saisir les informations suivantes pour associer votre environnement de production à votre branche principale, et un environnement de développement distinct à une branche de développement :

```
~/eb$ git checkout mainline
~/eb$ eb use prod
~/eb$ git checkout develop
~/eb$ eb use dev
```

## Déploiement des modifications

Par défaut, l'interface de ligne de commande EB déploie la dernière validation (commit) dans la branche actuelle, en utilisant respectivement l'ID de commit et le message comme libellé de version et comme description. Si vous souhaitez effectuer un déploiement dans votre environnement sans validation, vous pouvez utiliser l'option --staged afin de déployer les modifications qui ont été ajoutées à la zone de transit.

Pour déployer les modifications sans validation

1. Ajoutez les fichiers nouveaux et modifiés à la zone de transit :

```
~/eb$ git add .
```

2. Déployez les modifications indexées avec eb deploy :

```
~/eb$ eb deploy --staged
```

Si vous avez configuré l'interface de ligne de commande EB pour [déployer un artefact \(p. 1039\)](#) et que vous ne le validez pas dans le référentiel git, utilisez l'option --staged pour déployer la dernière version.

## Utilisation des sous-modules Git

Certains projets de code tirent parti des sous-modules Git, des référentiels dans le référentiel de niveau supérieur. Lorsque vous déployez votre code à l'aide de eb create ou de eb deploy, l'interface de commande EB peut inclure des sous-modules dans le fichier zip de la version de l'application et les charger avec le reste du code.

Vous pouvez contrôler l'inclusion de sous-modules à l'aide de l'option `include_git_submodules` dans la section `global` du fichier de configuration de l'interface de ligne de commande EB, le fichier `.elasticbeanstalk/config.yml` dans votre dossier de projet.

Pour inclure des sous-modules, définissez cette option sur `true` :

```
global:  
  include_git_submodules: true
```

Lorsque l'option `include_git_submodules` est manquante ou définie sur `false`, l'interface de ligne de commande EB n'inclut pas de sous-modules dans le fichier zip chargé.

Consultez [Git Tools - Submodules](#) pour obtenir plus d'informations sur les sous-modules Git.

#### Comportement par défaut

Lorsque vous exécutez `eb init` pour configurer votre projet, l'interface de ligne de commande EB ajoute l'option `include_git_submodules` et la définit sur `true`. De cette manière, tous les sous-modules de votre projet sont inclus dans vos déploiements.

L'interface de ligne de commande EB n'a pas toujours pris en charge l'inclusion de sous-modules. Pour éviter toute modification accidentelle ou indésirable aux projets qui existaient avant l'ajout de la prise en charge des sous-modules, l'interface de ligne de commande EB n'inclut pas les sous-modules pour lesquels l'option `include_git_submodules` est manquante. Si vous disposez d'un tel projet et que vous souhaitez inclure des sous-modules dans vos déploiements, ajoutez l'option et définissez-la sur `true`, comme expliqué dans cette section.

#### Comportement CodeCommit

L'intégration d'Elastic Beanstalk avec [CodeCommit \(p. 1048\)](#) ne prend pas en charge les sous-modules pour le moment. Si vous avez activé votre environnement afin de l'intégrer à CodeCommit, les sous-modules ne sont pas inclus dans vos déploiements.

## Affectation de balises Git à votre version de l'application

Vous pouvez utiliser une balise Git comme libellé de version afin d'identifier la version d'application en cours d'exécution dans votre environnement. Par exemple, saisissez la chaîne suivante :

```
~/eb$ git tag -a v1.0 -m "My version 1.0"
```

## Utilisation de l'interface de ligne de commande EB avec AWS CodeCommit

Vous pouvez utiliser l'interface de ligne de commande EB pour déployer votre application directement depuis votre référentiel AWS CodeCommit. Avec CodeCommit, vous pouvez charger uniquement les modifications apportées au référentiel lorsque vous déployez l'application, au lieu de charger l'ensemble du projet. Vous économisez ainsi du temps et de la bande passante si vous avez un vaste projet ou une connexion Internet limitée. L'interface de ligne de commande EB effectue une transmission de type push de vos validations locales et les emploie pour créer les versions de l'application lorsque vous utilisez `eb appversion`, `eb create` ou `eb deploy`.

Pour déployer vos modifications, l'intégration CodeCommit exige de valider d'abord les modifications. Cependant, lorsque vous développez ou déboguez, vous pouvez ne pas vouloir pousser les modifications que vous n'avez pas confirmées comme opérationnelles. Vous pouvez éviter de valider vos modifications en les définissant comme intermédiaires et en utilisant `eb deploy --staged` (qui effectue un déploiement standard). Ou validez d'abord vos modifications sur une branche de développement ou de test, puis fusionnez-les avec votre branche principale uniquement lorsque votre code est prêt. Avec `eb use`, vous

pouvez configurer l'interface de ligne de commande EB pour déployer vers un environnement de votre branche de développement et vers un autre environnement de votre branche principale.

#### Note

Certaines régions ne proposent pas CodeCommit. L'intégration entre Elastic Beanstalk et CodeCommit ne fonctionne pas dans ces régions.

Pour de plus amples informations sur les services AWS proposés dans chaque région, veuillez consulter le [Tableau des régions](#).

#### Sections

- [Prérequis \(p. 1049\)](#)
- [Création d'un référentiel CodeCommit avec l'interface de ligne de commande EB \(p. 1049\)](#)
- [Déploiement à partir de votre référentiel CodeCommit \(p. 1050\)](#)
- [Configuration de branches et environnements supplémentaires \(p. 1051\)](#)
- [Utilisation d'un référentiel CodeCommit existant \(p. 1052\)](#)

## Prérequis

Pour utiliser CodeCommit avec AWS Elastic Beanstalk, vous avez besoin d'un référentiel Git local (celui que vous possédez déjà ou un nouveau que vous créez) ayant au moins une validation, [l'autorisation d'utiliser CodeCommit](#) et un environnement Elastic Beanstalk dans une région prise en charge par CodeCommit. Votre environnement et votre référentiel doivent être dans la même région.

#### Pour initialiser un référentiel Git

1. Exécutez `git init` dans votre dossier de projet.

```
~/my-app$ git init
```

2. Organisez vos fichiers de projet avec `git add`.

```
~/my-app$ git add .
```

3. Validez les modifications avec `git commit`.

```
~/my-app$ git commit -m "Elastic Beanstalk application"
```

## Création d'un référentiel CodeCommit avec l'interface de ligne de commande EB

Pour démarrer avec CodeCommit, exécutez la commande [eb init \(p. 1093\)](#). Lors de la configuration du référentiel, l'interface de ligne de commande EB vous invite à utiliser CodeCommit pour stocker votre code et accélérer les déploiements. Même si vous avez déjà configuré votre projet à l'aide de la commande eb init, vous pouvez l'exécuter à nouveau pour configurer CodeCommit.

#### Pour créer un référentiel CodeCommit avec l'interface de ligne de commande EB

1. Exécutez `eb init` dans votre dossier de projet. Lors de la configuration, l'interface de ligne de commande EB vous demande si vous souhaitez utiliser CodeCommit pour stocker votre code et accélérer les déploiements. Si vous avez déjà configuré votre projet à l'aide de la commande `eb init`,

vous pouvez l'exécuter à nouveau pour configurer CodeCommit. Entrez **y** à l'invite pour configurer CodeCommit.

```
~/my-app$ eb init
Note: Elastic Beanstalk now supports AWS CodeCommit; a fully-managed source control
service. To learn more, see Docs: https://aws.amazon.com/codecommit/
Do you wish to continue with CodeCommit? (y/n)(default is n): y
```

2. Choisissez Create new Repository (Créer un nouveau référentiel).

```
Select a repository
1) my-repo
2) [ Create new Repository ]
(default is 2): 2
```

3. Tapez un nom de référentiel ou appuyez sur Entrée pour accepter le nom par défaut.

```
Enter Repository Name
(default is "codecommit-origin"): my-app
Successfully created repository: my-app
```

4. Sélectionnez une branche existante pour vos validations ou utilisez l'interface de ligne de commande EB pour créer une nouvelle branche.

```
Enter Branch Name
***** Must have at least one commit to create a new branch with CodeCommit *****
(default is "mainline"): ENTER
Successfully created branch: mainline
```

## Déploiement à partir de votre référentiel CodeCommit

Lorsque vous configurez CodeCommit avec votre référentiel, l'interface de ligne de commande EB utilise le contenu du référentiel pour créer des bundles de fichiers source. Lorsque vous exécutez eb deploy ou eb create, l'interface de ligne de commande EB exécute un push sur les nouvelles validations et utilise la révision HEAD de votre branche pour créer l'archive déployée sur les instances EC2 de votre environnement.

Pour utiliser l'intégration CodeCommit avec l'interface de ligne de commande EB

1. Créez un nouvel environnement avec eb create.

```
~/my-app$ eb create my-app-env
Starting environment deployment via CodeCommit
--- Waiting for application versions to be pre-processed ---
Finished processing application version app-ac1ea-161010_201918
Setting up default branch
Environment details for: my-app-env
  Application name: my-app
  Region: us-east-2
  Deployed Version: app-ac1ea-161010_201918
  Environment ID: e-pm5mvvkfnd
  Platform: 64bit Amazon Linux 2016.03 v2.1.6 running Java 8
  Tier: WebServer-Standard
  CNAME: UNKNOWN
  Updated: 2016-10-10 20:20:29.725000+00:00
  Printing Status:
  INFO: createEnvironment is starting.
  ...
```

L'interface de ligne de commande EB utilise la dernière validation de la branche suivie pour créer la version de l'application déployée dans l'environnement.

2. Lorsque vous avez de nouvelles validations locales, utilisez eb deploy pour pousser les validations et les déployer dans votre environnement.

```
~/my-app$ eb deploy
Starting environment deployment via CodeCommit
INFO: Environment update is starting.
INFO: Deploying new version to instance(s).
INFO: New application version was deployed to running EC2 instances.
INFO: Environment update completed successfully.
```

3. Pour tester les modifications avant de les valider, utilisez l'option --staged pour déployer les modifications que vous avez ajoutées à la zone intermédiaire avec git add.

```
~/my-app$ git add new-file
~/my-app$ eb deploy --staged
```

Le déploiement avec l'option --staged effectue un déploiement standard, qui ignore CodeCommit.

## Configuration de branches et environnements supplémentaires

La configuration CodeCommit s'applique à une seule branche. Vous pouvez utiliser eb use et eb codesource pour configurer des branches supplémentaires ou modifier la configuration de la branche actuelle.

Pour configurer l'intégration CodeCommit avec l'interface de ligne de commande EB

1. Pour modifier la branche distante, utilisez l'option [eb use \(p. 1123\)](#) de la commande --source.

```
~/my-app$ eb use test-env --source my-app/test
```

2. Pour créer une nouvelle branche et un nouvel environnement, extrayez une nouvelle branche, transmettez-la à CodeCommit, créez l'environnement, puis utilisez la commande eb use pour connecter la branche locale, la branche distante et l'environnement.

```
~/my-app$ git checkout -b production
~/my-app$ git push --set-upstream production
~/my-app$ eb create production-env
~/my-app$ eb use --source my-app/production production-env
```

3. Pour configurer CodeCommit de manière interactive, utilisez la commande [eb codesource codecommit \(p. 1070\)](#).

```
~/my-app$ eb codesource codecommit
Current CodeCommit setup:
  Repository: my-app
  Branch: test
Do you wish to continue (y/n): y

Select a repository
1) my-repo
2) my-app
3) [ Create new Repository ]
```

```
(default is 2): 2

Select a branch
1) mainline
2) test
3) [ Create new Branch with local HEAD ]
(default is 1): 1
```

- Pour désactiver l'intégration CodeCommit, utilisez la commande [eb codesource local \(p. 1070\)](#).

```
~/my-app$ eb codesource local
Current CodeCommit setup:
  Repository: my-app
  Branch: mainline
Default set to use local sources
```

## Utilisation d'un référentiel CodeCommit existant

Si vous possédez déjà un référentiel CodeCommit et que vous souhaitez l'utiliser avec Elastic Beanstalk, exécutez la commande eb init à la racine de votre référentiel Git local.

Pour utiliser un référentiel CodeCommit existant avec l'interface de ligne de commande EB

- Clonez votre référentiel CodeCommit.

```
~$ git clone ssh://git-codecommit.us-east-2.amazonaws.com/v1/repos/my-app
```

- Extrayez et transmettez une branche à utiliser pour votre environnement Elastic Beanstalk.

```
~/my-app$ git checkout -b dev-env
~/my-app$ git push --set-upstream origin dev-env
```

- Exécutez eb init. Sélectionnez les mêmes région, référentiel et nom de branche que ceux que vous utilisez actuellement.

```
~/my-app$ eb init
Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : Europe (Ireland)
5) eu-central-1 : Europe (Frankfurt)
6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
...
(default is 3): 1
...
Note: Elastic Beanstalk now supports AWS CodeCommit; a fully-managed source control
service. To learn more, see Docs: https://aws.amazon.com/codecommit/
Do you wish to continue with CodeCommit? (y/n)(default is n): y

Select a repository
1) my-app
2) [ Create new Repository ]
(default is 1): 1

Select a branch
1) mainline
2) dev-env
3) [ Create new Branch with local HEAD ]
```

```
(default is 2): 2
```

Pour de plus amples informations sur l'utilisation de la commande eb init, veuillez consulter [Configuration de l'interface de ligne de commande EB \(p. 1036\)](#).

## Utilisation de l'interface de ligne de commande EB pour surveiller l'intégrité de l'environnement

L'[interface de ligne de commande Elastic Beanstalk \(p. 1027\)](#) (EB CLI) est un outil de ligne de commande pour la gestion des environnements AWS Elastic Beanstalk. Vous pouvez également utiliser l'interface de ligne de commande EB pour surveiller l'intégrité de votre environnement en temps réel et avec davantage de granularité que celle qui est actuellement disponible dans la console Elastic Beanstalk.

Après avoir [installé \(p. 1028\)](#) et [configuré \(p. 1036\)](#) l'interface de ligne de commande EB, vous pouvez [lancer un nouvel environnement \(p. 1040\)](#) et y déployer votre code avec la commande eb create. Si vous avez déjà un environnement que vous avez créé dans la console Elastic Beanstalk, vous pouvez y attacher l'interface de ligne de commande EB en exécutant eb init dans un dossier de projet et en suivant les invites (le dossier du projet peut être vide).

### Important

Assurez-vous que vous utilisez la dernière version de l'interface de ligne de commande EB en exécutant pip install avec l'option --upgrade :

```
$ sudo pip install --upgrade awsebcli
```

Pour des instructions complètes d'installation de l'interface de ligne de commande EB, veuillez consulter [Installation de l'interface de ligne de commande EB \(p. 1028\)](#).

Pour utiliser l'interface de ligne de commande EB pour surveiller l'intégrité de votre environnement, vous devez tout d'abord configurer un dossier local du projet en exécutant eb init puis en suivant les invites. Pour obtenir des instructions complètes, veuillez consulter [Configuration de l'interface de ligne de commande EB \(p. 1036\)](#).

Si un environnement s'exécute déjà dans Elastic Beanstalk et que vous souhaitez utiliser l'interface de ligne de commande EB pour surveiller son état, procédez comme suit pour l'attacher à l'environnement existant.

Pour attacher l'interface de ligne de commande EB à un environnement existant

1. Ouvrez une terminal de ligne de commande et accédez à votre dossier utilisateur.
2. Créez et ouvrez un dossier pour votre environnement.
3. Exécutez la commande eb init, puis choisissez l'application et l'environnement dont vous souhaitez analyser l'intégrité. Si un seul environnement exécute l'application que vous avez choisie, l'interface de ligne de commande EB le sélectionne automatiquement et vous n'avez pas à choisir l'environnement, comme illustré dans l'exemple suivant.

```
~/project$ eb init
Select an application to use
1) elastic-beanstalk-example
2) [ Create new Application ]
(default is 2): 1
Select the default environment.
You can change this later by typing "eb use [environment_name]".
1) elasticBeanstalkEx2-env
```

```
2) elasticBeanstalkExa-env
(default is 1)
```

Pour surveillez l'intégrité en utilisant l'interface de ligne de commande EB

1. Ouvrez une ligne de commande et accédez au dossier de votre projet.
2. Exécutez la commande eb health pour afficher l'état d'intégrité des instances dans votre environnement. Dans cet exemple, il y a cinq instances en cours d'exécution dans un environnement Linux.

```
~/project $ eb health
elasticBeanstalkExa-env
2015-07-08 23:13:20
Ok
WebServer
Ruby 2.1 (Puma)
total      ok     warning   degraded   severe     info    pending   unknown
      5        5          0           0          0          0        0         0         0

instance-id  status      cause
                           health
Overall      Ok
i-d581497d  Ok
i-d481497c  Ok
i-136e00c0  Ok
i-126e00c1  Ok
i-8b2cf575  Ok

instance-id  r/sec    %2xx    %3xx    %4xx    %5xx
                           requests
p10
Overall    671.8    100.0    0.0     0.0     0.0
0.000
i-d581497d 143.0    1430     0       0       0
0.000
i-d481497c 128.8    1288     0       0       0
0.000
i-136e00c0 125.4    1254     0       0       0
0.000
i-126e00c1 133.4    1334     0       0       0
0.000
i-8b2cf575 141.2    1412     0       0       0
0.000

instance-id  type      az  running
idle% iowait% t2.micro 1a  12 mins
92.5    0.1
i-d581497d t2.micro 1a  12 mins
92.4    0.1
i-d481497c t2.micro 1a  12 mins
93.2    0.0
i-136e00c0 t2.micro 1b  12 mins
92.7    0.1
i-126e00c1 t2.micro 1b  12 mins
92.1    0.1
i-8b2cf575 t2.micro 1c  1 hour

instance-id  status      id  version
                           deployments
ago
i-d581497d  Deployed   1   Sample Application
12 mins
i-d481497c  Deployed   1   Sample Application
12 mins
i-136e00c0  Deployed   1   Sample Application
12 mins
i-126e00c1  Deployed   1   Sample Application
12 mins
i-8b2cf575  Deployed   1   Sample Application
1 hour
```

Dans cet exemple, il y a une seule instance en cours d'exécution dans un environnement Windows.

```
~/project $ eb health
WindowsSampleApp-env                                         Ok
2018-05-22 17:33:19
WebServer                                                 IIS 10.0 running on 64bit
Windows Server 2016/2.2.0
total      ok       warning   degraded   severe    info     pending   unknown
1          1         0           0          0         0        0         0         0

instance-id      status      cause
                           health
Overall          Ok
i-065716fba0e08a351  Ok

instance-id      r/sec      %2xx    %3xx    %4xx    %5xx    p99     p90     p75
p50             p10
Overall          13.7     100.0    0.0     0.0     0.0     1.403   0.970   0.710
0.413          0.079
i-065716fba0e08a351  2.4      100.0    0.0     0.0     0.0     1.102*  0.865   0.601
0.413          0.091

instance-id      type      az      running      % user time      % privileged time  %
idle time
i-065716fba0e08a351 t2.large  1b     4 hours            0.2                  0.1

instance-id      status      id      version      deployments      ago
                           deployments
i-065716fba0e08a351 Deployed   2      Sample Application  4 hours
```

## Lecture du résultat

Le résultat affiche le nom de l'environnement, l'intégrité globale de l'environnement et la date actuelle en haut de l'écran.

```
elasticBeanstalkExa-env                                         Ok
2015-07-08 23:13:20
```

Les trois lignes suivantes affichent le type d'environnement (« serveur web » dans le cas présent), la configuration (Ruby 2.1 avec Puma) et une répartition du nombre d'instances dans chacun des sept états.

```
WebServer                                                 Ruby
2.1 (Puma)
total      ok       warning   degraded   severe    info     pending   unknown
5          5         0           0          0         0        0         0         0
```

Le reste du résultat est divisé en quatre sections. La première affiche l'état et la cause de l'état de l'environnement dans l'ensemble, puis pour chaque instance. L'exemple suivant montre deux instances de l'environnement avec un état Info et une cause indiquant qu'un déploiement a commencé.

```
instance-id      status      cause
                           health
Overall          Ok
i-d581497d      Info       Performing application deployment (running for 3 seconds)
i-d481497c      Info       Performing application deployment (running for 3 seconds)
i-136e00c0      Ok
i-126e00c1      Ok
```

i-8b2cf575	Ok
------------	----

Pour de plus amples informations sur les couleurs et les états d'intégrité, veuillez consulter [Couleurs et états utilisés dans les rapports d'intégrité \(p. 854\)](#).

La section requests (demandes) affiche des informations à partir des journaux du serveur web sur chaque instance. Dans cet exemple, chaque instance accepte les demandes normalement et il n'y a pas d'erreurs.

instance-id	r/sec	%2xx	%3xx	%4xx	%5xx	p99	p90	p75	p50
p10									
Overall	13.7	100.0	0.0	0.0	0.0	1.403	0.970	0.710	0.413
0.079									
i-d581497d	2.4	100.0	0.0	0.0	0.0	1.102*	0.865	0.601	0.413
0.091									
i-d481497c	2.7	100.0	0.0	0.0	0.0	0.842*	0.788	0.480	0.305
0.062									
i-136e00c0	4.1	100.0	0.0	0.0	0.0	1.520*	1.088	0.883	0.524
0.104									
i-126e00c1	2.2	100.0	0.0	0.0	0.0	1.334*	0.791	0.760	0.344
0.197									
i-8b2cf575	2.3	100.0	0.0	0.0	0.0	1.162*	0.867	0.698	0.477
0.076									

La section cpu illustre les métriques du système d'exploitation pour chaque instance. La sortie diffère selon le système d'exploitation. Voici la sortie pour les environnements Linux.

instance-id	type	az	running	cpu	load 1	load 5	user%	nice%	system%	idle
% iowait%										
i-d581497d	t2.micro	1a	12 mins		0.0	0.03	0.2	0.0	0.0	0.0
99.7 0.1										
i-d481497c	t2.micro	1a	12 mins		0.0	0.03	0.3	0.0	0.0	0.0
99.7 0.0										
i-136e00c0	t2.micro	1b	12 mins		0.0	0.04	0.1	0.0	0.0	0.0
99.9 0.0										
i-126e00c1	t2.micro	1b	12 mins		0.01	0.04	0.2	0.0	0.0	0.0
99.7 0.1										
i-8b2cf575	t2.micro	1c	1 hour		0.0	0.01	0.2	0.0	0.1	
99.6 0.1										

Voici la sortie pour les environnements Windows.

instance-id	type	az	running	% user time	% privileged time	%
idle time						
i-065716fba0e08a351	t2.large	1b	4 hours	0.2		0.0
99.8						

Pour de plus amples informations sur les métriques du serveur et du système d'exploitation affichées, veuillez consulter [Métriques des instances \(p. 856\)](#).

La section finale, deployments (déploiements), présente l'état de déploiement de chaque instance. En cas d'échec d'un déploiement propagé, vous pouvez utiliser l'ID, le statut et l'étiquette de version indiqués pour identifier les instances de votre environnement exécutant la mauvaise version.

instance-id	status	id	version	ago
			deployments	
i-d581497d	Deployed	1	Sample Application	12 mins
i-d481497c	Deployed	1	Sample Application	12 mins
i-136e00c0	Deployed	1	Sample Application	12 mins
i-126e00c1	Deployed	1	Sample Application	12 mins
i-8b2cf575	Deployed	1	Sample Application	1 hour

## Vue d'intégrité interactive

La commande eb health affiche un instantané de l'intégrité de votre environnement. Pour actualiser les informations affichées toutes les 10 secondes, utilisez l'option --refresh.

```
$ eb health --refresh
elasticBeanstalkExa-env
2015-07-09 22:10:04 (1 secs)
WebServer
    Ruby 2.1 (Puma)
total      ok     warning   degraded   severe     info     pending   unknown
  5          5         0           0           0           0         0         0         0
instance-id  status    cause
                           health
Overall      Ok
i-bb65c145  Ok        Application deployment completed 35 seconds ago and took 26
seconds
i-ba65c144  Ok        Application deployment completed 17 seconds ago and took 25
seconds
i-f6a2d525  Ok        Application deployment completed 53 seconds ago and took 26
seconds
i-e8a2d53b  Ok        Application deployment completed 32 seconds ago and took 31
seconds
i-e81cca40  Ok

instance-id  r/sec    %2xx    %3xx    %4xx    %5xx    p99     p90     p75     p50
p10
Overall      671.8   100.0    0.0     0.0     0.0     0.003   0.002   0.001   0.001
0.000
i-bb65c145  143.0   1430     0       0       0       0.003   0.002   0.001   0.001
0.000
i-ba65c144  128.8   1288     0       0       0       0.003   0.002   0.001   0.001
0.000
i-f6a2d525  125.4   1254     0       0       0       0.004   0.002   0.001   0.001
0.000
i-e8a2d53b  133.4   1334     0       0       0       0.003   0.002   0.001   0.001
0.000
i-e81cca40  141.2   1412     0       0       0       0.003   0.002   0.001   0.001
0.000

instance-id  type      az      running      load 1      load 5      user%    nice%    system%    idle
% iowait%
i-bb65c145  t2.micro  1a     12 mins     0.0       0.03       0.2       0.0       0.0       0.0
99.7       0.1
i-ba65c144  t2.micro  1a     12 mins     0.0       0.03       0.3       0.0       0.0       0.0
99.7       0.0
i-f6a2d525  t2.micro  1b     12 mins     0.0       0.04       0.1       0.0       0.0       0.0
99.9       0.0
i-e8a2d53b  t2.micro  1b     12 mins     0.01      0.04       0.2       0.0       0.0       0.0
99.7       0.1
i-e81cca40  t2.micro  1c     1 hour      0.0       0.01       0.2       0.0       0.0       0.1
99.6       0.1

instance-id  status    id      version      deployments      ago
i-bb65c145  Deployed  1      Sample Application  12 mins
i-ba65c144  Deployed  1      Sample Application  12 mins
i-f6a2d525  Deployed  1      Sample Application  12 mins
i-e8a2d53b  Deployed  1      Sample Application  12 mins
i-e81cca40  Deployed  1      Sample Application  1 hour

.Commands: Help,Quit, # # # #)
```

Cet exemple illustre un environnement qui a récemment été monté en puissance d'une à cinq instances. L'opération de mise à l'échelle a abouti, et toutes les instances transmettent maintenant des vérifications de l'état et sont prêtes à prendre des demandes. En mode interactif, l'état d'intégrité est mis à jour toutes les 10 secondes. Dans l'angle supérieur droit, un minuteur assure le décompte jusqu'à la prochaine mise à jour.

Dans l'angle inférieur gauche, le rapport affiche une liste d'options. Pour quitter le mode interactif, appuyez sur Q. Pour faire défiler, appuyez sur les flèches. Pour afficher une liste de commandes supplémentaires, appuyez sur H.

## Options d'affichage d'intégrité interactif

Lors de l'affichage d'intégrité de l'environnement de manière interactive, vous pouvez utiliser les touches du clavier pour ajuster la vue et dire à Elastic Beanstalk de remplacer ou de redémarrer des instances individuelles. Pour afficher la liste des commandes disponibles lors de la consultation du rapport d'intégrité en mode interactif, appuyez sur H.

```
up,down,home,end      Scroll vertically
left,right            Scroll horizontally
F                     Freeze/unfreeze data
X                     Replace instance
B                     Reboot instance
<,>                 Move sort column left/right
-,+                 Sort order descending/ascending
P                     Save health snapshot data file
Z                     Toggle color/mono mode
Q                     Quit this program

Views
1                   All tables/split view
2                   Status Table
3                   Request Summary Table
4                   CPU%/Load Table
H                   This help menu

(press Q or ESC to return)
```

## Gestion de plusieurs environnements Elastic Beanstalk en tant que groupe avec l'interface de ligne de commande EB

Vous pouvez utiliser l'interface de ligne de commande EB pour créer des groupes d'environnements AWS Elastic Beanstalk, chacun d'entre eux exécutant un composant distinct d'une application d'architecture orientée services. L'interface de ligne de commande EB gère ces groupes à l'aide de l'API [ComposeEnvironments](#).

### Note

Les groupes d'environnements sont différents des conteneurs multiples qui composent un environnement Docker multiconteneurs. Avec les groupes d'environnements, chaque composant de votre application s'exécute dans un environnement Elastic Beanstalk distinct, avec son propre ensemble d'instances Amazon EC2. Chaque composant peut être mis à l'échelle séparément. Avec Docker multiconteneurs, vous pouvez combiner plusieurs composants d'une application en un seul environnement. Tous les composants partagent le même ensemble d'instances

Amazon EC2, et chaque instance exécute plusieurs conteneurs Docker. Choisissez l'une de ces architectures en fonction des besoins de votre application.

Pour plus de détails sur Docker multi-conteneurs, consultez [Utilisation de la plateforme Docker multiconteneurs \(AMI Amazon Linux\) \(p. 64\)](#).

Organisez les composants de votre application selon la structure de dossier suivante :

```
~/project-name
|-- component-a
|   '-- env.yaml
`-- component-b
    '-- env.yaml
```

Chaque sous-dossier contient le code source d'un composant indépendant d'une application qui s'exécutera dans son propre environnement et un fichier de définition d'environnement nommé `env.yaml`. Pour de plus amples informations sur le format `env.yaml`, veuillez consulter [Manifeste d'environnement \(env.yaml\) \(p. 785\)](#).

Pour utiliser l'API `Compose Environments`, commencez par exécuter `eb init` à partir du dossier de projet, en spécifiant chaque composant par le nom du dossier qui le contient via l'option `--modules` :

```
~/workspace/project-name$ eb init --modules component-a component-b
```

L'interface de ligne de commande EB vous invite à [configurer chaque composant \(p. 1036\)](#), puis crée le répertoire `.elasticbeanstalk` dans le dossier de chaque composant. L'interface de ligne de commande EB ne crée pas les fichiers de configuration dans le répertoire parent.

```
~/project-name
|-- component-a
|   |-- .elasticbeanstalk
|   |   '-- env.yaml
`-- component-b
    |-- .elasticbeanstalk
    |   '-- env.yaml
```

Ensuite, exécutez la commande `eb create` avec une liste des environnements à créer (un pour chaque composant) :

```
~/workspace/project-name$ eb create --modules component-a component-b --env-group-
suffix group-name
```

Cette commande crée un environnement pour chaque composant. Les noms des environnements sont créés en concaténant la valeur `EnvironmentName` spécifiée dans le fichier `env.yaml` avec le nom du groupe, séparé par un trait d'union. La longueur totale de ces deux options et du trait d'union ne doit pas dépasser la longueur maximale autorisée pour le nom d'un environnement, soit 23 caractères.

Pour mettre à jour l'environnement, utilisez la commande `eb deploy` :

```
~/workspace/project-name$ eb deploy --modules component-a component-b
```

Vous pouvez mettre à jour chaque composant individuellement ou les mettre à jour en tant que groupe. Spécifiez les composants que vous souhaitez mettre à jour avec l'option `--modules`.

L'interface de ligne de commande EB stocke le nom du groupe que vous avez utilisé avec `eb create` dans la section `branch-defaults` du fichier de configuration de l'interface de ligne de commande EB, sous `/.elasticbeanstalk/config.yaml`. Pour déployer votre application vers un autre groupe, utilisez

l'option `--env-group-suffix` lorsque vous exécutez `eb deploy`. Si le groupe n'existe pas encore, l'interface de ligne de commande EB crée un groupe d'environnements :

```
~/workspace/project-name$ eb deploy --modules component-a component-b --env-group-suffix group-2-name
```

Pour mettre des environnements hors service, exécutez `eb terminate` dans le dossier pour chaque module. Par défaut, l'interface de ligne de commande EB affiche une erreur si vous essayez de suspendre un environnement dont dépend un autre environnement en cours d'exécution. Commencez par suspendre l'environnement dépendant ou utilisez l'option `--ignore-links` pour remplacer le comportement par défaut :

```
~/workspace/project-name/component-b$ eb terminate --ignore-links
```

## Résolution de problèmes avec l'interface de ligne de commande EB

Cette rubrique répertorie les messages d'erreur courants rencontrés lors de l'utilisation de l'interface de ligne de commande EB et les solutions possibles. Si vous rencontrez un message d'erreur qui n'est pas présenté ici, utilisez les liens Feedback pour nous en faire part.

**ERROR: An error occurred while handling git command. Error code: 128** **Error: fatal: Not a valid object name HEAD**

Cause : ce message d'erreur apparaît lorsque vous avez initialisé un référentiel Git mais que vous ne l'avez pas encore validé. L'interface de ligne de commande EB cherche la révision HEAD lorsque votre dossier de projet contient un référentiel Git.

Solution : ajoutez les fichiers dans votre dossier de projet à la zone intermédiaire et valides :

```
~/my-app$ git add .
~/my-app$ git commit -m "First commit"
```

**ERROR: This branch does not have a default environment. You must either specify an environment by typing "eb status my-env-name" or set a default environment by typing "eb use my-env-name".**

Cause : lorsque vous créez une branche dans git, elle n'est pas attachée à un environnement Elastic Beanstalk par défaut.

Solution : exécutez `eb list` pour afficher une liste des environnements disponibles. Exécutez ensuite `eb use env-name` pour utiliser l'un des environnements disponibles.

**ERROR: 2.0+ Platforms require a service role. You can provide one with --service-role option**

Cause : si vous spécifiez un nom d'environnement avec `eb create` (par exemple, `eb create my-env`), l'interface de ligne de commande EB n'essaie pas de créer un rôle de service pour vous. Si vous n'avez pas le rôle de service par défaut, l'erreur ci-dessus s'affiche.

Solution : exécutez `eb create` sans un nom d'environnement et suivez les instructions pour créer le rôle du service par défaut.

## Résolution de problèmes de déploiement

Si votre déploiement Elastic Beanstalk ne s'est pas déroulé comme prévu, vous pouvez obtenir une réponse 404 (en cas d'échec du lancement de votre application) ou 500 (en cas d'échec de votre

application pendant l'exécution), au lieu de voir votre site web. Pour résoudre de nombreuses questions courantes, vous pouvez utiliser l'interface de ligne de commande EB pour vérifier l'état de votre déploiement, afficher ses journaux, accéder à votre instance EC2 avec SSH ou ouvrir la page de la console de gestion AWS pour votre environnement d'application.

Pour utiliser l'interface de ligne de commande EB pour aider à dépanner votre déploiement

1. Exécutez eb status pour afficher l'état de votre déploiement actuel et l'état de vos hôtes EC2. Par exemple :

```
$ eb status --verbose

Environment details for: python_eb_app
  Application name: python_eb_app
  Region: us-west-2
  Deployed Version: app-150206_035343
  Environment ID: e-wa8u6rrmqy
  Platform: 64bit Amazon Linux 2014.09 v1.1.0 running Python 2.7
  Tier: WebServer-Standard-
  CNAME: python_eb_app.elasticbeanstalk.com
  Updated: 2015-02-06 12:00:08.557000+00:00
  Status: Ready
  Health: Green
  Running instances: 1
    i-8000528c: InService
```

#### Note

L'utilisation du commutateur `--verbose` fournit des informations sur l'état de vos instances en cours d'exécution. Sans cela, eb status imprimera uniquement des informations générales sur votre environnement.

2. Exécutez eb health afin d'afficher des informations d'intégrité sur votre environnement :

```
$ eb health --refresh
elasticBeanstalkExa-env                                         Degraded
2016-03-28 23:13:20

WebServer
Ruby 2.1 (Puma)
  total      ok     warning   degraded   severe     info     pending   unknown
      5        2          0         2         1         0         0         0         0

  instance-id  status      cause
  Overall      Degraded  Incorrect application version found on 3 out of 5 instances.
  Expected version "Sample Application" (deployment 1).
  i-d581497d  Degraded  Incorrect application version "v2" (deployment 2). Expected
  version "Sample Application" (deployment 1).
  i-d481497c  Degraded  Incorrect application version "v2" (deployment 2). Expected
  version "Sample Application" (deployment 1).
  i-136e00c0  Severe   Instance ELB health has not been available for 5 minutes.
  i-126e00c1  Ok
  i-8b2cf575  Ok

  instance-id  r/sec    %2xx    %3xx    %4xx    %5xx      p99      p90      p75      p50
  p10
  Overall      646.7   100.0    0.0     0.0     0.0     0.003   0.002   0.001   0.001
  0.000
  i-dac3f859   167.5   1675     0       0       0       0.003   0.002   0.001   0.001
  0.000
  i-05013a81   161.2   1612     0       0       0       0.003   0.002   0.001   0.001
  0.000
  i-04013a80   0.0      -        -       -        -       -        -        -        -        -
```

i-3ab524a1	155.9	1559	0	0	0	0.003	0.002	0.001	0.001
0.000									
i-bf300d3c	162.1	1621	0	0	0	0.003	0.002	0.001	0.001
0.000									
<b>instance-id type az running load 1 load 5 user% nice% system%</b>									
idle%	iowait%								
i-d581497d	t2.micro	1a	25 mins	0.16	0.1	7.0	0.0	1.7	
91.0	0.1								
i-d481497c	t2.micro	1a	25 mins	0.14	0.1	7.2	0.0	1.6	
91.1	0.0								
i-136e00c0	t2.micro	1b	25 mins	0.0	0.01	0.0	0.0	0.0	
99.9	0.1								
i-126e00c1	t2.micro	1b	25 mins	0.03	0.08	6.9	0.0	2.1	
90.7	0.1								
i-8b2cf575	t2.micro	1c	1 hour	0.05	0.41	6.9	0.0	2.0	
90.9	0.0								
<b>instance-id status id version ago</b>									
<b>deployments</b>									
i-d581497d	Deployed	2	v2		9 mins				
i-d481497c	Deployed	2	v2		7 mins				
<b>i-136e00c0</b>	<b>Failed</b>	<b>2</b>	<b>v2</b>		<b>5 mins</b>				
i-126e00c1	Deployed	1	Sample Application		25 mins				
i-8b2cf575	Deployed	1	Sample Application		1 hour				

L'exemple ci-dessus illustre un environnement avec cinq cas où le déploiement de la version « v2 » a échoué sur la troisième instance. Après un échec de déploiement, la version attendue revient sur la dernière version qui a abouti, ce qui dans ce cas correspond à « Exemple d'Application » depuis le premier déploiement. Pour de plus amples informations, veuillez consulter [Utilisation de l'interface de ligne de commande EB pour surveiller l'intégrité de l'environnement \(p. 1053\)](#).

- Exécutez eb logs pour télécharger et afficher les journaux associés au déploiement de votre application.

```
$ eb logs
```

- Exécutez eb ssh pour vous connecter à l'instance EC2 qui exécute votre application et examinez-la directement. Sur l'instance, votre application déployée est disponible dans le répertoire /opt/python/current/app et votre environnement Python se trouve dans /opt/python/run/venv/.
- Exécutez eb console pour afficher l'environnement de votre application sur la console de gestion AWS. Vous pouvez utiliser l'interface web afin d'examiner facilement les différents aspects de votre déploiement, y compris les journaux, les événements, l'état et la configuration de votre application. Vous pouvez également télécharger les versions d'application actuelles ou passées que vous avez déployées sur le serveur.

## Guide de référence des commandes de l'interface de ligne de commande (CLI) EB

Vous pouvez utiliser l'interface de ligne de commande Elastic Beanstalk (EB CLI) pour effectuer diverses opérations de déploiement et de gestion de vos applications et environnements Elastic Beanstalk. L'interface de ligne de commande (CLI) EB s'intègre à Git si vous souhaitez déployer le code source de l'application qui est sous le contrôle source Git. Pour de plus amples informations, veuillez consulter [Utilisation de l'interface de ligne de commande Elastic Beanstalk \(EB\) \(p. 1027\)](#) et [Utilisation de l'interface de ligne de commande EB avec Git \(p. 1046\)](#).

### Commandes

- [eb abort \(p. 1063\)](#)
- [eb appversion \(p. 1064\)](#)
- [eb clone \(p. 1068\)](#)
- [eb codesource \(p. 1070\)](#)
- [eb config \(p. 1071\)](#)
- [eb console \(p. 1078\)](#)
- [eb create \(p. 1078\)](#)
- [eb deploy \(p. 1089\)](#)
- [eb events \(p. 1090\)](#)
- [eb health \(p. 1092\)](#)
- [eb init \(p. 1093\)](#)
- [eb labs \(p. 1096\)](#)
- [eb list \(p. 1096\)](#)
- [eb local \(p. 1097\)](#)
- [eb logs \(p. 1100\)](#)
- [eb open \(p. 1102\)](#)
- [eb platform \(p. 1103\)](#)
- [eb printenv \(p. 1110\)](#)
- [eb restore \(p. 1111\)](#)
- [eb scale \(p. 1112\)](#)
- [eb setenv \(p. 1112\)](#)
- [eb ssh \(p. 1113\)](#)
- [eb status \(p. 1115\)](#)
- [eb swap \(p. 1117\)](#)
- [eb tags \(p. 1118\)](#)
- [eb terminate \(p. 1120\)](#)
- [eb upgrade \(p. 1122\)](#)
- [eb use \(p. 1123\)](#)
- [Options courantes \(p. 1123\)](#)

## eb abort

### Description

Annule une mise à niveau lorsque les modifications de la configuration de l'environnement apportées à des instances sont toujours en cours.

#### Note

Si vous avez plus de deux environnements qui font l'objet d'une mise à jour, vous êtes invité à sélectionner le nom de l'environnement pour lequel vous souhaitez annuler les modifications.

### Syntax

eb abort

eb abort **environment-name**

## Options

Nom	Description
Options courantes (p. 1123)	

## Output

La commande affiche une liste d'environnements actuellement mis à jour et vous invite à choisir la mise à jour que vous souhaitez annuler. Si un seul environnement est actuellement en cours de mise à jour, vous n'avez pas besoin de spécifier le nom de l'environnement. En cas de réussite, la commande annule les changements de configuration d'environnement. Le processus de restauration continue jusqu'à ce que toutes les instances de l'environnement aient la configuration de l'environnement précédent ou jusqu'à ce que le processus de restauration échoue.

## Example

L'exemple suivant annule la mise à niveau de la plateforme.

```
$ eb abort
Aborting update to environment "tmp-dev".
<list of events>
```

## eb appversion

### Description

La commande `appversion` de l'interface de ligne de commande (CLI) EB gère les [versions de votre application \(p. 14\)](#) Elastic Beanstalk. Vous pouvez créer une nouvelle version de l'application sans déployer, supprimer une version de l'application ou créer la [stratégie de cycle de vie de la version de l'application \(p. 411\)](#). Si vous appelez la commande sans ajouter d'options, elle passe en [mode interactif \(p. 1066\)](#).

Utilisez l'option `--create` pour créer une nouvelle version de l'application.

Utilisez l'option `--delete` pour supprimer une version de l'application.

Utilisez l'option `lifecycle` pour afficher ou créer la stratégie de cycle de vie d'une version de l'application. Pour plus d'informations, consultez [the section called “Cycle de vie des versions” \(p. 411\)](#).

## Syntax

`eb appversion`

`eb appversion [-c | --create]`

`eb appversion [-d | --delete] version-label`

`eb appversion lifecycle [-p | --print]`

## Options

Nom	Description
	Type : chaîne
-a <i>application-name</i> ou --application_name <i>application-name</i>	Nom de l'application . Si une application portant le nom spécifié n'est pas trouvée, l'interface de ligne de commande (CLI) EB crée une version d'application pour une nouvelle application.  Applicable uniquement avec l'option --create.  Type : chaîne
-c ou --create	Créez une <a href="#">nouvelle version (p. 14)</a> de l'application.
-d <i>version-label</i> ou --delete <i>version-label</i>	Supprimez la version de l'application étiquetée <i>version-label</i> .
-l <i>version_label</i> ou --label <i>version_label</i>	Spécifiez une étiquette à utiliser pour la version créée par l'interface de ligne de commande (CLI) EB. Si vous n'utilisez pas cette option, l'interface de ligne de commande (CLI) EB génère une nouvelle étiquette unique. Si vous fournissez une étiquette de version, assurez-vous qu'elle est unique.  Applicable uniquement avec l'option --create.  Type : chaîne
lifecycle	Appeler l'éditeur par défaut pour créer une stratégie de cycle de vie d'une version de l'application. Utilisez cette stratégie pour éviter d'atteindre le <a href="#">quota de version de l'application</a> .
lifecycle -p ou lifecycle --print	Afficher la stratégie de cycle de vie actuelle de l'application.
-m " <i>version_description</i> " ou --message " <i>version_description</i> "	Description de la version de l'application. Elle est entourée de guillemets doubles.  Applicable uniquement avec l'option --create.  Type : chaîne
-p ou --process	Prétraitez et validez les fichiers de configuration et le manifeste d'environnement dans le groupe source. La validation des fichiers de configuration peut identifier des problèmes. Nous vous recommandons de le faire avant de déployer la version de l'application dans l'environnement.  Applicable uniquement avec l'option --create.

Nom	Description
	Type : chaîne
--source <code>codecommit/<i>repository-name</i>/<i>branch-name</i></code>	Référentiel CodeCommit et branche. Pour plus d'informations, consultez <a href="#">Utilisation de l'interface de ligne de commande EB avec AWS CodeCommit (p. 1048)</a> . Applicable uniquement avec l'option --create.
--staged	Utilisez les fichiers mis en scène dans l'index git, au lieu du commit HEAD, pour créer la version de l'application. Applicable uniquement avec l'option --create.
--timeout <i>minutes</i>	Le nombre de minutes avant que la commande expire. Applicable uniquement avec l'option --create.
<a href="#">Options courantes (p. 1123)</a>	

## Utilisation interactive de la commande

Si vous utilisez la commande sans arguments, la sortie affiche les versions de l'application. Elles sont répertoriées par ordre chronologique inverse, la dernière version étant répertoriée en premier. Consultez la section Examples (Exemples) pour obtenir des exemples de ce à quoi ressemble l'écran. Notez que la ligne d'état s'affiche en bas. La ligne d'état affiche des informations contextuelles.

Appuyez sur **d** pour supprimer une version de l'application, appuyez sur **l** pour gérer la stratégie de cycle de vie de votre application, ou appuyez sur **q** pour quitter sans apporter de modifications.

### Note

Si la version est déployée sur un environnement, vous ne pouvez pas la supprimer.

## Output

La commande avec l'option --create affiche un message qui confirme que la version d'application a été créée.

La commande avec l'option --delete *version-label* affiche un message qui confirme que la version d'application a été supprimée.

## Examples

L'exemple suivant montre la fenêtre interactive pour une application sans déploiement.

```
No Environment Specified
Environment Status: Unknown Health Unknown
Current version # deployed: None

# Version Label Date Created Age Description
3 v4 2016/12/22 13:28 56 secs new features
2 v3 2016/12/22 13:27 1 min important update
1 v1 2016/12/15 23:51 6 days wow

(Commands: Quit, Delete, Lifecycle, ▼▲◀▶)
```

L'exemple suivant montre la fenêtre interactive pour une application avec la quatrième version, avec l'étiquette de version Sample Application (Exemple d'application), déployée.

```
Sample-env
Environment Status: Launching Health Green
Current version # deployed: 4

# Version Label Date Created Age Description
4 Sample Application 2016/12/22 13:30 2 mins -
3 v4 2016/12/22 13:28 4 mins new features
2 v3 2016/12/22 13:27 5 mins important update
1 v1 2016/12/15 23:51 6 days wow

(Commands: Quit, Delete, Lifecycle, ▼▲◀▶)
```

L'exemple suivant illustre la sortie d'une commande eb appversion lifecycle -p, où **ACCOUNT-ID** est l'ID de compte de l'utilisateur :

```
Application details for: lifecycle
Region: sa-east-1
Description: Application created from the EB CLI using "eb init"
Date Created: 2016/12/20 02:48 UTC
Date Updated: 2016/12/20 02:48 UTC
Application Versions: ['Sample Application']
Resource Lifecycle Config(s):
```

```
VersionLifecycleConfig:  
  MaxCountRule:  
    DeleteSourceFromS3: False  
    Enabled: False  
    MaxCount: 200  
  MaxAgeRule:  
    DeleteSourceFromS3: False  
    Enabled: False  
    MaxAgeInDays: 180  
  ServiceRole: arn:aws:iam::ACCOUNT-ID:role/aws-elasticbeanstalk-service-role
```

## eb clone

### Description

Clone un environnement dans un nouvel environnement afin que les deux aient des paramètres d'environnement identiques.

#### Note

Par défaut, quelle que soit la version de pile de solutions de l'environnement à partir de laquelle vous créez le clone, la commande eb clone crée l'environnement clone avec la pile de solutions plus récente. Vous pouvez supprimer cela en incluant l'option --exact lorsque vous exécutez la commande.

### Syntax

eb clone

eb clone *environment-name*

### Options

Nom	Description
<code>-n <i>chaîne</i></code> ou <code>--clone_name <i>chaîne</i></code>	Nom souhaité pour l'environnement cloné.
<code>-c <i>chaîne</i></code> ou <code>--cname <i>chaîne</i></code>	Préfixe CNAME souhaité pour l'environnement cloné.
<code>--envvars</code>	Propriétés d'environnement dans une liste séparée par des virgules au format <code><i>name</i>=<i>value</i></code> .  Type : chaîne  Contraintes : <ul style="list-style-type: none"><li>• Les paires clé-valeur doivent être séparées par des virgules.</li><li>• Les clés et les valeurs peuvent contenir tout caractère alphabétique dans n'importe quelle langue, tout caractère</li></ul>

Nom	Description
	<p>numérique, espace blanc, séparateur invisible et les symboles suivants : _ . : / + \ - @</p> <ul style="list-style-type: none"> <li>Les clés peuvent contenir jusqu'à 128 caractères. Les valeurs peuvent contenir jusqu'à 256 caractères.</li> <li>Les clés et les valeurs sont sensibles à la casse.</li> <li>Les valeurs ne peuvent pas correspondre au nom de l'environnement.</li> <li>Les valeurs ne peuvent pas inclure aws : ni elasticbeanstalk:..</li> <li>La taille totale de toutes les propriétés de l'environnement ne peut pas dépasser 4 096 octets.</li> </ul>
--exact	Empêche Elastic Beanstalk de mettre à jour la version de pile de solutions pour le nouvel environnement clone vers la version la plus récente disponible (pour la plateforme de l'environnement d'origine).
--scale <i>number</i>	Le nombre d'instances à exécuter dans l'environnement clone au moment du lancement.
--tags <i>name=value</i>	Balises (p. 629) pour les ressources dans votre environnement dans une liste séparée par des virgules au format <i>name=value</i> .
--timeout	Le nombre de minutes avant que la commande expire.
Options courantes (p. 1123)	

## Output

En cas d'aboutissement, la commande crée un environnement qui a les mêmes paramètres que l'environnement d'origine ou avec des modifications à l'environnement comme indiqué par toutes les options eb clone.

## Example

L'exemple suivant clone l'environnement spécifié.

```
$ eb clone
Enter name for Environment Clone
(default is tmp-dev-clone):
Enter DNS CNAME prefix
(default is tmp-dev-clone):
Environment details for: tmp-dev-clone
  Application name: tmp
  Region: us-west-2
  Deployed Version: app-141029_144740
  Environment ID: e-vjvrqnn5pv
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev-clone.elasticbeanstalk.com
  Updated: 2014-10-29 22:00:23.008000+00:00
Printing Status:
2018-07-11 21:04:20    INFO: createEnvironment is starting.
2018-07-11 21:04:21    INFO: Using elasticbeanstalk-us-west-2-888888888888 as Amazon S3
                           storage bucket for environment data.
...
2018-07-11 21:07:10    INFO: Successfully launched environment: tmp-dev-clone
```

## eb codesource

### Description

Configure l'interface de ligne de commande (CLI) EB pour [déployer à partir d'un référentiel CodeCommit \(p. 1048\)](#), ou désactive l'intégration CodeCommit et charge le bundle source à partir de votre machine locale.

#### Note

Certaines régions AWS ne proposent pas CodeCommit. L'intégration entre Elastic Beanstalk et CodeCommit ne fonctionne pas dans ces régions.

Pour de plus amples informations sur les services AWS proposés dans chaque région, veuillez consulter le [Tableau des régions](#).

### Syntax

eb codesource

eb codesource codecommit

eb codesource local

### Options

Nom	Description
<a href="#">Options courantes (p. 1123)</a>	

### Output

eb codesource vous invite à choisir entre une intégration CodeCommit et les déploiements standard.

eb codesource codecommit démarre la configuration interactive du référentiel pour l'intégration CodeCommit.

eb codesource local affiche la configuration d'origine et désactive l'intégration CodeCommit.

### Examples

Utilisez eb codesource codecommit pour configurer l'intégration CodeCommit pour la branche actuelle.

```
~/my-app$ eb codesource codecommit
Select a repository
1) my-repo
2) my-app
3) [ Create new Repository ]
(default is 1): 1

Select a branch
1) mainline
2) test
3) [ Create new Branch with local HEAD ]
(default is 1): 1
```

Utilisez eb codesource local pour configurer l'intégration CodeCommit pour la branche actuelle.

```
~/my-app$ eb codesource local
Current CodeCommit setup:
  Repository: my-app
  Branch: mainline
Default set to use local sources
```

## eb config

### Description

Gère les paramètres [de configuration \(p. 14\)](#) actifs et les [configurations enregistrées \(p. 15\)](#) de votre environnement. Vous pouvez utiliser cette commande pour charger, télécharger ou répertorier les configurations enregistrées de votre environnement. Vous pouvez également l'utiliser pour télécharger, afficher ou mettre à jour ses paramètres de configuration actifs.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande modifie également les paramètres de configuration du générateur. Cela se fait en fonction des valeurs définies dans `platform.yaml`.

#### Note

eb config n'affiche pas les propriétés de l'environnement. Pour définir les propriétés de l'environnement que vous pouvez lire à partir de votre application, utilisez [eb setenv \(p. 676\)](#) à la place.

### Syntax

Les éléments suivants font partie de la syntaxe utilisée pour que la commande eb config fonctionne avec les [paramètres de configuration \(p. 14\)](#) actifs de votre environnement. Pour obtenir des exemples spécifiques, consultez la section [Examples \(p. 1075\)](#) plus loin dans cette rubrique.

- eb config – Affiche les paramètres de configuration actifs de votre environnement dans un éditeur de texte que vous avez configuré en tant que variable d'environnement EDITOR. Lorsque vous enregistrez des modifications dans le fichier et que vous fermez l'éditeur, l'environnement est mis à jour avec les paramètres d'option que vous avez enregistrés dans le fichier.

#### Note

Si vous n'avez pas configuré de variable d'environnement EDITOR, l'interface de ligne de commande (CLI) EB affiche vos paramètres d'options dans votre éditeur par défaut pour les fichiers YAML.

- eb config **environment-name** – Affiche et met à jour la configuration de l'environnement nommé. La configuration s'affiche soit dans un éditeur de texte que vous avez configuré, soit dans les fichiers YAML de votre éditeur par défaut.
- eb config save – Enregistre les paramètres de configuration actifs de l'environnement actuel dans `.elasticbeanstalk/saved_configs/` avec le nom de fichier `[configuration-name].cfg.yml`. Par défaut, l'interface de ligne de commande (CLI) EB enregistre les paramètres de configuration avec un **configuration-name** basé sur le nom de l'environnement. Vous pouvez spécifier un nom de configuration différent en incluant l'option `--cfg` avec votre nom de la configuration souhaité lorsque vous exécutez la commande.

Vous pouvez baliser votre configuration enregistrée à l'aide de l'option `--tags`.

- eb config **--display** – Écrit les paramètres de configuration actifs d'un environnement dans `stdout` au lieu d'un fichier. Par défaut, cela affiche les paramètres de configuration dans le terminal.

- **eb config --update *configuration\_string* | *file\_path***– Met à jour les paramètres de configuration actifs de l'environnement actuel avec les informations spécifiées dans *configuration\_string* ou dans le fichier identifié par *file\_path*.

#### Note

Les options `--display` et `--update` offrent une flexibilité pour lire et réviser les paramètres de configuration d'un environnement par programmation.

Ce qui suit décrit la syntaxe pour l'utilisation de la commande eb config permettant de travailler avec des [configurations enregistrées \(p. 15\)](#). Pour obtenir des exemples, consultez la section [Examples \(p. 1075\)](#) plus loin dans cette rubrique.

- **eb config get *config-name***– Télécharge la configuration enregistrée nommée depuis Amazon S3.
- **eb config delete *config-name*** – Supprime la configuration enregistrée nommée d'Amazon S3. Elle la supprime également localement, si vous l'avez déjà téléchargée.
- **eb config list** – Répertorie les configurations enregistrées que vous avez dans Amazon S3.
- **eb config put *filename*** – Télécharge la nommée configuration enregistrée nommée dans un compartiment Amazon S3. Le *nom de fichier* doit avoir l'extension de fichier `.cfg.yml`. Pour spécifier le nom du fichier sans chemin d'accès, vous pouvez enregistrer le fichier dans le dossier `.elasticbeanstalk` ou dans le dossier `.elasticbeanstalk/saved_configs/` avant d'exécuter la commande. Vous pouvez sinon spécifier le *nom de fichier* en fournissant le chemin d'accès complet.

## Options

Nom	Description
<code>--cfg <i>config-name</i></code>	Nom à utiliser pour une configuration enregistrée.  Cette option fonctionne avec eb config save uniquement.
<code>-d</code> ou <code>--display</code>	Affiche les paramètres de configuration de l'environnement actuel (écrit dans <code>stdout</code> ).  Utilisez l'option <code>--format</code> pour spécifier que la sortie doit être au format JSON ou YAML. Si vous ne le spécifiez pas, la sortie est au format YAML.  Cette option ne fonctionne que si vous utilisez la commande eb config sans aucune autre sous-commande.
<code>-f <i>format_type</i></code> ou <code>--format <i>format_type</i></code>	Spécifie le format d'affichage. Les valeurs valides sont JSON ou YAML.  La valeur par défaut est YAML.  Cette option fonctionne uniquement avec l'option <code>--display</code> .
<code>--tags <i>key1=value1[,key2=value2]</i></code>	Les balises à ajouter à votre configuration enregistrée. Lorsque vous spécifiez des balises dans la liste, spécifiez-les en tant que paires clé=valeur et séparez-les par une virgule.  Pour plus d'informations, consultez <a href="#">Balisage de configurations enregistrées (p. 783)</a> .

Nom	Description
	Cette option fonctionne avec eb config save uniquement.
--timeout <i>timeout</i>	Le nombre de minutes avant que la commande expire.

Nom	Description
<code>-u configuration_string   file_path</code>	Met à jour les paramètres de configuration actifs de l'environnement actuel.
ou	Cette option ne fonctionne que si vous utilisez la commande eb config sans aucune autre sous-commande.
<code>--update configuration_string   file_path</code>	<p>Le paramètre <code>configuration_string</code>   <code>file_path</code> est de type chaîne. La chaîne fournit la liste des espaces de noms et des options correspondantes pour ajouter, mettre à jour ou supprimer des paramètres de configuration de votre environnement. La chaîne d'entrée peut également représenter un fichier contenant les mêmes informations.</p> <p>Pour spécifier un nom de fichier, la chaîne d'entrée doit suivre le format "file://&lt;path&gt;&lt;filename&gt;". Pour spécifier le nom de fichier sans un <code>path</code>, enregistrez-le dans le dossier dans lequel vous exécutez la commande. Vous pouvez sinon spécifier le nom de fichier en fournissant le chemin d'accès complet.</p> <p>Les informations de configuration doivent répondre aux conditions suivantes. Au moins une des sections, OptionSettings ou OptionsToRemove, est requise. Utilisez OptionSettings pour ajouter ou modifier des options. Utilisez OptionsToRemove pour supprimer des options d'un espace de noms. Pour obtenir des exemples spécifiques, consultez la section <a href="#">Examples (p. 1075)</a> plus loin dans cette rubrique.</p>
	<p><b>Example</b></p> <p><b>Format YAML</b></p> <pre>OptionSettings:   namespace1:     option-name-1: option-value-1     option-name-2: option-value-2     ... OptionsToRemove:   namespace1:     option-name-1     option-name-2     ...</pre>
	<p><b>Example</b></p> <p><b>Format JSON</b></p> <pre>{   "OptionSettings": {     "namespace1": {       "option-name-1": "option-value-1",       "option-name-2": "option-value-2",       ...     }   },   "OptionsToRemove": {     "namespace1": {       "option-name-1":</pre>

Nom	Description
	<pre>"option-name-2", ... } }</pre>
<a href="#">Options courantes (p. 1123)</a>	

## Output

Si la commande eb config ou eb config **environment-name** est exécutée avec succès sans ajout d'une sous-commande ou d'une option, la commande affiche vos paramètres d'options actuels dans l'éditeur de texte que vous avez configuré en tant que variable d'environnement EDITOR. Si vous n'avez pas configuré de variable d'environnement EDITOR, l'interface de ligne de commande (CLI) EB affiche vos paramètres d'options dans votre éditeur par défaut pour les fichiers YAML.

Lorsque vous enregistrez des modifications dans le fichier et que vous fermez l'éditeur, l'environnement est mis à jour avec les paramètres d'option que vous avez enregistrés dans le fichier. La sortie suivante s'affiche pour confirmer la mise à jour de la configuration.

```
$ eb config myApp-dev
Printing Status:
2021-05-19 18:09:45    INFO    Environment update is starting.
2021-05-19 18:09:55    INFO    Updating environment myApp-dev's configuration settings.
2021-05-19 18:11:20    INFO    Successfully deployed new configuration to environment.
```

Si la commande s'exécute avec succès avec l'option --display, elle affiche les paramètres de configuration de l'environnement actuel (écrit dansstdout).

Si la commande s'exécute avec succès avec le paramètre get, la commande affiche l'emplacement de la copie locale que vous avez téléchargée.

Si la commande s'exécute avec succès avec le paramètre save, la commande affiche l'emplacement du fichier enregistré.

## Examples

Cette section décrit comment modifier l'éditeur de texte que vous utilisez pour afficher et modifier votre fichier de paramètres d'option.

Pour Linux et UNIX, l'exemple suivant remplace l'éditeur par vim :

```
$ export EDITOR=vim
```

Pour Linux et UNIX, l'exemple suivant remplace l'éditeur par ce qui est installé à l'emplacement /usr/bin/kate.

```
$ export EDITOR=/usr/bin/kate
```

Pour Windows, l'exemple suivant remplace l'éditeur par Notepad++.

```
> set EDITOR="C:\Program Files\Notepad++\Notepad++.exe
```

Cette section fournit des exemples de la commande eb config lorsqu'elle est exécutée avec des sous-commandes.

L'exemple suivant supprime la configuration enregistrée nommée app-tmp.

```
$ eb config delete app-tmp
```

L'exemple suivant télécharge la configuration enregistrée avec le nom app-tmp depuis votre compartiment Amazon S3.

```
$ eb config get app-tmp
```

L'exemple suivant répertorie les noms des configurations enregistrées qui sont stockées dans votre compartiment Amazon S3.

```
$ eb config list
```

L'exemple suivant télécharge la copie locale de la configuration enregistrée nommée app-tmp dans votre compartiment Amazon S3.

```
$ eb config put app-tmp
```

L'exemple suivant enregistre les paramètres de configuration de l'environnement en cours d'exécution actuel. Si vous ne fournissez pas un nom à utiliser pour la configuration enregistrée, alors Elastic Beanstalk nomme le fichier de configuration en fonction du nom de l'environnement. Par exemple, un environnement nommé tmp-dev serait appelé tmp-dev.cfg.yml. Elastic Beanstalk enregistre le fichier dans le dossier /.elasticbeanstalk/saved\_configs/.

```
$ eb config save
```

Dans l'exemple suivant, l'option --cfg est utilisée pour enregistrer les paramètres de configuration de l'environnement tmp-dev dans un fichier appelé v1-app-tmp.cfg.yml. Elastic Beanstalk enregistre le fichier dans le dossier /.elasticbeanstalk/saved\_configs/. Si vous ne spécifiez pas un nom d'environnement, Elastic Beanstalk enregistre les paramètres de configuration de l'environnement en cours d'exécution actuel.

```
$ eb config save tmp-dev --cfg v1-app-tmp
```

Cette section fournit des exemples de commande eb config lorsqu'elle est exécutée sans sous-commande.

La commande suivante affiche les paramètres d'options de votre environnement actuel dans un éditeur de texte.

```
$ eb config
```

La commande suivante affiche les paramètres d'options de l'environnement my-env dans un éditeur de texte.

```
$ eb config my-env
```

L'exemple suivant affiche les paramètres d'options de votre environnement actuel. Il sort au format YAML car aucun format spécifique n'a été spécifié avec l'option --format.

```
$ eb config --display
```

L'exemple suivant met à jour les paramètres d'options de votre environnement actuel avec les spécifications du fichier nommé example.txt. Le fichier a un format YAML ou JSON. L'interface de ligne de commande (CLI) EB détecte automatiquement le format de fichier.

- L'option Minsize est définie sur 1 pour l'espace de noms aws:autoscaling:asg.
- La taille du lot de l'espace de noms aws:elasticbeanstalk:command est définie sur 30 %.
- Cela supprime le paramètre d'option IdleTimeout: None de l'espace de noms AWSEBV2LoadBalancer.aws:elbv2:loadbalancer.

```
$ eb config --update "file://example.txt"
```

Example - nom de fichier :**example.txt** - format YAML

```
OptionSettings:  
  'aws:elasticbeanstalk:command':  
    BatchSize: '30'  
    BatchSizeType: Percentage  
  'aws:autoscaling:asg':  
    MinSize: '1'  
OptionsToRemove:  
  'AWSEBV2LoadBalancer.aws:elbv2:loadbalancer':  
    IdleTimeout
```

Example - nom de fichier :**example.txt** - format JSON

```
{  
  "OptionSettings": {  
    "aws:elasticbeanstalk:command": {  
      "BatchSize": "30",  
      "BatchSizeType": "Percentage"  
    },  
    "aws:autoscaling:asg": {  
      "MinSize": "1"  
    }  
  },  
  "OptionsToRemove": {  
    "AWSEBV2LoadBalancer.aws:elbv2:loadbalancer": {  
      "IdleTimeout"  
    }  
  }  
}
```

Les exemples suivants mettent à jour les paramètres d'options de votre environnement actuel. La commande définit l'option Minsize sur 1 pour l'espace de noms aws:autoscaling:asg.

#### Note

Ces exemples sont spécifiques à Windows PowerShell. Ils échappent aux occurrences littérales du caractère double guillemet ("") en le faisant précéder d'une barre oblique (\). Différents systèmes d'exploitation et environnements de ligne de commande peuvent comporter des séquences d'échappement différentes. Pour cette raison, nous vous recommandons d'utiliser l'option de fichier reprise dans les exemples précédents. La spécification des options de configuration dans un fichier ne nécessite pas l'échappement de caractères et est cohérente entre différents systèmes d'exploitation.

L'exemple suivant est au format JSON. L'interface de ligne de commande (CLI) EB détecte si le format est en JSON ou YAML.

```
PS C:\Users\myUser\EB_apps\myApp-env>eb config --update '{\"OptionSettings\":\n  {\"aws:autoscaling:asg\":{\"MaxSize\":\"1\"}}}'
```

L'exemple suivant est au format YAML. Pour entrer la chaîne YAML au format correct, la commande inclut l'espacement et les retours de fin de ligne requis dans un fichier YAML.

- Terminez chaque ligne avec la touche « Entrée » ou « retour ».
- Commencez la deuxième ligne par deux espaces et commencez la troisième ligne par quatre espaces.

```
PS C:\Users\myUser\EB_apps\myApp-env>eb config --update 'OptionSettings:  
>> aws:autoscaling:asg:  
>> MinSize: \"1\"'
```

## eb console

### Description

Ouvre un navigateur pour afficher le tableau de bord de configuration de l'environnement d'Elastic Beanstalk Management Console.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande affiche également la configuration de l'environnement de génération, telle qu'indiquée dans `platform.yaml`, dans Elastic Beanstalk Management Console.

### Syntax

`eb console`

`eb console environment-name`

### Options

Nom	Description
<a href="#">Options courantes (p. 1123)</a>	

## eb create

### Description

Crée un environnement et y déploie une version de l'application.

#### Note

- Pour utiliser `eb create` sur une application .NET, vous devez créer un package de déploiement comme décrit dans [Création d'une solution groupée source pour une application .NET \(p. 421\)](#), puis définir la configuration de l'interface de ligne de commande (CLI) pour déployer le package sous forme d'artefact comme décrit dans [Déploiement d'un artefact à la place du dossier de projet \(p. 1039\)](#).
- Créer des environnements avec l'interface de ligne de commande (CLI) EB nécessite un [rôle de service \(p. 21\)](#). Vous pouvez créer un rôle de service en créant un environnement dans la console Elastic Beanstalk. Si vous n'avez pas de rôle de service, l'interface de ligne de commande (CLI) EB essaie d'en créer un lorsque vous exécutez `eb create`.

Vous pouvez déployer la version de l'application à partir de différentes sources :

- Par défaut : depuis le code source de l'application dans le répertoire de projet local.
- À l'aide de l'option `--version` : à partir d'une version de l'application qui existe déjà dans votre application.
- Lorsque votre projet de répertoire ne dispose pas d'un code d'application ou que vous utilisez l'option `--sample` : déployée à partir d'un exemple d'application spécifique à la plateforme de votre environnement.

## Syntax

`eb create`

`eb create environment-name`

La longueur d'un nom d'environnement doit être comprise entre 4 et 40 caractères. Il ne peut contenir que des lettres, des chiffres et des traits d'union (-). Un nom d'environnement ne peut pas commencer ou se terminer par un trait d'union.

Si vous incluez un nom d'environnement dans la commande, l'interface de ligne de commande (CLI) EB ne vous invite pas à sélectionner ou créer un rôle de service.

Si vous exécutez la commande sans argument de nom d'environnement, elle s'exécute dans un flux interactif et vous invite à saisir ou sélectionner des valeurs pour certains paramètres. Dans ce flux interactif, si vous déployez un exemple d'application, l'interface de ligne de commande (CLI) EB vous demande aussi si vous souhaitez télécharger cet exemple d'application sur votre répertoire de projet local. En la téléchargeant, vous pouvez utiliser l'interface de ligne de commande (CLI) EB avec le nouvel environnement pour exécuter des opérations qui nécessitent le code de l'application, comme [eb deploy \(p. 1089\)](#).

Certaines invites de flux interactives ne sont affichées que dans certaines conditions. Par exemple, si vous choisissez d'utiliser un équilibrEUR de charge Application Load Balancer et que votre compte dispose d'au moins un équilibrEUR de charge Application Load Balancer partageable, Elastic Beanstalk affiche une invite qui vous demande si vous souhaitez utiliser un équilibrEUR de charge partagé. Si aucun équilibrEUR de charge Application Load Balancer partageable n'existe dans votre compte, cette invite ne s'affiche pas.

## Options

Aucune de ces options n'est obligatoire. Si vous exécutez `eb create` sans spécifier d'options, l'interface de ligne de commande (CLI) EB vous invite à entrer ou à sélectionner une valeur pour chaque paramètre.

Nom	Description
<code>-d</code> ou <code>--branch_default</code>	Définissez l'environnement en tant qu'environnement par défaut pour le référentiel actuel.
<code>--cfg <b>config-name</b></code>	Utilisez des paramètres de plateforme à partir d'une configuration enregistrée (p. 668) dans <code>.elasticbeanstalk/saved_configs/</code> ou votre compartiment Amazon S3. Spécifiez le nom du fichier uniquement, sans l'extension <code>.cfg.yml</code> .
<code>-c <b>subdomain-name</b></code> ou <code>--cname <b>subdomain-name</b></code>	Nom du sous-domaine devant préfixer l'entrée DNS CNAME qui permet d'accéder à votre site web.  Type : chaîne  Par défaut : le nom de l'environnement

Nom	Description
-db ou --database	Attache une base de données à l'environnement. Si vous exécutez eb create avec l'option --database, mais sans les options --database.username et --database.password, l'interface de ligne de commande (CLI) EB vous invite à saisir le nom et le mot de passe de l'utilisateur principal de la base de données.
-db.engine <i>engine</i> ou --database.engine <i>engine</i>	<p>Le type de moteur de base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée. C'est le cas même si vous n'avez pas exécuté la commande avec l'option --database.</p> <p>Type : chaîne</p> <p>Valeurs valides: mysql, oracle-se1, postgres, sqlserver-ex, sqlserver-web, sqlserver-se</p>
-db.i <i>instance_type</i> ou --database.instance <i>instance_type</i>	<p>Le type d'instance Amazon EC2 à utiliser pour la base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée. C'est le cas même si vous n'avez pas exécuté la commande avec l'option --database.</p> <p>Type : chaîne</p> <p>Valeurs valides : consultez <a href="#">Valeurs d'option</a>.</p>
-db.pass <i>password</i> ou --database.password <i>password</i>	Le mot de passe pour la base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée. C'est le cas même si vous n'avez pas exécuté la commande avec --database.
-db.size <i>number_of_gigabytes</i> ou --database.size <i>number_of_gigabytes</i>	<p>Le nombre de gigaoctets (Go) à allouer pour le stockage de base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée. C'est le cas même si vous n'avez pas exécuté la commande avec l'option --database.</p> <p>Type : nombre</p> <p>Valeurs valides :</p> <ul style="list-style-type: none"> <li>• MySQL – 5 à 1024. La valeur par défaut est 5.</li> <li>• Postgres – 5 sur 1024. La valeur par défaut est 5.</li> <li>• Oracle – 10 à 1024. La valeur par défaut est 10.</li> <li>• Microsoft SQL Server Express Edition – 30.</li> <li>• Microsoft SQL Server Web Edition – 30.</li> <li>• Microsoft SQL Server Standard Edition – 200.</li> </ul>

Nom	Description
<code>-db.user <i>username</i></code> ou <code>--database.username <i>username</i></code>	Le nom utilisateur pour la base de données. Si vous exécutez eb create avec cette option, alors l'interface de ligne de commande (CLI) EB lance l'environnement avec une base de données attachée même si vous n'avez pas exécuté la commande avec l'option --database. Si vous exécutez eb create avec l'option --database, mais sans les options --database.username et --database.password, alors l'interface de ligne de commande (CLI) EB vous invite pour le mot de passe et le nom utilisateur de base de données principale.
<code>-db.version <i>Version</i></code> ou <code>--database.version <i>Version</i></code>	Utilisé pour spécifier la version du moteur de base de données. Si cet indicateur est présent, l'environnement se lance avec une base de données avec le numéro de version spécifié, même si l'indicateur --database n'est pas présent.
<code>--elb-type <i>type</i></code>	Type de l'équilibrer de charge (p. 562).  Type : chaîne  Valeurs valides: <code>classic</code> , <code>application</code> , <code>network</code>  Par défaut: <code>application</code>
<code>-es</code> ou <code>--enable-spot</code>	Activez les demandes d'instances Spot pour votre environnement. Pour plus d'informations, consultez Groupe Auto Scaling (p. 547).  Options connexes : <ul style="list-style-type: none"><li>• <code>--instance-types</code></li><li>• <code>--on-demand-base-capacity</code></li><li>• <code>--on-demand-above-base-capacity</code></li><li>• <code>--spot-max-price</code></li></ul>
<code>--env-group-suffix <i>groupname</i></code>	Le nom du groupe à ajouter au nom de l'environnement. A utiliser uniquement avec Compose Environments (p. 1058).
<code>--envvars</code>	Propriétés d'environnement (p. 632) dans une liste séparée par des virgules au format <code>name=value</code> . Pour connaître les limites, consultez Configuration des propriétés de l'environnement (p. 634).
<code>-ip <i>profile_name</i></code> ou <code>--instance_profile <i>profile_name</i></code>	Le profil d'instance avec le rôle IAM avec les informations d'identification de sécurité temporaires dont votre application a besoin pour accéder aux ressources AWS.

Nom	Description
<p>-it ou --instance-types <code>type1[, type2 ...]</code></p>	<p>Liste séparée par des virgules des types d'instance Amazon EC2 que votre environnement doit utiliser. Si vous ne spécifiez pas cette option, Elastic Beanstalk fournit les types d'instance par défaut.</p> <p>Pour de plus amples informations, veuillez consulter <a href="#">Instances Amazon EC2 (p. 538)</a> et <a href="#">Groupe Auto Scaling (p. 547)</a>.</p> <p><b>Important</b></p> <p>L'interface de ligne de commande (CLI) EB applique uniquement cette option aux instances Spot. À moins que cette option soit utilisée avec l'option --enable-spot, l'interface de ligne de commande (CLI) EB l'ignore. Pour spécifier un type d'instance pour une instance à la demande, utilisez l'option --instance-type (pas de « s ») à la place.</p>
<p>-i ou --instance_type</p>	<p>Type d'instance Amazon EC2 que votre environnement doit utiliser. Si vous ne spécifiez pas cette option, Elastic Beanstalk fournit un type d'instance par défaut.</p> <p>Pour plus d'informations, consultez <a href="#">Instances Amazon EC2 (p. 538)</a>.</p> <p><b>Important</b></p> <p>L'interface de ligne de commande (CLI) EB applique uniquement cette option aux instances à la demande. N'utilisez pas cette option avec l'option --enable-spot, car l'interface de ligne de commande (CLI) EB l'ignore lorsque vous le faites. Pour spécifier des types d'instances pour une instance Spot, utilisez l'option --instance-types (avec un « s ») à la place.</p>
<p>-k <code>key_name</code> ou --keyname <code>key_name</code></p>	<p>Nom de la paire de clés Amazon EC2 à utiliser avec le client Secure Shell (SSH) pour se connecter en toute sécurité aux instances Amazon EC2 qui exécutent votre application Elastic Beanstalk. Si vous incluez cette option avec la commande eb create, la valeur que vous fournissez remplace n'importe quel nom de clé que vous pourriez avoir spécifié avec eb init.</p> <p>Valeurs valides : un nom de clé existant qui est enregistré avec Amazon EC2</p>
<p>-im <code>number-of-instances</code> ou --min-instances <code>number-of-instances</code></p>	<p>Le nombre minimum d'instances Amazon EC2 dont vous avez besoin pour votre environnement.</p> <p>Type : Nombre (entier)</p> <p>Par défaut: 1</p> <p>Valeurs valides : de 1 à 10000</p>

Nom	Description
<b>-ix <i>number-of-instances</i></b> ou <b>--max-instances <i>number-of-instances</i></b>	Le nombre maximum d'instances Amazon EC2 que vous autorisez votre environnement à posséder. Type : Nombre (entier) Par défaut: 4 Valeurs valides : de 1 à 10000
<b>--modules <i>component-a component-b</i></b>	Une liste des environnements de composants à créer. Cela doit être utilisé uniquement avec <a href="#">Compose Environments (p. 1058)</a> (Composer des environnements).
<b>-sb</b> ou <b>--on-demand-base-capacity</b>	Nombre minimal d'instances à la demande que votre groupe Auto Scaling alloue avant de considérer les instances Spot à mesure que votre environnement augmente. Cette option ne peut être spécifiée qu'avec l'option <b>--enable-spot</b> . Pour plus d'informations, consultez <a href="#">Groupe Auto Scaling (p. 547)</a> . Type : Nombre (entier) Par défaut: 0 Valeurs valables : 0 à <b>--max-instances</b> (en son absence : <b>MaxSize</b> option dans l'espace de noms <a href="#">aws:autoscaling:asg (p. 680)</a> )
<b>-sp</b> ou <b>--on-demand-above-base-capacity</b>	Pourcentage d'instances à la demande dans le cadre de la capacité supplémentaire allouée par votre groupe Auto Scaling supérieur au nombre d'instances qui est spécifié par l'option <b>--on-demand-base-capacity</b> . Cette option ne peut être spécifiée qu'avec l'option <b>--enable-spot</b> . Pour en savoir plus, consultez <a href="#">Groupe Auto Scaling (p. 547)</a> . Type : Nombre (entier) Par défaut : 0 pour un environnement à instance unique ; 70 pour un environnement à charge équilibrée Valeurs valides : de 0 à 100

Nom	Description
<p>-p <i>platform-version</i> ou --platform <i>platform-version</i></p>	<p>La <a href="#">version de la plateforme (p. 31)</a> à utiliser. Vous pouvez spécifier un nom de plateforme, une plateforme et une version de plateforme, une branche de plateforme, un nom de pile de solutions ou un ARN de pile de solutions. Exemples :</p> <ul style="list-style-type: none"> <li>• php, PHP, node.js – Dernière version pour la plateforme spécifiée</li> <li>• php-7.2, "PHP 7.2" – Version recommandée (généralement la plus récente) de la plateforme PHP 7.2</li> <li>• "PHP 7.2 running on 64bit Amazon Linux" – Version recommandée (généralement la plus récente) de la plateforme PHP dans cette branche de plateforme</li> <li>• "64bit Amazon Linux 2017.09 v2.6.3 running PHP 7.1" – Version de la plateforme PHP spécifiée par ce nom de pile de solutions</li> <li>• "arn:aws:elasticbeanstalk:us-east-2::platform/PHP 7.1 running on 64bit Amazon Linux/2.6.3" – Version de la plateforme PHP spécifiée par cet ARN de pile de solutions</li> </ul> <p>Utilisez <a href="#">eb platform list (p. 1103)</a> pour obtenir la liste des configurations disponibles.</p> <p>Si vous spécifiez l'option --platform, elle remplace la valeur fournie au cours d'eb init.</p>
<p>-pr ou --process</p>	Prétraitez et validez les fichiers de configuration et le manifeste d'environnement dans le groupe source. La validation des fichiers de configuration permet d'identifier les problèmes avant de déployer la version de l'application dans un environnement.
<p>-r <i>région</i> ou --region <i>région</i></p>	<p>La région AWS dans laquelle vous souhaitez déployer l'application.</p> <p>Pour obtenir la liste des valeurs que vous pouvez spécifier pour cette option, consultez <a href="#">Points de terminaison et quotas AWS Elastic Beanstalk</a> dans la section Référence générale AWS.</p>
--sample	Déployez l'exemple d'application dans le nouvel environnement au lieu du code dans votre référentiel.
--scale <i>number-of-instances</i>	Démarrez avec le nombre d'instances spécifié
--service-role <i>servicerole</i>	<p>Attribuez un rôle de service autre que celui par défaut à l'environnement.</p> <p><b>Note</b></p> <p>Ne saisissez pas d'ARN. Saisissez uniquement le nom du rôle. Elastic Beanstalk fait précéder le nom de rôle des valeurs correctes pour créer l'ARN en interne.</p>

Nom	Description
<p><b>-ls <i>équilibrEUR de charge</i></b> ou <b>--shared-lb <i>équilibrEUR de charge</i></b></p>	<p>Configurez l'environnement pour utiliser un équilibrEUR de charge partagé. Indiquez le nom ou l'ARN d'un équilibrEUR de charge partageable dans votre compte, un équilibrEUR de charge Application Load Balancer que vous avez créé explicitement, et non pas un autre environnement Elastic Beanstalk. Pour plus d'informations, consultez <a href="#">ÉquilibrEUR de charge Application Load Balancer partagé (p. 591)</a>.</p> <p>Exemples de paramètres :</p> <ul style="list-style-type: none"> <li>• <code>FrontEndLB</code> – Nom de l'équilibrEUR de charge</li> <li>• <code>arn:aws:elasticloadbalancing:us-east-2:123456789012:loadbalancer/app/FrontEndLB/0dbf78d8ad96abbc</code> – ARN de l'équilibrEUR de charge Application Load Balancer.</li> </ul> <p>Vous pouvez spécifier cette option uniquement avec <code>--elb-type application</code>. Si vous spécifiez cette option et que vous ne spécifiez pas <code>--shared-lb</code>, Elastic Beanstalk crée un équilibrEUR de charge dédié pour l'environnement.</p>
<p><b>-lp <i>port</i></b> ou <b>--shared-lb-port <i>port</i></b></p>	<p>Port d'écoute par défaut de l'équilibrEUR de charge partagé pour cet environnement. Elastic Beanstalk ajoute une règle d'écoute qui achemine tout le trafic de ce processus d'écoute vers le processus d'environnement par défaut. Pour plus d'informations, consultez <a href="#">ÉquilibrEUR de charge Application Load Balancer partagé (p. 591)</a>.</p> <p>Type : Nombre (entier)</p> <p>Par défaut: 80</p> <p>Valeurs valides : entier représentant un port d'écoute de l'équilibrEUR de charge partagé.</p>
<b>--single</b>	<p>Créez l'environnement avec une seule instance Amazon EC2 et sans équilibrEUR de charge.</p> <p><b>Warning</b></p> <p>Un environnement d'instance unique n'est pas prêt pour la production. Si l'instance devient instable lors du déploiement, ou si Elastic Beanstalk résilie l'instance lors d'une mise à jour de configuration, votre application peut être indisponible pendant un certain temps. Utilisez des environnements d'instance unique pour le développement ou les tests, ou comme environnement intermédiaire. Utilisez des environnements à charge équilibrée pour la production.</p>

Nom	Description
-sm ou --spot-max-price	Prix maximum par heure d'unité, en dollars américains, que vous êtes prêt à payer pour une instance Spot.  Cette option ne peut être spécifiée qu'avec l'option --enable-spot. Pour en savoir plus, consultez <a href="#">Groupe Auto Scaling (p. 547)</a> .  Type : nombre (flottant)  Par défaut : prix à la demande, par type d'instance. La valeur de l'option dans ce cas est <code>null</code> .  Valeurs valides : de 0.001 à 20.0  Pour des recommandations sur les options tarifaires maximales pour les instances Spot, consultez <a href="#">Historique de tarification d'instances Spot</a> dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.
--tags <code>key1=value1[,key2=value2]</code>	Ajouter des balises aux ressources de votre environnement. Les balises sont spécifiées sous la forme d'une liste séparée par des virgules et composée de paires key=value.  Pour plus d'informations, consultez <a href="#">Balisage des environnements (p. 629)</a> .
-t worker ou --tier worker	Créez un environnement de travail. Ignorez cette option pour créer un environnement de serveur web.
--timeout <code>minutes</code>	Définissez le nombre de minutes avant que la commande expire.
--version <code>version_label</code>	Spécifie la version de l'application que vous voulez voir déployée dans l'environnement à la place du code source d'application dans le répertoire du projet local.  Type : chaîne  Valeurs valides : une étiquette de version de l'application existante
--vpc	Configurez un VPC pour votre environnement. Lorsque vous incluez cette option, l'interface de ligne de commande (CLI) EB vous invite à entrer tous les paramètres obligatoires avant de lancer l'environnement.
--vpc.dbsubnets <code>subnet1,subnet2</code>	Spécifie les sous-réseaux pour les instances de base de données dans un VPC. Obligatoire lorsque --vpc.id est spécifié.
--vpc.ec2subnets <code>subnet1,subnet2</code>	Spécifie des sous-réseaux pour les instances Amazon EC2 dans un VPC. Obligatoire lorsque --vpc.id est spécifié.

Nom	Description
--vpc.elbpublic	Lance votre équilibrer de charge Elastic Load Balancing dans un sous-réseau public de votre VPC.  Vous ne pouvez pas spécifier cette option avec les options --tier worker ou --single.
--vpc.elbsubnets <i>subnet1, subnet2</i>	Spécifie des sous-réseaux pour l'équilibrer de charge Elastic Load Balancing dans un VPC.  Vous ne pouvez pas spécifier cette option avec les options --tier worker ou --single.
--vpc.id <i>ID</i>	Lance votre environnement dans le VPC spécifié.
--vpc.publicip	Lance vos instances Amazon EC2 dans un sous-réseau public dans votre VPC.  Vous ne pouvez pas spécifier cette option avec l'option --tier worker.
--vpc.securitygroups <i>securitygroup1, securitygroup2</i>	Spécifie les ID des groupes de sécurité. Obligatoire lorsque --vpc.id est spécifié.
Options courantes (p. 1123)	

## Output

En cas de succès, la commande vous soumet des questions, puis renvoie le statut de l'opération de création. Si des problèmes se sont produits lors du lancement, vous pouvez utiliser l'opération [eb events \(p. 1090\)](#) pour obtenir plus d'informations.

Si vous avez activé la prise en charge de CodeBuild dans votre application, eb create affiche les informations de CodeBuild au fur et à mesure de la création du code. Pour de plus amples informations sur la prise en charge de CodeBuild dans Elastic Beanstalk, veuillez consulter [Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild \(p. 1044\)](#).

## Examples

L'exemple suivant crée un environnement en mode interactif.

```
$ eb create
Enter Environment Name
(default is tmp-dev): ENTER
Enter DNS CNAME prefix
(default is tmp-dev): ENTER
Select a load balancer type
1) classic
2) application
3) network
(default is 2): ENTER
Environment details for: tmp-dev
Application name: tmp
Region: us-east-2
Deployed Version: app-141029_145448
Environment ID: e-um3yfrzq22
Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
Tier: WebServer-Standard-1.0
```

```
CNAME: tmp-dev.elasticbeanstalk.com
Updated: 2014-10-29 21:54:51.063000+00:00
Printing Status:
...
```

L'exemple suivant crée aussi un environnement en mode interactif. Dans cet exemple, votre projet de répertoire ne dispose pas d'un code d'application. La commande déploie un exemple d'application et le télécharge dans votre répertoire de projet local.

```
$ eb create
Enter Environment Name
(default is tmp-dev): ENTER
Enter DNS CNAME prefix
(default is tmp-dev): ENTER
Select a load balancer type
1) classic
2) application
3) network
(default is 2): ENTER
NOTE: The current directory does not contain any source code. Elastic Beanstalk is
launching the sample application instead.
Do you want to download the sample application into the current directory?
(Y/n): ENTER
INFO: Downloading sample application to the current directory.
INFO: Download complete.
Environment details for: tmp-dev
Application name: tmp
Region: us-east-2
Deployed Version: Sample Application
Environment ID: e-um3yfrzq22
Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
Tier: WebServer-Standard-1.0
CNAME: tmp-dev.elasticbeanstalk.com
Updated: 2017-11-08 21:54:51.063000+00:00
Printing Status:
...
```

La commande suivante crée un environnement sans afficher d'invités.

```
$ eb create dev-env
Creating application version archive "app-160312_014028".
Uploading test/app-160312_014028.zip to S3. This may take a while.
Upload Complete.
Application test has been created.
Environment details for: dev-env
Application name: test
Region: us-east-2
Deployed Version: app-160312_014028
Environment ID: e-6fgpkjxyyi
Platform: 64bit Amazon Linux 2015.09 v2.0.8 running PHP 5.6
Tier: WebServer-Standard
CNAME: UNKNOWN
Updated: 2016-03-12 01:40:33.614000+00:00
Printing Status:
...
```

La commande suivante crée un environnement dans un VPC personnalisé.

```
$ eb create dev-vpc --vpc.id vpc-0ce8dd99 --vpc.elbsubnets subnet-b356d7c6,subnet-02f74b0c
--vpc.ec2subnets subnet-0bb7f0cd,subnet-3b6697c1 --vpc.securitygroup sg-70cff265
Creating application version archive "app-160312_014309".
```

```
Uploading test/app-160312_014309.zip to S3. This may take a while.  
Upload Complete.  
Environment details for: dev-vpc  
  Application name: test  
  Region: us-east-2  
  Deployed Version: app-160312_014309  
  Environment ID: e-pqkcia3mns  
  Platform: 64bit Amazon Linux 2015.09 v2.0.8 running Java 8  
  Tier: WebServer-Standard  
  CNAME: UNKNOWN  
  Updated: 2016-03-12 01:43:14.057000+00:00  
Printing Status:  
...
```

## eb deploy

### Description

Déploie le groupe source de l'application depuis le répertoire du projet initialisé vers l'application en cours d'exécution.

Si git est installé, l'interface de ligne de commande (CLI) EB utilise la commande `git archive` pour créer un fichier `.zip` à partir du contenu de la commande `git commit` la plus récente.

Cependant, quand `.ebignore` est présent dans le répertoire de votre projet, l'interface de ligne de commande (CLI) EB n'utilise pas de commandes et de sémantiques Git pour créer votre bundle de fichiers source. Autrement dit, l'interface de ligne de commande (CLI) EB ignore les fichiers spécifiés dans `.ebignore` et inclut tous les autres fichiers. Plus particulièrement, elle comprend des fichiers source non validés.

#### Note

Vous pouvez configurer l'interface de ligne de commande (CLI) EB pour déployer un artefact de votre processus de construction au lieu de créer un fichier ZIP de votre dossier de projet. Consultez [Déploiement d'un artefact à la place du dossier de projet \(p. 1039\)](#) pour plus de détails.

### Syntax

`eb deploy`

`eb deploy environment-name`

### Options

Nom	Description
<code>-l <b>version_label</b></code> ou <code>--label <b>version_label</b></code>	Spécifiez une étiquette à utiliser pour la version créée par l'interface de ligne de commande (CLI) EB. Si l'étiquette a déjà été utilisée, l'interface de ligne de commande (CLI) EB redéploie la version précédente avec cette étiquette. Type : chaîne
<code>--env-group-suffix <b>groupname</b></code>	Nom du groupe à ajouter au nom de l'environnement. A utiliser uniquement avec <a href="#">Compose Environments (p. 1058)</a> .

Nom	Description
-m " <i>version_description</i> " ou --message " <i>version_description</i> "	La description de la version de l'application, entourée de guillemets doubles. Type : chaîne
--modules <i>component-a</i> <i>component-b</i>	Liste des composants à mettre à jour. A utiliser uniquement avec <a href="#">Compose Environments (p. 1058)</a> .
-p ou --process	Prétraitez et validez les fichiers de configuration et le manifeste d'environnement dans le groupe source. La validation des fichiers de configuration permet d'identifier les problèmes avant de déployer la version de l'application dans un environnement.
--source codecommit/ <i>repository-name</i> / <i>branch-name</i>	Référentiel CodeCommit et branche. Voir <a href="#">Utilisation de l'interface de ligne de commande EB avec AWS CodeCommit (p. 1048)</a> .
--staged	Déployez des fichiers mis en lots dans l'index git au lieu de la validation HEAD.
--timeout <i>minutes</i>	Le nombre de minutes avant que la commande expire.
--version <i>version_label</i>	Une version de l'application existante à déployer. Type : chaîne
Options courantes (p. 1123)	

## Output

En cas de réussite, la commande renvoie l'état de l'opération deploy.

Si vous avez activé la prise en charge de CodeBuild dans votre application, eb deploy affiche les informations de CodeBuild au fur et à mesure de la création du code. Pour de plus amples informations sur la prise en charge de CodeBuild dans Elastic Beanstalk, veuillez consulter [Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild \(p. 1044\)](#).

## Example

L'exemple suivant déploie l'application en cours.

```
$ eb deploy
2018-07-11 21:05:22    INFO: Environment update is starting.
2018-07-11 21:05:27    INFO: Deploying new version to instance(s).
2018-07-11 21:05:53    INFO: New application version was deployed to running EC2 instances.
2018-07-11 21:05:53    INFO: Environment update completed successfully.
```

## eb events

### Description

Renvoie les événements les plus récents pour l'environnement.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande renvoie également les événements les plus récents pour l'environnement de génération.

## Syntax

`eb events`

`eb events environment-name`

## Options

Nom	Description
<code>-f</code>	Événements de flux. Pour annuler, appuyez sur CTRL+C.
ou	
<code>--follow</code>	

## Output

En cas de succès, la commande renvoie des événements récents.

## Example

L'exemple suivant renvoie les événements les plus récents.

```
$ eb events
2014-10-29 21:55:39      INFO    createEnvironment is starting.
2014-10-29 21:55:40      INFO    Using elasticbeanstalk-us-west-2-111122223333 as Amazon S3
                                storage bucket for environment data.
2014-10-29 21:55:57      INFO    Created load balancer named: awseb-e-r-AWSEBLoa-
NSKUOK5X6Z9J
2014-10-29 21:56:16      INFO    Created security group named: awseb-e-rxgrhjr9bx-stack-
AWSEBSecurityGroup-1UUHU5LZ20ZY7
2014-10-29 21:57:18      INFO    Waiting for EC2 instances to launch. This may take a few
                                minutes.
2014-10-29 21:57:18      INFO    Created Auto Scaling group named: awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingGroup-1TE320ZCJ9RPD
2014-10-29 21:57:22      INFO    Created Auto Scaling group policy named:
                                arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:2cced9e6-859b-421a-
be63-8ab34771155a:autoScalingGroupName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingGroup-1TE320ZCJ9RPD:policyName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingScaleUpPolicy-1I2ZSNVU4APRY
2014-10-29 21:57:22      INFO    Created Auto Scaling group policy named:
                                arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:1f08b863-
bf65-415a-b584-b7fa3a69a0d5:autoScalingGroupName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingGroup-1TE320ZCJ9RPD:policyName/awseb-e-rxgrhjr9bx-stack-
AWSEBAutoScalingScaleDownPolicy-1E3G7PZKZPSOG
2014-10-29 21:57:25      INFO    Created CloudWatch alarm named: awseb-e-rxgrhjr9bx-stack-
AWSEBCloudwatchAlarmLow-VF5EJ549FZBL
2014-10-29 21:57:25      INFO    Created CloudWatch alarm named: awseb-e-rxgrhjr9bx-stack-
AWSEBCloudwatchAlarmHigh-LA9YEW3O6WJO
2014-10-29 21:58:50      INFO    Added EC2 instance 'i-c7ee492d' to Auto ScalingGroup
                                'awseb-e-rxgrhjr9bx-stack-AWSEBAutoScalingGroup-1TE320ZCJ9RPD'.
2014-10-29 21:58:53      INFO    Successfully launched environment: tmp-dev
2014-10-29 21:59:14      INFO    Environment health has been set to GREEN
```

```
2014-10-29 21:59:43      INFO      Adding instance 'i-c7ee492d' to your environment.
```

## eb health

### Description

Renvoie l'intégrité la plus récente pour l'environnement.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande renvoie également l'état le plus récent pour l'environnement de génération.

### Syntax

```
eb health
```

```
eb health environment-name
```

### Options

Nom	Description
-r ou --refresh	Affichez des informations d'état de santé interactivement et mettez à jour toutes les 10 secondes à mesure que de nouvelles informations sont rapportées.
--mono	N'affichez pas la couleur dans le résultat.

### Output

En cas de réussite, la commande renvoie l'intégrité récente.

### Example

L'exemple suivant renvoie les informations d'intégrité les plus récentes pour un environnement Linux.

```
~/project $ eb health
elasticBeanstalkExa-env
2015-07-08 23:13:20
WebServer
 2.1 (Puma)
    total      ok     warning   degraded   severe     info     pending   unknown
      5         5          0           0          0         0         0          0
instance-id  status      cause
               health
Overall      Ok
i-d581497d  Ok
i-d481497c  Ok
i-136e00c0  Ok
i-126e00c1  Ok
i-8b2cf575  Ok
Ruby
```

instance-id	r/sec	%2xx	%3xx	%4xx	%5xx	p99	p90	p75	p50
		requests							
p10									
Overall	671.8	100.0	0.0	0.0	0.0	0.003	0.002	0.001	0.001
i-d581497d	143.0	1430	0	0	0	0.003	0.002	0.001	0.001
i-d481497c	128.8	1288	0	0	0	0.003	0.002	0.001	0.001
i-136e00c0	125.4	1254	0	0	0	0.004	0.002	0.001	0.001
i-126e00c1	133.4	1334	0	0	0	0.003	0.002	0.001	0.001
i-8b2cf575	141.2	1412	0	0	0	0.003	0.002	0.001	0.001
instance-id	type	az	running	cpu	load 1	load 5	user%	nice%	system% idle
% iowait%									
i-d581497d	t2.micro	1a	12 mins		0.0	0.04	6.2	0.0	1.0
92.5	0.1								
i-d481497c	t2.micro	1a	12 mins		0.01	0.09	5.9	0.0	1.6
92.4	0.1								
i-136e00c0	t2.micro	1b	12 mins		0.15	0.07	5.5	0.0	0.9
93.2	0.0								
i-126e00c1	t2.micro	1b	12 mins		0.17	0.14	5.7	0.0	1.4
92.7	0.1								
i-8b2cf575	t2.micro	1c	1 hour		0.19	0.08	6.5	0.0	1.2
92.1	0.1								
instance-id	status	id	version	deployments	ago				
i-d581497d	Deployed	1	Sample Application		12 mins				
i-d481497c	Deployed	1	Sample Application		12 mins				
i-136e00c0	Deployed	1	Sample Application		12 mins				
i-126e00c1	Deployed	1	Sample Application		12 mins				
i-8b2cf575	Deployed	1	Sample Application		1 hour				

## eb init

### Description

Définit les valeurs par défaut pour les applications Elastic Beanstalk créées avec l'interface de ligne de commande (CLI) EB en vous soumettant une série de questions.

#### Note

Les valeurs que vous définissez avec eb init s'appliquent uniquement au répertoire et au référentiel actuels sur l'ordinateur utilisé.

La commande ne crée rien dans votre compte Elastic Beanstalk. Pour créer un environnement Elastic Beanstalk, exécutez [eb create \(p. 1078\)](#) après l'exécution de eb init.

### Syntax

`eb init`

`eb init application-name`

### Options

Si vous exécutez eb init sans spécifier l'option `--platform`, l'interface de ligne de commande (CLI) EB vous invite à entrer une valeur pour chaque paramètre.

## Note

Pour utiliser eb init afin de créer une nouvelle paire de clés, l'outil ssh-keygen doit être installé sur votre ordinateur local et disponible à partir de la ligne de commande.

Nom	Description
<code>-i</code> <code>--interactive</code>	<p>Force l'interface de ligne de commande (CLI) EB à vous inviter à fournir une valeur pour chaque option de commande eb init.</p> <p><b>Note</b></p> <p>La commande <code>init</code> vous invite à fournir des valeurs pour les options de commande eb init qui n'ont pas une valeur (par défaut). Après la première fois que vous exécutez la commande eb init dans un répertoire, l'interface de ligne de commande (CLI) EB ne vous soumettra peut-être aucune options de commande. Par conséquent, utilisez l'option <code>--interactive</code> lorsque vous voulez modifier un paramètre que vous avez préalablement défini.</p>
<code>-k <i>keyname</i></code> <code>--keyname <i>keyname</i></code>	Nom de la paire de clés Amazon EC2 à utiliser avec le client Secure Shell (SSH) pour se connecter en toute sécurité aux instances Amazon EC2 exécutant votre application Elastic Beanstalk.
<code>--modules <i>folder-1</i> <i>folder-2</i></code>	Liste des répertoires enfants à initialiser. A utiliser uniquement avec <a href="#">Compose Environments (p. 1058)</a> .
<code>-p <i>platform-version</i></code> <code>--platform <i>platform-version</i></code>	<p>La <a href="#">version de la plateforme (p. 31)</a> à utiliser. Vous pouvez spécifier un nom de plateforme, une plateforme et une version de plateforme, une branche de plateforme, un nom de pile de solutions ou un ARN de pile de solutions.</p> <p>Exemples :</p> <ul style="list-style-type: none"> <li>• <code>php</code>, <code>PHP</code>, <code>node.js</code> – Dernière version pour la plateforme spécifiée</li> <li>• <code>php-7.2</code>, <code>"PHP 7.2"</code> – Version recommandée (généralement la plus récente) de la plateforme PHP 7.2</li> <li>• <code>"PHP 7.2 running on 64bit Amazon Linux"</code> – Version recommandée (généralement la plus récente) de la plateforme PHP dans cette branche de plateforme</li> <li>• <code>"64bit Amazon Linux 2017.09 v2.6.3 running PHP 7.1"</code> – Version de la plateforme PHP spécifiée par ce nom de pile de solutions</li> <li>• <code>"arn:aws:elasticbeanstalk:us-east-2::platform/PHP 7.1 running on 64bit Amazon Linux/2.6.3"</code> – Version de la plateforme PHP spécifiée par cet ARN de pile de solutions</li> </ul> <p>Utilisez <code>eb platform list (p. 1103)</code> pour obtenir la liste des configurations disponibles.</p>

Nom	Description	
	<p>Spécifiez l'option <code>--platform</code> pour éviter la configuration interactive.</p> <p><b>Note</b></p> <p>Si vous spécifiez cette option, l'interface de ligne de commande (CLI) EB ne vous invite pas à saisir les valeurs d'autres options. Au lieu de cela, il est seul responsable des valeurs par défaut pour chaque option. Vous pouvez spécifier des options pour tout ce pour quoi vous ne souhaitez pas utiliser les valeurs par défaut.</p>	
<code>--source codecommit/ repository-name/branch-name</code>	Référentiel CodeCommit et branche. Voir <a href="#">Utilisation de l'interface de ligne de commande EB avec AWS CodeCommit (p. 1048)</a> .	
<code>--tags key1=value1[,key2=value2]</code>	<p>Balisez votre application. Les balises sont spécifiées sous la forme d'une liste séparée par des virgules et composée de paires <code>key=value</code>.</p> <p>Pour en savoir plus, consultez <a href="#">Balisage des applications (p. 424)</a>.</p>	
<a href="#">Options courantes (p. 1123)</a>		

## Prise en charge de CodeBuild

Si vous exécutez eb init dans un dossier contenant un fichier `buildspec.yml`, Elastic Beanstalk analyse le fichier pour une entrée `eb_codebuild_settings` avec des options spécifiques à Elastic Beanstalk. Pour de plus amples informations sur la prise en charge de CodeBuild dans Elastic Beanstalk, veuillez consulter [Utilisation de l'interface de ligne de commande EB avec AWS CodeBuild \(p. 1044\)](#).

## Output

En cas de succès, la commande vous guide à travers la mise en place d'une nouvelle application Elastic Beanstalk grâce à une série d'invites.

## Example

L'exemple de demande suivant initialise l'interface de ligne de commande (CLI) EB et vous invite à saisir des informations relatives à votre application. Remplacez le texte `d'espace réservé` par vos propres valeurs.

```
$ eb init -i
Select a default region
1) us-east-1 : US East (N. Virginia)
2) us-west-1 : US West (N. California)
3) us-west-2 : US West (Oregon)
4) eu-west-1 : Europe (Ireland)
5) eu-central-1 : Europe (Frankfurt)
6) ap-south-1 : Asia Pacific (Mumbai)
7) ap-southeast-1 : Asia Pacific (Singapore)
...
(default is 3): 3
```

```
Select an application to use
1) HelloWorldApp
2) NewApp
3) [ Create new Application ]
(default is 3): 3

Enter Application Name
(default is "tmp"):
Application tmp has been created.

It appears you are using PHP. Is this correct?
(y/n): y

Select a platform branch.
1) PHP 7.2 running on 64bit Amazon Linux
2) PHP 7.1 running on 64bit Amazon Linux (Deprecated)
3) PHP 7.0 running on 64bit Amazon Linux (Deprecated)
4) PHP 5.6 running on 64bit Amazon Linux (Deprecated)
5) PHP 5.5 running on 64bit Amazon Linux (Deprecated)
6) PHP 5.4 running on 64bit Amazon Linux (Deprecated)
(default is 1): 1
Do you want to set up SSH for your instances?
(y/n): y

Select a keypair.
1) aws-eb
2) [ Create new KeyPair ]
(default is 2): 1
```

## eb labs

### Description

Sous-commandes de fonctionnalité expérimentale ou des travaux en cours de prise en charge eb labs. Ces commandes peuvent être supprimées ou retravaillées dans des versions futures de l'interface de ligne de commande (CLI) EB et ne sont pas garanties pour être compatibles.

Pour une liste des descriptions et des sous-commandes disponibles, exéutez eb labs --help.

## eb list

### Description

Répertorie tous les environnements dans l'application actuelle ou tous les environnements dans toutes les applications, comme indiqué par l'option --all.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande répertorie également les environnements de génération.

### Syntax

eb list

### Options

Nom	Description
-a	Répertorie tous les environnements de toutes les applications.

Nom	Description
ou	
--all	
-v	Fournit des informations plus détaillées sur tous les environnements, y compris des instances.
ou	
--verbose	
<a href="#">Options courantes (p. 1123)</a>	

## Output

En cas de réussite, la commande renvoie une liste de noms d'environnements dans laquelle votre environnement actuel est marqué d'un astérisque (\*).

### Exemple 1

L'exemple suivant répertorie vos environnements et indique que tmp-dev est votre environnement par défaut.

```
$ eb list
* tmp-dev
```

### Exemple 2

L'exemple suivant répertorie vos environnements avec plus de détails.

```
$ eb list --verbose
Region: us-west-2
Application: tmp
  Environments: 1
    * tmp-dev : ['i-c7ee492d']
```

## eb local

### Description

Utilisez eb local run pour exécuter des conteneurs de votre application localement dans Docker. Vérifiez l'état du conteneur de l'application avec eb local status. Ouvrez l'application dans un navigateur web avec eb local open. Récupérez l'emplacement des journaux de l'application avec eb local logs.

eb local setenv et eb local printenv vous permettent de définir et d'afficher des variables d'environnement qui sont fournies par des conteneurs Docker que vous exécutez localement avec eb local run.

Vous devez exécuter toutes les commandes eb local dans le répertoire de projet d'une application Docker ayant fait l'objet d'une initialisation en tant que référentiel de l'interface de ligne de commande (CLI) EB en utilisant eb init.

#### Note

Utilisez eb local sur un ordinateur local exécutant les systèmes d'exploitation Linux ou macOS. La commande ne prend pas en charge Windows.

Avant d'utiliser la commande sur macOS, installez Docker pour Mac et assurez-vous que boot2docker n'est pas installé (ou n'est pas dans le chemin d'exécution). La commande eb local essaie d'utiliser boot2docker s'il est présent, mais ne fonctionne pas bien avec elle sur macOS.

## Syntax

eb local run  
eb local status  
eb local open  
eb local logs  
eb local setenv  
eb local printenv

## Options

eb local run

Nom	Description
--envvars <i>key1=value1, key2=value2</i>	Définit des variables d'environnement que l'interface de ligne de commande (CLI) EB va passer aux conteneurs Docker locaux. Dans des environnements multicontainer, toutes les variables sont passées à tous les conteneurs.
--port <i>hostport</i>	Mappe un port sur l'hôte au port exposé sur le conteneur. Si vous ne spécifiez pas cette option, l'interface de ligne de commande (CLI) EB utilise le même port sur l'hôte et sur le conteneur.  Cette option fonctionne uniquement avec les applications de plateforme Docker. Elle ne s'applique pas à la plateforme multi-conteneurs Docker.
<a href="#">Options courantes (p. 1123)</a>	

eb local status  
eb local open  
eb local logs  
eb local setenv  
eb local printenv

Nom	Description
<a href="#">Options courantes (p. 1123)</a>	

## Output

eb local run

Messages d'état de Docker. Reste actif aussi longtemps que l'application est en cours d'exécution.  
Appuyez sur Ctrl+C pour arrêter l'application.

eb local status

L'état de chaque conteneur utilisé par l'application, en cours d'exécution ou non.

eb local open

Ouvre l'application dans un navigateur web puis se ferme.

eb local logs

L'emplacement des journaux générés dans votre répertoire de projet par des applications qui s'exécutent localement sous eb local run.

eb local setenv

Aucune

eb local printenv

Le nom et les valeurs de variables d'environnement définies avec eb local setenv.

## Examples

eb local run

```
~/project$ eb local run
Creating elasticbeanstalk_phpapp_1...
Creating elasticbeanstalk_nginxproxy_1...
Attaching to elasticbeanstalk_phpapp_1, elasticbeanstalk_nginxproxy_1
phpapp_1    | [23-Apr-2015 23:24:25] NOTICE: fpm is running, pid 1
phpapp_1    | [23-Apr-2015 23:24:25] NOTICE: ready to handle connections
```

eb local status

Affichez le statut de vos conteneurs locaux :

```
~/project$ eb local status
Platform: 64bit Amazon Linux 2014.09 v1.2.1 running Multi-container Docker 1.3.3 (Generic)
Container name: elasticbeanstalk_nginxproxy_1
Container ip: 127.0.0.1
Container running: True
Exposed host port(s): 80
Full local URL(s): 127.0.0.1:80

Container name: elasticbeanstalk_phpapp_1
Container ip: 127.0.0.1
Container running: True
Exposed host port(s): None
Full local URL(s): None
```

eb local logs

Affichez le chemin d'accès au journal pour le projet actuel :

```
~/project$ eb local logs
```

```
Elastic Beanstalk will write logs locally to /home/user/project/.elasticbeanstalk/logs/local.  
Logs were most recently created 3 minutes ago and written to /home/user/project/.elasticbeanstalk/logs/local/150420_234011665784.
```

## eb local setenv

Définissez des variables d'environnement à utiliser avec eb local run.

```
~/project$ eb local setenv PARAM1=value
```

Imprimez des variables d'environnement définies avec eb local setenv.

```
~/project$ eb local printenv  
Environment Variables:  
PARAM1=value
```

# eb logs

## Description

La commande eb logs a deux finalités distinctes : activer ou désactiver la diffusion de journaux vers CloudWatch Logs, et récupérer les journaux d'instance ou les journaux CloudWatch Logs. Avec l'option --cloudwatch-logs (-cw), la commande active ou désactive la diffusion de journaux. Sans cette option, elle récupère les journaux.

En cas de récupération des journaux, spécifiez l'option --all ou --zip, --stream pour récupérer les journaux complets. Si vous ne spécifiez aucune de ces options, Elastic Beanstalk récupère les journaux de queue.

La commande traite les journaux associés à l'environnement par défaut ou l'environnement spécifié. Les journaux concernés varient selon le type de conteneur. Si le répertoire racine contient un fichier platform.yaml spécifiant une plateforme personnalisée, cette commande traite également les journaux associés à l'environnement de génération.

Pour plus d'informations, consultez [the section called “CloudWatch Logs” \(p. 895\)](#).

## Syntax

Pour activer ou désactiver la diffusion de journaux vers CloudWatch Logs :

```
eb logs --cloudwatch-logs [enable | disable] [--cloudwatch-log-source instance | environment-health | all] [environment-name]
```

Pour récupérer des journaux d'instance :

```
eb logs [-all | --zip | --stream] [--cloudwatch-log-source instance] [--instance instance-id] [--log-group log-group] [environment-name]
```

Pour extraire les journaux d'intégrité de l'environnement :

```
eb logs [-all | --zip | --stream] --cloudwatch-log-source environment-health [environment-name]
```

## Options

Nom	Description
-cw [enable   disable] ou --cloudwatch-logs [enable   disable]	Active ou désactive la diffusion de journaux vers CloudWatch Logs. Si aucun argument n'est fourni, la diffusion de journaux est activée. Si l'option --cloudwatch-log-source (-cls) n'est pas ajoutée, la diffusion des journaux d'instance est activée ou désactivée.
-cls instance   environment-health   all ou --cloudwatch-log-source instance   environment- health   all	Spécifie la source des journaux lors de l'utilisation de CloudWatch Logs. Si l'on utilise l'option d'activation ou de désactivation de la commande, ce sont les journaux pour lesquels vous activez ou désactivez la diffusion à CloudWatch Logs. Si l'on utilise l'option d'extraction de la commande, ce sont les journaux qui sont récupérés à partir de CloudWatch Logs.  Valeurs valides : <ul style="list-style-type: none"><li>• Avec --cloudwatch-logs (activation ou désactivation) - instance   environment-health   all</li><li>• Sans --cloudwatch-logs (extraction) - instance   environment-health</li></ul> Significations des valeurs : <ul style="list-style-type: none"><li>• instance (valeur par défaut) - Journaux d'instance</li><li>• environment-health - Journaux d'intégrité d'environnement (uniquement pris en charge lorsque la fonction d'intégrité améliorée est activée dans l'environnement)</li><li>• all - Les deux sources de journaux</li></ul>
-a ou --all	Récupère les journaux complets et les enregistre dans le répertoire .elasticbeanstalk/logs.
-z ou --zip	Récupère les journaux complets, les compresse dans un fichier .zip, puis enregistre le fichier dans le répertoire .elasticbeanstalk/logs.
--stream	Diffuse (sorties en continu) les journaux complets. Avec cette option, la commande poursuit son exécution jusqu'à ce que vous l'interrompiez (appuyez sur Ctrl+C).
-i <i>instance-id</i> ou --instance <i>instance-id</i>	Extrait les journaux de l'instance spécifiée uniquement.
-g <i>log-group</i> ou	Spécifie le groupe de journaux CloudWatch Logs à partir duquel extraire les journaux. L'option est valide uniquement lorsque la diffusion des journaux d'instance dans CloudWatch Logs est activée.

Nom	Description
--log-group <i>log-group</i>	<p>Si la diffusion des journaux d'instance est activée et que vous ne spécifiez pas l'option --log-group, le groupe de journaux par défaut est l'un des suivants :</p> <ul style="list-style-type: none"> <li>Plateformes Linux – /aws/elasticbeanstalk/<i>environment-name</i>/var/log/eb-activity.log</li> <li>Plateformes Windows – /aws/elasticbeanstalk/<i>environment-name</i>/EBDeploy-Log</li> </ul> <p>Pour plus d'informations sur le groupe de journaux correspondant à chaque fichier journal, consultez <a href="#">Méthode de configuration de CloudWatch Logs par Elastic Beanstalk (p. 898)</a>.</p>
<a href="#">Options courantes (p. 1123)</a>	

## Output

Par défaut, affiche les journaux directement sur le terminal. Utilise un programme de pagination pour afficher la sortie. Appuyez sur **Q** ou **q** pour quitter.

Avec **--stream**, affiche les journaux existants sur le terminal et poursuit son exécution. Appuyez sur **Ctrl +C** pour quitter.

Avec **--all** et **--zip**, enregistre les journaux dans des fichiers locaux et affiche l'emplacement des fichiers.

## Examples

L'exemple suivant active la diffusion des journaux d'instance dans CloudWatch Logs.

```
$ eb logs -cw enable
Enabling instance log streaming to CloudWatch for your environment
After the environment is updated you can view your logs by following the link:
https://console.aws.amazon.com/cloudwatch/home?region=us-east-1#logs:prefix=/aws/
elasticbeanstalk/environment-name
Printing Status:
2018-07-11 21:05:20    INFO: Environment update is starting.
2018-07-11 21:05:27    INFO: Updating environment environment-name's configuration
settings.
2018-07-11 21:06:45    INFO: Successfully deployed new configuration to environment.
```

L'exemple suivant extrait les journaux d'instance dans un fichier .zip.

```
$ eb logs --zip
Retrieving logs...
Logs were saved to /home/workspace/environment/.elasticbeanstalk/logs/150622_173444.zip
```

## eb open

### Description

Ouvre l'URL publique de votre site web dans le navigateur par défaut.

## Syntax

eb open

eb open **environment-name**

## Options

Nom	Description
Options courantes (p. 1123)	

## Output

La commande eb open n'a pas de sortie. Au lieu de cela, elle ouvre l'application dans une fenêtre de navigateur.

## eb platform

### Description

Cette commande prend en charge deux espaces de travail différents :

[Plateforme \(p. 1103\)](#)

Utilisez cet espace de travail pour gérer les plateformes personnalisées.

[Environnement \(p. 1109\)](#)

Utilisez cet espace de travail pour sélectionner une plateforme par défaut ou afficher des informations sur la plateforme actuelle.

Elastic Beanstalk fournit le raccourci ebp pour eb platform.

#### Note

Windows PowerShell utilise ebp comme alias de commande. Si vous exécutez l'interface de ligne de commande (CLI) EB dans Windows PowerShell, utilisez la forme longue de cette commande — eb platform.

## Utilisation de la plateforme eb pour les plateformes personnalisées

Répertorie les versions de la plateforme actuelle et vous permet de gérer des plateformes personnalisées.

### Syntax

eb platform create [**version**] [**options**]

eb platform delete [**version**] [**options**]

eb platform events [**version**] [**options**]

```
eb platform init [platform] [options]
eb platform list [options]
eb platform logs [version] [options]
eb platform status [version] [options]
eb platform use [platform] [options]
```

## Options

Nom	Description
<code>create [<i>version</i>] [<i>options</i>]</code>	Créer une nouvelle version de la plateforme. <a href="#">En savoir plus (p. 1105)</a> .
<code>delete <i>version</i> [<i>options</i>]</code>	Supprimer une version de plateforme. <a href="#">En savoir plus (p. 1106)</a> .
<code>events [<i>version</i>] [<i>options</i>]</code>	Afficher les événements issus de la version de plateforme. <a href="#">En savoir plus (p. 1106)</a> .
<code>init [<i>platform</i>] [<i>options</i>]</code>	Initialiser un référentiel de plateforme. <a href="#">En savoir plus (p. 1107)</a> .
<code>list [<i>options</i>]</code>	Répertorier les versions de la plateforme actuelle. <a href="#">En savoir plus (p. 1107)</a> .
<code>logs [<i>version</i>] [<i>options</i>]</code>	Afficher les journaux de l'environnement de génération pour une version de plateforme. <a href="#">En savoir plus (p. 1108)</a> .
<code>status [<i>version</i>] [<i>options</i>]</code>	Afficher le statut de la version de plateforme. <a href="#">En savoir plus (p. 1108)</a> .
<code>use [<i>platform</i>] [<i>options</i>]</code>	Choisir une autre plateforme à partir de laquelle créer de nouvelles versions. <a href="#">En savoir plus (p. 1108)</a> .
Options courantes (p. 1123)	

## Options courantes

Toutes les commandes eb platform contiennent les options courantes suivantes.

Nom	Description
<code>-h</code>	Affiche un message d'aide, puis se ferme.
OR	
<code>--help</code>	
<code>--debug</code>	Affiche une sortie de débogage supplémentaire.
<code>--quiet</code>	Supprime toutes les sorties.
<code>-v</code>	Affiche une sortie supplémentaire.
OR	

Nom	Description
--verbose	
--profile <i>PROFILE</i>	Utilise le profil <i>PROFILE</i> spécifié à partir de vos informations d'identification.
-r <i>REGION</i> OR --region <i>REGION</i>	Utilise la région <i>REGION</i> .
--no-verify-ssl	Ne vérifie pas les certificats SSL AWS.

## Eb platform create

Crée une nouvelle version de la plateforme et renvoie l'ARN de la nouvelle version. Si aucun environnement de génération n'est en cours d'exécution dans la région actuelle, cette commande en lance un. Les options *version* et increment (-M, -m et -p) s'excluent mutuellement.

### Options

Nom	Description
<i>version</i>	Si l'option <i>version</i> n'est pas spécifiée, crée une nouvelle version basée sur la plateforme la plus récente avec incrémentation de la version de correctif (N dans n.n.N).
-M OR --major-increment	Incrémente le numéro de version majeure (N dans N.n.n).
-m OR --minor-increment	Incrémente le numéro de version mineure (N dans n.N.n).
-p OR --patch-increment	Incrémente le numéro de version du correctif (N dans n.n.N).
-i <i>INSTANCE_TYPE</i> OR --instance-type <i>INSTANCE_TYPE</i>	Utilisez <i>INSTANCE_TYPE</i> comme type d'instance, par exemple <b>t1.micro</b> .
-ip <i>INSTANCE_PROFILE</i> OR --instance-profile <i>INSTANCE_PROFILE</i>	Utilisez <i>INSTANCE_PROFILE</i> comme profil d'instance lors de la création d'AMI pour une plateforme personnalisée.  Si l'option -ip n'est pas spécifiée, crée le profil d'instance <code>aws-elasticbeanstalk-custom-platform-ec2-role</code> et l'utilise pour la plateforme personnalisée.

Nom	Description
--tags <code>key1=value1[,key2=value2]</code>	Balisez votre version de plateforme personnalisée. Les balises sont spécifiées sous la forme d'une liste séparée par des virgules et composée de paires key=value.  Pour en savoir plus, consultez <a href="#">Balisage des versions de plateforme personnalisée (p. 1157)</a> .
--timeout <code>minutes</code>	Définissez le nombre de minutes avant que la commande expire.
--vpc.id <code>VPC_ID</code>	ID du VPC dans lequel Packer effectue la création.
--vpc.subnets <code>VPC_SUBNETS</code>	Sous-réseaux VPC dans lesquels Packer effectue la création.
--vpc.publicip	Associe des adresses IP publiques aux instances EC2 lancées.

## Eb platform delete

Supprimer une version de plateforme. La version n'est pas supprimée si un environnement l'utilise.

### Options

Nom	Description
<code>version</code>	La version à supprimer. Cette valeur est obligatoire.
--cleanup	Supprimer toutes les versions de plateforme à l'état Failed.
--all-platforms	Si --cleanup est spécifié, supprimer toutes les versions de plateforme à l'état Failed pour toutes les plateformes.
--force	Ne pas demander de confirmation lors de la suppression d'une version.

## Eb platform events

Afficher les événements issus de la version de plateforme. Si `version` est spécifié, afficher les événements de cette version, sinon, afficher les événements de la version actuelle.

### Options

Nom	Description
<code>version</code>	La version pour laquelle des événements sont affichés. Cette valeur est obligatoire.
-f OR --follow	Continuer à afficher les événements au fur et à mesure qu'ils se produisent.

## Eb platform init

Initialiser un référentiel de plateforme.

### Options

Nom	Description
<code>platform</code>	Le nom de la plateforme à initialiser. Cette valeur est obligatoire, sauf si le mode interactif <code>-i</code> est activé.
<code>-i</code> OR <code>--interactive</code>	Utiliser le mode interactif.
<code>-k KEYNAME</code> OR <code>--keyname KEYNAME</code>	Le nom de la clé EC2 par défaut.

Vous pouvez exécuter cette commande dans un répertoire qui a été initialisé auparavant, mais dans ce cas, vous ne pouvez pas modifier le type de l'espace de travail.

Pour relancer l'initialisation avec d'autres options, utilisez l'option `-i`.

## Eb platform list

Répertorier les versions de la plateforme associées à un espace de travail (répertoire) ou à une région.

La commande renvoie des résultats différents en fonction du type d'espace de travail dans lequel vous l'exécutez, comme suit :

- Dans un espace de travail de plateforme (un répertoire initialisé par `eb platform init`), la commande renvoie une liste de toutes les versions de plateforme de la plateforme personnalisée définie dans l'espace de travail. Ajoutez l'option `--all-platforms` ou `--verbose` pour obtenir une liste de toutes les versions de plateforme de toutes les plateformes personnalisées que votre compte possède dans la région associée à l'espace de travail.
- Dans un espace de travail d'application (un répertoire initialisé par `eb init`), la commande renvoie une liste de toutes les versions de plateforme, à la fois pour les plateformes prises en charge par Elastic Beanstalk et pour les plateformes personnalisées de votre compte. La liste utilise des noms de version de plateforme courts, et certaines variantes de version de plateforme peuvent être associées. Ajoutez l'option `--verbose` pour obtenir une liste détaillée avec les noms complets et toutes les variantes répertoriées séparément.
- Dans un répertoire non initialisé, la commande fonctionne uniquement avec l'option `--region`. Elle renvoie une liste de toutes les versions de plateforme gérées par Elastic Beanstalk prises en charge dans la région. La liste utilise des noms de version de plateforme courts, et certaines variantes de version de plateforme peuvent être associées. Ajoutez l'option `--verbose` pour obtenir une liste détaillée avec les noms complets et toutes les variantes répertoriées séparément.

## Options

Nom	Description
-a OR --all-platforms	Valide uniquement dans un espace de travail initialisé (un répertoire initialisé par <code>eb platform init</code> ou <code>eb init</code> ). Répertorie les versions de plateforme de toutes les plateformes personnalisées associées à votre compte.
-s <i>STATUS</i> OR --status <i>STATUS</i>	Répertorier uniquement les plateformes correspondant à <i>STATUS</i> : <ul style="list-style-type: none"><li>• Prêt</li><li>• Échec</li><li>• Suppression en cours</li><li>• Création</li></ul>

## Eb platform logs

Afficher les journaux de l'environnement de génération pour une version de plateforme.

## Options

Nom	Description
<i>version</i>	La version de la plateforme pour laquelle des journaux sont affichés. Si cette option est omise, afficher les journaux de la version actuelle.
--stream	Diffuser des journaux de déploiement qui ont été configurés avec CloudWatch.

## Eb platform status

Afficher le statut de la version de plateforme.

## Options

Nom	Description
<i>version</i>	La version de la plateforme pour laquelle le statut est extrait. Si cette option est omise, afficher le statut de la version actuelle.

## Eb platform use

Choisir une autre plateforme à partir de laquelle créer de nouvelles versions.

## Options

Nom	Description
<i>platform</i>	Spécifiez <i>platform</i> comme version active pour cet espace de travail. Cette valeur est obligatoire.

## Utilisation de la plateforme eb pour les environnements

Répertorie les plateformes prises en charge et vous permet de définir la plate-forme par défaut et la version de la plateforme à utiliser quand vous lancez un environnement. Utilisez eb platform list pour afficher une liste de toutes les plateformes prises en charge. Utilisez eb platform select pour modifier la plateforme pour votre projet. Utilisez eb platform show pour afficher la plateforme sélectionnée de votre projet.

### Syntax

```
eb platform list  
eb platform select  
eb platform show
```

### Options

Nom	Description
list	Répertorier la version de la plateforme actuelle.
select	Sélectionner la plateforme par défaut.
show	Afficher des informations sur la plateforme actuelle.

### Exemple 1

L'exemple suivant répertorie les noms de toutes les configurations pour toutes les plateformes qu'Elastic Beanstalk prend en charge.

```
$ eb platform list  
docker-1.5.0  
glassfish-4.0-java-7-(preconfigured-docker)  
glassfish-4.1-java-8-(preconfigured-docker)  
go-1.3-(preconfigured-docker)  
go-1.4-(preconfigured-docker)  
iis-7.5  
iis-8  
iis-8.5  
multi-container-docker-1.3.3-(generic)  
node.js  
php-5.3  
php-5.4  
php-5.5  
python  
python-2.7  
python-3.4  
python-3.4-(preconfigured-docker)  
ruby-1.9.3  
ruby-2.0-(passenger-standalone)  
ruby-2.0-(puma)  
ruby-2.1-(passenger-standalone)  
ruby-2.1-(puma)  
ruby-2.2-(passenger-standalone)  
ruby-2.2-(puma)  
tomcat-6  
tomcat-7  
tomcat-7-java-6  
tomcat-7-java-7
```

```
tomcat-8-java-8
```

## Exemple 2

L'exemple suivant vous invite à choisir parmi une liste de plateformes, ainsi que la version que vous souhaitez déployer pour la plateforme spécifiée.

```
$ eb platform select
Select a platform.
1) PHP
2) Node.js
3) IIS
4) Tomcat
5) Python
6) Ruby
7) Docker
8) Multi-container Docker
9) GlassFish
10) Go
(default is 1): 5

Select a platform version.
1) Python 2.7
2) Python
3) Python 3.4 (Preconfigured - Docker)
```

## Exemple 3

L'exemple suivant présente des informations sur la plateforme par défaut actuelle.

```
$ eb platform show
Current default platform: Python 2.7
New environments will be running: 64bit Amazon Linux 2014.09 v1.2.0 running Python 2.7

Platform info for environment "tmp-dev":
Current: 64bit Amazon Linux 2014.09 v1.2.0 running Python
Latest: 64bit Amazon Linux 2014.09 v1.2.0 running Python
```

# eb printenv

## Description

Affiche toutes les propriétés de l'environnement dans la fenêtre de commande.

## Syntax

eb printenv

eb printenv **environment-name**

## Options

Nom	Description
Options courantes (p. 1123)	

## Output

En cas de réussite, la commande renvoie l'état de l'opération `printenv`.

## Example

L'exemple suivant affiche les propriétés de l'environnement pour l'environnement spécifié.

```
$ eb printenv
Environment Variables:
    PARAM1 = Value1
```

## eb restore

### Description

Reconstruit un environnement suspendu, en créant un nouvel environnement avec le même nom, le même ID et la même configuration. Le nom de l'environnement, le nom de domaine et la version de l'application doivent être disponibles pour que la reconstruction réussisse.

### Syntax

`eb restore`

`eb restore environment_id`

### Options

Nom	Description
<a href="#">Options courantes (p. 1123)</a>	

## Output

L'interface de ligne de commande (CLI) EB affiche une liste d'environnements suspendus qui sont disponibles pour être restaurés.

## Example

```
$ eb restore
Select a terminated environment to restore

#      Name          ID           Application Version      Date Terminated      Ago
3      gamma        e-s7miej8e9  app-77e3-161213_211138  2016/12/14 20:32 PST  13 mins
2      beta         e-sj28uu2wia app-77e3-161213_211125  2016/12/14 20:32 PST  13 mins
1      alpha        e-gia8mphu6q app-77e3-161213_211109  2016/12/14 16:21 PST  4 hours

(Commands: Quit, Restore, # #)

Selected environment alpha
Application:      scorekeep
Description:      Environment created from the EB CLI using "eb create"
```

```
CNAME: alpha.h23tbtbm92.us-east-2.elasticbeanstalk.com
Version: app-77e3-161213_211109
Platform: 64bit Amazon Linux 2016.03 v2.1.6 running Java 8
Terminated: 2016/12/14 16:21 PST
Restore this environment? [y/n]: y

2018-07-11 21:04:20 INFO: restoreEnvironment is starting.
2018-07-11 21:04:39 INFO: Created security group named: sg-e2443f72
...
```

## eb scale

### Description

Adapte l'environnement pour qu'il s'exécute toujours sur un certain nombre d'instances, en définissant à la fois les nombres minimal et maximal d'instances sur le nombre spécifié.

### Syntax

eb scale **number-of-instances**

eb scale **number-of-instances environment-name**

### Options

Nom	Description
--timeout	Le nombre de minutes avant que la commande expire.
<a href="#">Options courantes (p. 1123)</a>	

### Output

En cas de réussite, la commande met à jour les nombres minimal et maximal d'instances à exécuter sur le nombre spécifié.

### Example

L'exemple suivant définit le nombre d'instances sur 2.

```
$ eb scale 2
2018-07-11 21:05:22 INFO: Environment update is starting.
2018-07-11 21:05:27 INFO: Updating environment tmp-dev's configuration settings.
2018-07-11 21:08:53 INFO: Added EC2 instance 'i-5fce3d53' to Auto Scaling Group 'awseb-e-2cpfjbra9a-stack-AWSEBAutoScalingGroup-7AXY7U13ZQ6E'.
2018-07-11 21:08:58 INFO: Successfully deployed new configuration to environment.
2018-07-11 21:08:59 INFO: Environment update completed successfully.
```

## eb setenv

### Description

Définit les [propriétés d'environnement \(p. 632\)](#) pour l'environnement par défaut.

## Syntax

eb setenv **key=value**

Vous pouvez inclure autant de propriétés que vous voulez, mais la taille totale de toutes les propriétés ne peut pas dépasser 4 096 octets. Vous pouvez supprimer une variable en laissant la valeur vierge. Pour connaître les limites, consultez [Configuration des propriétés de l'environnement \(p. 634\)](#).

### Note

Si *value* contient un [caractère spécial](#), vous devez l'échapper en le faisant précédé d'un caractère `\`.

## Options

Nom	Description
--timeout	Le nombre de minutes avant que la commande expire.
<a href="#">Options courantes (p. 1123)</a>	

## Output

En cas de réussite, la commande affiche que la mise à jour de l'environnement a abouti.

## Example

L'exemple suivant définit la variable d'environnement ExampleVar.

```
$ eb setenv ExampleVar=ExampleValue
2018-07-11 21:05:25      INFO: Environment update is starting.
2018-07-11 21:05:29      INFO: Updating environment tmp-dev's configuration settings.
2018-07-11 21:06:50      INFO: Successfully deployed new configuration to environment.
2018-07-11 21:06:51      INFO: Environment update completed successfully.
```

La commande suivante définit plusieurs propriétés d'environnement. Elle ajoute la propriété de l'environnement nommée `foo` et définit sa valeur sur `bar`, modifie la valeur de la propriété `JDBC_CONNECTION_STRING` et supprime les propriétés `PARAM4` et `PARAM5`.

```
$ eb setenv foo=bar JDBC_CONNECTION_STRING=hello PARAM4= PARAM5=
```

## eb ssh

## Description

### Note

Cette commande ne fonctionne pas avec des environnements exécutant des instances Windows Server.

Connectez-vous à une instance Linux Amazon EC2 dans votre environnement à l'aide de SSH (Secure Shell). Si un environnement comporte plusieurs instances en cours d'exécution, l'interface de ligne de commande (CLI) EB vous invite à spécifier à quelle instance vous souhaitez vous connecter. Pour utiliser cette commande, SSH doit être installé sur votre ordinateur local et disponible à partir de la ligne

de commande. Les fichiers de clé privée doivent se trouver dans un dossier nommé `.ssh` sous votre répertoire utilisateur, et les instances EC2 de votre environnement doivent avoir des adresses IP publiques.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande se connecte également à des instances dans l'environnement personnalisé.

### Clés SSH

Si vous n'avez pas déjà configuré SSH, vous pouvez utiliser l'interface de ligne de commande (CLI) EB pour créer une clé lorsque vous exécutez `eb init`. Si vous avez déjà exécuté `eb init`, exécutez-le à nouveau avec l'option `--interactive` et sélectionnez Yes (Oui) et Create New Keypair (Créer une nouvelle paire de clés) lorsque vous êtes invité à configurer SSH. Les clés créées au cours de ce processus sont stockées dans le dossier approprié par l'interface de ligne de commande (CLI) EB.

Cette commande ouvre temporairement le port 22 dans le groupe de sécurité de votre environnement pour le trafic entrant à partir de 0.0.0.0/0 (toutes les adresses IP) si aucune règle pour le port 22 n'est déjà en place. Si vous avez configuré le groupe de sécurité de votre environnement pour ouvrir le port 22 sur une plage CIDR limitée pour une sécurité accrue, l'interface de ligne de commande (CLI) EB respectera ce paramètre et renoncera à toutes modifications apportées au groupe de sécurité. Pour substituer ce comportement et forcer l'interface de ligne de commande (CLI) EB à ouvrir le port 22 à tout trafic entrant, utilisez l'option `--force`.

Consultez [Groupes de sécurité \(p. 542\)](#) pour obtenir des informations sur la configuration du groupe de sécurité de votre environnement.

## Syntax

`eb ssh`

`eb ssh environment-name`

## Options

Nom	Description
<code>-i</code> ou <code>--instance</code>	Spécifie l'ID d'instance de l'instance à laquelle vous vous connectez. Nous vous recommandons d'utiliser cette option.
<code>-n</code> ou <code>--number</code>	Spécifiez l'instance à laquelle se connecter à l'aide d'un chiffre.
<code>-o</code> ou <code>--keep_open</code>	Laissez le port 22 ouvert sur le groupe de sécurité à la fin de la session SSH.
<code>--command</code>	Exécutez une commande shell sur l'instance spécifiée au lieu de démarrer une session SSH.
<code>--custom</code>	Spécifiez une commande SSH à utiliser au lieu de « <code>ssh -i keyfile</code> ». N'incluez pas d'utilisateur distant ni de nom d'hôte.

Nom	Description
--setup	Modifiez la paire de clés attribuée aux instances de l'environnement (requiert le remplacement des instances).
--force	Ouvrez le port 22 au trafic entrant à partir de 0.0.0.0/0 dans le groupe de sécurité de l'environnement, même si le groupe de sécurité est déjà configuré pour SSH.  Utilisez cette option si le groupe de sécurité de votre environnement est configuré pour ouvrir le port 22 à une plage CIDR limitée qui n'inclut pas l'adresse IP à partir de laquelle vous essayez de vous connecter.
--timeout <i>minutes</i>	Définissez le nombre de minutes avant que la commande expire.  Peut être utilisé uniquement avec l'argument --setup.
Options courantes (p. 1123)	

## Output

En cas de réussite, la commande ouvre une connexion SSH sur l'instance.

## Example

L'exemple suivant vous connecte à l'environnement spécifié.

```
$ eb ssh
Select an instance to ssh into
1) i-96133799
2) i-5931e053
(default is 1): 1
INFO: Attempting to open port 22.
INFO: SSH port 22 open.
The authenticity of host '54.191.45.125 (54.191.45.125)' can't be established.
RSA key fingerprint is ee:69:62:df:90:f7:63:af:52:7c:80:60:1b:3b:51:a9.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.191.45.125' (RSA) to the list of known hosts.

      _|_ _|_
      _|(_ /   Amazon Linux AMI
      ___\__|__|_

https://aws.amazon.com/amazon-linux-ami/2014.09-release-notes/
No packages needed for security; 1 packages available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-8-185 ~]$ ls
[ec2-user@ip-172-31-8-185 ~]$ exit
logout
Connection to 54.191.45.125 closed.
INFO: Closed port 22 on ec2 instance security group
```

## eb status

### Description

Fournit des informations sur l'état de l'environnement.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande fournit également des informations sur l'environnement de génération.

## Syntax

`eb status`

`eb status environment-name`

## Options

Nom	Description
<code>-v</code> ou <code>--verbose</code>	Fournit plus d'informations sur les instances individuelles, telles que leur statut avec l'équilibrage de charge Elastic Load Balancing.
<a href="#">Options courantes (p. 1123)</a>	

## Output

En cas de réussite, la commande renvoie les informations suivantes sur l'environnement :

- Nom de l'environnement
- Nom de l'application
- Version de l'application déployée
- ID de l'environnement
- Plateforme
- Niveau de l'environnement
- CNAME
- Heure de dernière mise à jour de l'environnement
- État
- Santé

Si vous utilisez le mode détaillé, l'interface de ligne de commande (CLI) EB vous fournit également le nombre d'instances Amazon EC2 en cours d'exécution.

## Example

L'exemple suivant affiche l'état du tmp-dev de l'environnement.

```
$ eb status
Environment details for: tmp-dev
  Application name: tmp
  Region: us-west-2
  Deployed Version: None
  Environment ID: e-2cpfjbra9a
  Platform: 64bit Amazon Linux 2014.09 v1.0.9 running PHP 5.5
  Tier: WebServer-Standard-1.0
  CNAME: tmp-dev.elasticbeanstalk.com
  Updated: 2014-10-29 21:37:19.050000+00:00
  Status: Launching
```

Health: Grey

## eb swap

### Description

Echange le CNAME de l'environnement avec le CNAME d'un autre environnement (par exemple, pour éviter des temps d'arrêt lorsque vous mettez à jour la version de votre application).

#### Note

Si vous avez plus de deux environnements, vous êtes invité à sélectionner dans une liste d'environnements le nom de l'environnement qui utilise actuellement le CNAME que vous souhaitez. Pour supprimer cela, vous pouvez spécifier le nom de l'environnement à utiliser en incluant l'option `-n` lorsque vous exécutez la commande.

### Syntax

`eb swap`

`eb swap environment-name`

#### Note

La valeur `environment-name` correspond à l'environnement pour lequel vous voulez un autre CNAME. Si vous ne spécifiez pas `environment-name` comme paramètre de ligne de commande lorsque vous exécutez `eb swap`, l'interface de ligne de commande (CLI) EB met à jour le CNAME de l'environnement par défaut.

### Options

Nom	Description
<code>-n</code> ou <code>--destination_name</code>	Spécifie le nom de l'environnement avec lequel vous souhaitez échanger des CNAME. Si vous exécutez <code>eb swap</code> sans cette option, alors l'interface de ligne de commande (CLI) EB vous invite à choisir dans une liste de vos environnements.
<a href="#">Options courantes (p. 1123)</a>	

### Output

En cas de réussite, la commande renvoie l'état de l'opération `swap`.

### Examples

L'exemple suivant échange le `tmp-dev` de l'environnement avec `live-env`.

```
$ eb swap
Select an environment to swap with.
1) staging-dev
2) live-env
(default is 1): 2
2018-07-11 21:05:25    INFO: swapEnvironmentCNAMEs is starting.
2018-07-11 21:05:26    INFO: Swapping CNAMEs for environments 'tmp-dev' and 'live-env'.
```

```
2018-07-11 21:05:30    INFO: 'tmp-dev.elasticbeanstalk.com' now points to 'awseb-e-j-  
AWSEBLoa-M7U21VXNLWHN-487871449.us-west-2.elb.amazonaws.com'.  
2018-07-11 21:05:30    INFO: Completed swapping CNAMEs for environments 'tmp-dev' and  
'live-env'.
```

L'exemple suivant échange le tmp-dev de l'environnement avec le live-env de l'environnement mais ne vous invite pas à entrer ou à sélectionner une valeur pour aucun paramètre.

```
$ eb swap tmp-dev --destination_name live-env  
2018-07-11 21:18:12    INFO: swapEnvironmentCNAMEs is starting.  
2018-07-11 21:18:13    INFO: Swapping CNAMEs for environments 'tmp-dev' and 'live-env'.  
2018-07-11 21:18:17    INFO: 'tmp-dev.elasticbeanstalk.com' now points to 'awseb-e-j-  
AWSEBLoa-M7U21VXNLWHN-487871449.us-west-2.elb.amazonaws.com'.  
2018-07-11 21:18:17    INFO: Completed swapping CNAMEs for environments 'tmp-dev' and  
'live-env'.
```

## eb tags

### Description

Ajoutez, supprimez, mettez à jour et listez des balises d'une ressource Elastic Beanstalk.

Pour de plus amples informations sur le balisage des ressources dans Elastic Beanstalk, veuillez consulter [Balisage des ressources d'application Elastic Beanstalk \(p. 423\)](#).

### Syntax

```
eb tags [environment-name] [--resource ARN] -l | --list  
eb tags [environment-name] [--resource ARN] -a | --add key1=value1[,key2=value2 ...]  
eb tags [environment-name] [--resource ARN] -u | --update key1=value1[,key2=value2 ...]  
eb tags [environment-name] [--resource ARN] -d | --delete key1[,key2 ...]
```

Vous pouvez combiner les options de sous-commande --add, --update et --delete en une seule commande. Au moins l'une d'elles est obligatoire. Vous ne pouvez pas combiner n'importe laquelle des trois options de sous-commande avec --list.

Sans tous les arguments supplémentaires, tous ces commandes répertorient ou modifient les balises de l'environnement par défaut dans le répertoire actuel de l'application. Avec un argument *environment-name*, les commandes listent ou modifient les balises de cet environnement. Avec l'option --resource, les commandes listent ou modifient les balises de n'importe quelle ressource Elastic Beanstalk – une application, un environnement, une version d'application, une configuration enregistrée, ou une version de plateforme personnalisée. Spécifiez la ressource par son Amazon Resource Name (ARN).

### Options

Aucune de ces options n'est obligatoire. Si vous exécutez eb create sans aucune option, vous êtes invité à saisir ou à sélectionner une valeur pour chaque paramètre.

Nom	Description
-l	Liste tous les balises qui sont actuellement appliquées à la ressource.
ou	
--list	

Nom	Description
-a <code>key1=value1[,key2=value2 ...]</code> ou --add <code>key1=value1[,key2=value2 ...]</code>	Appliquez de nouvelles balises à la ressource. Spécifiez les balises sous la forme d'une liste séparée par des virgules et composée de paires key=value. Vous ne pouvez pas spécifier les clés de balises existantes. ...] Valeurs valides : consultez <a href="#">Balisage des ressources (p. 423)</a> .
-u <code>key1=value1[,key2=value2 ...]</code> ou --update <code>key1=value1[,key2=value2 ...]</code>	Mettez à jour les valeurs des balises de ressources existantes. Spécifiez les balises sous la forme d'une liste séparée par des virgules et composée de paires key=value. Vous devez spécifier les clés de balises existantes. ...] Valeurs valides : consultez <a href="#">Balisage des ressources (p. 423)</a> .
-d <code>key1[,key2 ...]</code> ou --delete <code>key1[,key2 ...]</code>	Supprimez des balises de ressources existantes. Spécifiez les balises sous la forme d'une liste de clés séparées par des virgules. Vous devez spécifier les clés de balises existantes. ...] Valeurs valides : consultez <a href="#">Balisage des ressources (p. 423)</a> .
-r <code>région</code> ou --region <code>région</code>	Région Région AWS dans laquelle votre ressource existe. Par défaut : la région configuré par défaut.  Pour obtenir la liste des valeurs que vous pouvez spécifier pour cette option, consultez <a href="#">Points de terminaison et quotas AWS Elastic Beanstalk</a> dans la section Référence générale AWS.
--resource <code>ARN</code>	L'ARN de la ressource que la commande modifie ou pour laquelle elle répertorie les balises. Si elle n'est pas spécifiée, la commande fait référence à l'environnement spécifié (par défaut) de l'application du répertoire actuel.  Valeurs valides : voir l'une des sous-rubriques <a href="#">Balisage des ressources (p. 423)</a> qui est spécifique à la ressource qui vous intéresse. Ces rubriques expliquent comment la ressource de l'ARN est construite et expliquent comment obtenir une liste d'ARN de cette ressource qui existent pour votre application ou votre compte.

## Output

L'option de sous-commande --list affiche la liste des balises de la ressource. La sortie montre à la fois les balises qu'Elastic Beanstalk applique par défaut et vos balises personnalisées.

```
$ eb tags --list
Showing tags for environment 'MyApp-env':

Key                      Value
Name                     MyApp-env
elasticbeanstalk:environment-id  e-63cmxwjaut
elasticbeanstalk:environment-name  MyApp-env
mytag                     tagvalue
tag2                      2nd value
```

Les options de sous-commande `--add`, `--update` et `--delete`, lorsqu'elles aboutissent, n'ont pas de sortie. Vous pouvez ajouter l'option `--verbose` pour voir la sortie détaillée de l'activité de la commande.

```
$ eb tags --verbose --update "mytag=tag value"
Updated Tags:

Key           Value
mytag         tag value
```

## Examples

La commande suivante ajoute une balise avec succès avec la clé `tag1` et la valeur `value1` à l'environnement par défaut de l'application, et, dans le même temps, supprime la balise `tag2`.

```
$ eb tags --add tag1=value1 --delete tag2
```

La commande suivante ajoute une balise avec succès à une configuration enregistrée au sein d'une application.

```
$ eb tags --add tag1=value1 \
--resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:configurationtemplate/my-app/my-template"
```

La commande suivante échoue, car elle essaie de mettre à jour une balise inexistante.

```
$ eb tags --update tag3=newval
ERROR: Tags with the following keys can't be updated because they don't exist:
tag3
```

La commande suivante échoue, car elle essaie de mettre à jour et de supprimer la même clé.

```
$ eb tags --update mytag=newval --delete mytag
ERROR: A tag with the key 'mytag' is specified for both '--delete' and '--update'. Each tag can be either deleted or updated in a single operation.
```

## eb terminate

### Description

Résilie l'environnement en cours d'exécution afin que vous ne payiez pas de frais pour des ressources AWS inutilisées.

Via l'option `--all`, supprime l'application sur laquelle le répertoire actuel a été initialisé à l'aide de [eb init \(p. 1093\)](#). La commande met fin à tous les environnements de l'application. Elle résilie également les [versions d'application \(p. 409\)](#) et les [configurations enregistrées \(p. 779\)](#) de l'application, puis supprime l'application.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande met hors service l'environnement personnalisé en cours d'exécution.

#### Note

Vous pouvez toujours lancer un nouvel environnement en utilisant la même version ultérieurement. Si vous possédez des données d'un environnement que vous souhaitez

conserver, définissez la stratégie de suppression de base de données sur `Retain` avant de résilier l'environnement. Cela permet de maintenir la base de données opérationnelle en dehors d'Elastic Beanstalk. Tous les environnements Elastic Beanstalk doivent ensuite s'y connecter en tant que base de données externe. Si vous souhaitez sauvegarder les données sans maintenir la base de données opérationnelle, définissez la stratégie de suppression pour qu'elle prenne un instantané de la base de données avant de résilier l'environnement. Pour de plus amples informations, consultez [Cycle de vie de base de données \(p. 618\)](#) dans le chapitre Configuration des environnements de ce guide.

## Syntax

`eb terminate`

`eb terminate environment-name`

## Options

Name (Nom)	Description
<code>--all</code>	Met fin à tous les environnements dans l'application, suspend les <a href="#">versions d'application (p. 409)</a> et les <a href="#">configurations enregistrées (p. 779)</a> de l'application, puis supprime l'application.
<code>--force</code>	Résilie l'environnement sans demande de confirmation.
<code>--ignore-links</code>	Résilie l'environnement même s'il existe des environnements dépendants qui y sont liés. Consultez <a href="#">Compose Environments (p. 1058)</a> .
<code>--timeout</code>	Le nombre de minutes avant que la commande expire.

## Output

En cas de réussite, la commande renvoie l'état de l'opération `terminate`.

## Example

L'exemple de demande suivant résilie le tmp-dev DE l'environnement.

```
$ eb terminate
The environment "tmp-dev" and all associated instances will be terminated.
To confirm, type the environment name: tmp-dev
2018-07-11 21:05:25    INFO: terminateEnvironment is starting.
2018-07-11 21:05:40    INFO: Deleted CloudWatch alarm named: awseb-e-2cpfjbra9a-stack-
AWSEBCloudwatchAlarmHigh-16V08YOF2KQ7U
2018-07-11 21:05:41    INFO: Deleted CloudWatch alarm named: awseb-e-2cpfjbra9a-stack-
AWSEBCloudwatchAlarmLow-6ZAWH9F20P7C
2018-07-11 21:06:42    INFO: Deleted Auto Scaling group policy named:
arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:5d7d3e6b-
d59b-47c5-b102-3e11fe3047be:autoScalingGroupName/awseb-e-2cpfjbra9a-stack-
AWSEBAutoScalingGroup-7AXY7U13ZQ6E:policyName/awseb-e-2cpfjbra9a-stack-AWSEBAutoSca
lingScaleUpPolicy-1876U27JEC34J
2018-07-11 21:06:43    INFO: Deleted Auto Scaling group policy named:
arn:aws:autoscaling:us-east-2:11122223333:scalingPolicy:29c6e7c7-7ac8-46fc-91f5-
cfabb65b985b:autoScalingGroupName/awseb-e-2cpfjbra9a-stack-
AWSEBAutoScalingGroup-7AXY7U13ZQ6E:policyName/awseb-e-2cpfjbra9a-stack-AWSEBAutoSca
```

```
lingScaleDownPolicy-SL4LHDMOMU
2018-07-11 21:06:48    INFO: Waiting for EC2 instances to terminate. This may take a few
minutes.
2018-07-11 21:08:55    INFO: Deleted Auto Scaling group named: awseb-e-2cpfjbra9a-stack-
AWSEBAutoScalingGroup-7AXY7U13ZQ6E
2018-07-11 21:09:10    INFO: Deleted security group named: awseb-e-2cpfjbra9a-stack-
AWSEBSecurityGroup-XT4YYGFL7I99
2018-07-11 21:09:40    INFO: Deleted load balancer named: awseb-e-2-AWSEBLoa-AK6RRYFQVV3S
2018-07-11 21:09:42    INFO: Deleting SNS topic for environment tmp-dev.
2018-07-11 21:09:52    INFO: terminateEnvironment completed successfully.
```

## eb upgrade

### Description

Met à niveau la plateforme de votre environnement vers la version la plus récente de la plateforme sur laquelle il est actuellement en cours d'exécution.

Si le répertoire racine contient un fichier `platform.yaml` spécifiant une plateforme personnalisée, cette commande met à niveau l'environnement sur la version la plus récente de la plateforme personnalisée sur laquelle il est actuellement en cours d'exécution.

### Syntax

`eb upgrade`

`eb upgrade environment-name`

### Options

Nom	Description
<code>--force</code>	Met à niveau sans requérir que vous confirmiez le nom de l'environnement avant de commencer le processus de mise à niveau.
<code>--noroll</code>	Met à jour toutes les instances sans utiliser de mises à jour propagées pour maintenir certaines instances en service lors de la mise à niveau.
<a href="#">Options courantes (p. 1123)</a>	

### Output

La commande offre une présentation de la modification et vous invite à confirmer la mise à niveau en tapant le nom de l'environnement. En cas de réussite, votre environnement est mis à jour, puis lancé avec la version la plus récente de la plateforme.

### Example

L'exemple suivant met à niveau la version actuelle de la plateforme de l'environnement spécifié vers la version de la plateforme la plus récente.

```
$ eb upgrade
Current platform: 64bit Amazon Linux 2014.09 v1.0.9 running Python 2.7
```

```
Latest platform: 64bit Amazon Linux 2014.09 v1.2.0 running Python 2.7

WARNING: This operation replaces your instances with minimal or zero downtime. You may
cancel the upgrade after it has started by typing "eb abort".
You can also change your platform version by typing "eb clone" and then "eb swap".

To continue, type the environment name:
```

## eb use

### Description

Définit l'environnement spécifié en tant qu'environnement par défaut.

Lors de l'utilisation de Git, eb use définit l'environnement par défaut pour la branche actuelle. Exécutez cette commande une fois dans chaque branche que vous souhaitez déployer sur Elastic Beanstalk.

### Syntax

eb use **environment-name**

### Options

Nom	Description
--source codecommit/ <i>repository-name</i> / <i>branch-name</i>	Référentiel CodeCommit et branche. Voir <a href="#">Utilisation de l'interface de ligne de commande EB avec AWS CodeCommit (p. 1048)</a> .
-r <i>region</i> --region <i>region</i>	Modifiez la région dans laquelle vous créez les environnements.
Options courantes (p. 1123)	

### Options courantes

Vous pouvez utiliser les options suivantes avec toutes les commandes de l'interface de ligne de commande (CLI) EB.

Nom	Description
--debug	Imprimez des informations pour le débogage.
-h, --help	Affichez le message d'aide.  Type : chaîne  Par défaut : aucun
--no-verify-ssl	Ignorez la vérification du certificat SSL. Utilisez cette option si vous avez des problèmes pour utiliser l'interface de ligne de commande (CLI) avec un proxy.

Nom	Description
--profile	Utilisez un profil spécifique à partir de votre fichier d'informations d'identification AWS.
--quiet	Supprimez toutes les données de la commande.
--region	Utilisez la région spécifiée.
-v, --verbose	Affichez des informations détaillées.

## Interface de ligne de commande EB 2.6 (retirée)

Cette version de l'interface de ligne de commande EB et sa documentation ont été remplacées par la version 3 (dans cette section, EB CLI 3 représente la version 3 ou ultérieure de l'interface de ligne de commande EB). Pour de plus amples informations sur la nouvelle version, veuillez consulter [Utilisation de l'interface de ligne de commande Elastic Beanstalk \(EB\) \(p. 1027\)](#).

Vous devez migrer vers la dernière version de l'interface de ligne de commande EB 3. Elle peut gérer les environnements que vous avez lancés à l'aide de l'interface de ligne de commande EB 2.6 ou de versions antérieures.

## Différences par rapport à la version 3 de l'interface de ligne de commande EB

L'interface de ligne de commande Elastic Beanstalk est une interface de ligne de commande pour Elastic Beanstalk que vous pouvez utiliser pour déployer des applications rapidement et plus facilement. La dernière version d'Elastic Beanstalk a été introduite par Elastic Beanstalk dans l'interface de ligne de commande Elastic Beanstalk 3. L'interface de ligne de commande EB récupère les paramètres d'un environnement créé à l'aide d'EB si cet environnement est en cours d'exécution. Notez que l'interface de ligne de commande EB 3 ne stocke pas les paramètres d'option localement comme le font les versions antérieures.

L'interface de ligne de commande EB introduit les commandes eb create, eb deploy, eb open, eb console, eb scale, eb setenv, eb config, eb terminate, eb clone, eb list, eb use, eb printenv et eb ssh. Dans l'interface de ligne de commande EB 3.1 ou version ultérieure, vous pouvez également utiliser la commande eb swap. Dans l'interface de ligne de commande EB 3.2 uniquement, vous pouvez utiliser les commandes eb abort, eb platform et eb upgrade. En plus de ces nouvelles commandes, les commandes de l'interface de ligne de commande EB 3 diffèrent de celles de l'interface de ligne de commande EB 2.6 dans plusieurs cas :

- eb init : utilisez eb init pour créer un répertoire .elasticbeanstalk dans un répertoire de projet existant et créer une application Elastic Beanstalk pour le projet. Contrairement aux versions précédentes, l'interface de ligne de commande EB 3 et les versions ultérieures ne vous invitent pas à créer un environnement.
- eb start : l'interface de ligne de commande Elastic Beanstalk 3 n'inclut pas la commande eb start. Pour créer un environnement, utilisez eb create.
- eb stop : l'interface de ligne de commande Elastic Beanstalk 3 n'inclut pas la commande eb stop. Utilisez la commande eb terminate pour suspendre un environnement entièrement et le nettoyer.
- eb push et **git aws.push** : l'interface de ligne de commande Elastic Beanstalk 3 n'inclut pas les commandes eb push ou git aws.push. Utilisez eb deploy pour mettre à jour votre code d'application.
- eb update : l'interface de ligne de commande Elastic Beanstalk 3 n'inclut pas la commande eb update. Utilisez eb config pour mettre à jour un environnement.
- eb branch : l'interface de ligne de commande Elastic Beanstalk 3 n'inclut pas la commande eb branch.

Pour plus d'informations sur l'utilisation des commandes de l'interface de ligne de commande EB 3 pour créer et gérer une application, consultez [Guide de référence des commandes de l'interface de ligne de commande \(CLI\) EB \(p. 1062\)](#). Pour connaître la procédure permettant de déployer un exemple d'application à l'aide de l'interface de ligne de commande EB 3, consultez [Gestion des environnements Elastic Beanstalk avec l'interface de ligne de commande EB \(p. 1040\)](#).

## Migration vers l'interface de ligne de commande Elastic Beanstalk 3 et CodeCommit

Elastic Beanstalk 3 a non seulement retiré l'interface de ligne de commande EB 2.6, mais a également supprimé certaines fonctionnalités de la version 2.6. La modification la plus importante par rapport à la version 2.6 est que l'interface de ligne de commande ne prend plus en charge dans sa version native les mises à jour de code incrémentielles (eb push, git aws .push) ou les branches (eb branch). Cette section décrit comment migrer depuis l'interface de ligne de commande EB 2.6 vers la dernière version de l'interface de ligne de commande EB et utiliser CodeCommit comme référentiel de code.

Si vous ne l'avez pas déjà fait, créez un référentiel de code dans CodeCommit, comme décrit dans [Migration vers CodeCommit](#).

Une fois que vous avez [installé \(p. 1028\)](#) et [configuré \(p. 1036\)](#) l'interface de ligne de commande EB, vous avez deux façons d'associer votre application à votre référentiel CodeCommit, en incluant une branche spécifique.

- Lorsque vous exécutez eb init, comme dans l'exemple suivant où `myRepo` est le nom de votre référentiel CodeCommit et `myBranch` la branche dans CodeCommit.

```
eb init --source codecommit/myRepo/myBranch
```

- Lorsque vous exécutez eb deploy, comme dans l'exemple suivant où `myRepo` est le nom de votre référentiel CodeCommit et `myBranch` la branche dans CodeCommit.

```
eb deploy --source codecommit/myRepo/myBranch
```

Pour de plus amples informations, notamment sur la façon de déployer les mises à jour de code incrémentielles dans un environnement Elastic Beanstalk sans avoir à recharger tout votre projet, veuillez consulter [Utilisation de l'interface de ligne de commande EB avec AWS CodeCommit \(p. 1048\)](#).

## Interface de ligne de commande de l'API Elastic Beanstalk (mise hors service)

Cet outil, l'interface de ligne de commande de l'API Elastic Beanstalk, a été remplacé par l'AWS CLI, qui fournit des commandes équivalentes d'API pour tous les services AWS. Consultez le Guide de l'utilisateur AWS Command Line Interface pour commencer à utiliser l'AWS CLI. Essayez également l'[Interface de ligne de commande EB \(p. 1027\)](#) pour bénéficier d'une expérience de ligne de commande simplifiée et de niveau supérieur.

## Conversion des scripts d'interface de ligne de commande de l'API Elastic Beanstalk

Convertissez vos anciens scripts d'interface de ligne de commande de l'API EB afin d'utiliser l'AWS CLI ou Tools for Windows PowerShell pour accéder aux API Elastic Beanstalk les plus récentes. Le tableau ci-

dessous répertorie les commandes de l'interface de ligne de commande basée sur l'API Elastic Beanstalk et les commandes équivalentes dans l'AWS CLI et Tools for Windows PowerShell.

Interface de ligne de commande de l'API Elastic Beanstalk	AWS CLI	AWS Tools for Windows PowerShell
<code>elastic-beanstalk-check-dns-availability</code>	<code>check-dns-availability</code>	<code>Get-EBDNSAvailability</code>
<code>elastic-beanstalk-create-application</code>	<code>create-application</code>	<code>New-EBAApplication</code>
<code>elastic-beanstalk-create-application-version</code>	<code>create-application-version</code>	<code>New-EBAApplicationVersion</code>
<code>elastic-beanstalk-create-configuration-template</code>	<code>create-configuration-template</code>	<code>New-EBConfigurationTemplate</code>
<code>elastic-beanstalk-create-environment</code>	<code>create-environment</code>	<code>New-EBEnvironment</code>
<code>elastic-beanstalk-create-storage-location</code>	<code>create-storage-location</code>	<code>New-EBStorageLocation</code>
<code>elastic-beanstalk-delete-application</code>	<code>delete-application</code>	<code>Remove-EBAApplication</code>
<code>elastic-beanstalk-delete-application-version</code>	<code>delete-application-version</code>	<code>Remove-EBAApplicationVersion</code>
<code>elastic-beanstalk-delete-configuration-template</code>	<code>delete-configuration-template</code>	<code>Remove-EBConfigurationTemplate</code>
<code>elastic-beanstalk-delete-environment-configuration</code>	<code>delete-environment-configuration</code>	<code>Remove-EBEnvironmentConfiguration</code>
<code>elastic-beanstalk-describe-application-versions</code>	<code>describe-application-versions</code>	<code>Get-EBAApplicationVersion</code>
<code>elastic-beanstalk-describe-applications</code>	<code>describe-applications</code>	<code>Get-EBAApplication</code>
<code>elastic-beanstalk-describe-configuration-options</code>	<code>describe-configuration-options</code>	<code>Get-EBConfigurationOption</code>
<code>elastic-beanstalk-describe-configuration-settings</code>	<code>describe-configuration-settings</code>	<code>Get-EBConfigurationSetting</code>

Interface de ligne de commande de l'API Elastic Beanstalk	AWS CLI	AWS Tools for Windows PowerShell
<code>elastic-beanstalk-describe-environment-resources</code>	<code>describe-environment-resources</code>	<code>Get-EBEnvironmentResource</code>
<code>elastic-beanstalk-describe-environments</code>	<code>describe-environments</code>	<code>Get-EBEnvironment</code>
<code>elastic-beanstalk-describe-events</code>	<code>describe-events</code>	<code>Get-EEvent</code>
<code>elastic-beanstalk-list-available-solution-stacks</code>	<code>list-available-solution-stacks</code>	<code>Get-EBAvailableSolutionStack</code>
<code>elastic-beanstalk-rebuild-environment</code>	<code>rebuild-environment</code>	<code>Start-EBEnvironmentRebuild</code>
<code>elastic-beanstalk-request-environment-info</code>	<code>request-environment-info</code>	<code>Request-EBEnvironmentInfo</code>
<code>elastic-beanstalk-restart-app-server</code>	<code>restart-app-server</code>	<code>Restart-EBAppServer</code>
<code>elastic-beanstalk-retrieve-environment-info</code>	<code>retrieve-environment-info</code>	<code>Get-EBEnvironmentInfo</code>
<code>elastic-beanstalk-swap-environment-cnames</code>	<code>swap-environment-cnames</code>	<code>Set-EBEnvironmentCNAME</code>
<code>elastic-beanstalk-terminate-environment</code>	<code>terminate-environment</code>	<code>Stop-EBEnvironment</code>
<code>elastic-beanstalk-update-application</code>	<code>update-application</code>	<code>Update-EApplication</code>
<code>elastic-beanstalk-update-application-version</code>	<code>update-application-version</code>	<code>Update-EApplicationVersion</code>
<code>elastic-beanstalk-update-configuration-template</code>	<code>update-configuration-template</code>	<code>Update-EBConfigurationTemplate</code>
<code>elastic-beanstalk-update-environment</code>	<code>update-environment</code>	<code>Update-EBEnvironment</code>
<code>elastic-beanstalk-validate-configuration-settings</code>	<code>validate-configuration-settings</code>	<code>Test-EBConfigurationSetting</code>

# Sécurité AWS Elastic Beanstalk

Chez AWS, la sécurité dans le cloud est notre priorité numéro 1. En tant que client AWS, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des organisations les plus pointilleuses en termes de sécurité.

La sécurité est une responsabilité partagée entre AWS et vous. Le [modèle de responsabilité partagée](#) décrit ceci comme la sécurité du cloud et la sécurité dans le cloud.

Sécurité du cloud – AWS est responsable de la protection de l'infrastructure qui exécute tous les services proposés dans le cloud AWS et doit vous fournir des services que vous pouvez utiliser en toute sécurité. Chez AWS, notre responsabilité en matière de sécurité est la priorité la plus élevée, et l'efficacité de notre sécurité est régulièrement testée et vérifiée par des auditeurs tiers dans le cadre des [programmes de conformité AWS](#). Consultez la page [Services AWS concernés par le programme de conformité AWS](#) pour obtenir de plus amples informations concernant Elastic Beanstalk.

Sécurité dans le cloud– Votre responsabilité est déterminée par le service AWS que vous utilisez et par d'autres facteurs, notamment la sensibilité de vos données, les exigences de votre organisation, ainsi que les lois et réglementations applicables. Cette documentation est conçue pour vous aider à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Elastic Beanstalk.

Consultez les rubriques de sécurité suivantes pour en savoir plus sur les tâches de sécurité dont Elastic Beanstalk est responsable et les configurations de sécurité que vous devez prendre en compte lorsque vous utilisez Elastic Beanstalk pour atteindre vos objectifs de sécurité et de conformité.

## Rubriques

- [Protection des données dans Elastic Beanstalk \(p. 1128\)](#)
- [Gestion des identités et des accès pour Elastic Beanstalk \(p. 1130\)](#)
- [Journalisation et surveillance dans Elastic Beanstalk \(p. 1134\)](#)
- [Validation de la conformité pour Elastic Beanstalk \(p. 1136\)](#)
- [Résilience dans Elastic Beanstalk \(p. 1136\)](#)
- [Sécurité de l'infrastructure dans Elastic Beanstalk \(p. 1137\)](#)
- [Configuration et analyse des vulnérabilités dans Elastic Beanstalk \(p. 1137\)](#)
- [Bonnes pratiques de sécurité pour Elastic Beanstalk \(p. 1137\)](#)

## Protection des données dans Elastic Beanstalk

Le [modèle de responsabilité partagée](#) AWS s'applique à la protection des données dans AWS Elastic Beanstalk. Comme décrit dans ce modèle, AWS est responsable de la protection de l'infrastructure globale sur laquelle l'ensemble de AWS Cloud s'exécute. La gestion du contrôle de votre contenu hébergé sur cette infrastructure est de votre responsabilité. Ce contenu comprend les tâches de configuration et de gestion de la sécurité des services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez [FAQ sur la confidentialité des données](#). Pour plus d'informations sur la protection des données en Europe, consultez le billet de blog [Modèle de responsabilité partagée AWS et RGPD](#) sur le Blog de sécurité AWS.

À des fins de protection des données, nous vous recommandons de protéger les autorisations du Compte AWS et de configurer les comptes d'utilisateur individuels avec AWS Identity and Access Management

(IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multi-facteurs (MFA) avec chaque compte.
- Utilisez SSL/TLS pour communiquer avec des ressources AWS. Nous recommandons TLS 1.2 ou version ultérieure.
- Configurez l'API et la consignation des activités utilisateur avec AWS CloudTrail.
- Utilisez des solutions de chiffrement AWS, ainsi que tous les contrôles de sécurité par défaut au sein des services AWS.
- Utilisez des services de sécurité gérés avancés tels que Amazon Macie, qui contribuent à la découverte et à la sécurisation des données personnelles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés FIPS 140-2 lorsque vous accédez à AWS via une interface de ligne de commande (CLI) ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS disponibles, consultez [Federal Information Processing Standard \(FIPS\) 140-2](#).

Nous vous recommandons vivement de ne jamais placer d'informations confidentielles ou sensibles, telles que des adresses e-mail, dans des balises ou des champs de format libre tels que Nom. Cela s'applique aussi lorsque vous utilisez Elastic Beanstalk ou d'autres services AWS à l'aide de la console, de l'API, de la AWS CLI ou des kits SDK AWS. Toutes les données que vous entrez dans les balises ou les champs de format libre utilisés pour les noms peuvent être utilisées pour les journaux de facturation ou de diagnostic. Si vous fournissez une URL à un serveur externe, nous vous recommandons fortement de ne pas inclure les informations d'identification non chiffrées dans l'URL pour valider votre demande adressée au serveur.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

#### Rubriques

- [Protection des données à l'aide du chiffrement \(p. 1129\)](#)
- [Confidentialité du trafic inter-réseaux \(p. 1130\)](#)

## Protection des données à l'aide du chiffrement

Elastic Beanstalk stocke divers objets dans un compartiment Amazon Simple Storage Service (Amazon S3) qu'il crée pour chaque région AWS dans laquelle vous créez des environnements. Pour plus d'informations, consultez [the section called "Amazon S3" \(p. 1003\)](#).

Vous fournissez certains des objets stockés, comme les versions d'application et les bundles de fichiers source par exemple, puis les envoyez à Elastic Beanstalk. Elastic Beanstalk génère d'autres objets, comme des fichiers journaux par exemple. Outre les données stockées par Elastic Beanstalk, votre application peut transférer et/ou stocker des données dans le cadre de son fonctionnement.

La protection des données fait référence au fait de protéger les données en transit (lorsqu'elles sont transmises à Elastic Beanstalk ou à partir de ce dernier) et au repos (lorsqu'elles sont stockées dans les centres de données AWS).

### Chiffrement en transit

Vous pouvez obtenir la protection des données en transit de deux manières : chiffrer la connexion à l'aide du protocole SSL (Secure Sockets Layer) ou utiliser le chiffrement côté client (où l'objet est chiffré avant d'être envoyé). Les deux méthodes sont valides pour protéger les données de votre application. Pour sécuriser la connexion, chiffrer-la à l'aide de SSL chaque fois que votre application, ses développeurs et administrateurs, ainsi que ses utilisateurs finaux, envoient ou reçoivent des objets.

Pour plus d'informations sur le chiffrement du trafic web vers et à partir de votre application, consultez [the section called "HTTPS" \(p. 792\)](#).

Le chiffrement côté client n'est pas une méthode valide pour protéger votre code source dans les versions d'application et les bundles de fichiers source que vous chargez. Elastic Beanstalk ayant besoin d'accéder à ces objets, ils ne peuvent pas être chiffrés. Par conséquent, veillez à sécuriser la connexion entre votre environnement de développement ou de déploiement et Elastic Beanstalk.

## Chiffrement au repos

Pour protéger les données de votre application au repos, découvrez la protection des données dans le service de stockage utilisé par votre application. Par exemple, consultez [Protection des données dans Amazon RDS](#) dans le Guide de l'utilisateur Amazon RDS, [Protection des données dans Amazon S3](#) dans le Guide de l'utilisateur Amazon Simple Storage Service, ou [Chiffrement des données et métadonnées dans EFS](#) dans le Guide de l'utilisateur Amazon Elastic File System.

Elastic Beanstalk n'active pas le chiffrement par défaut pour le compartiment Amazon S3 qu'il crée. Cela signifie que, par défaut, les objets sont stockés non chiffrés dans le compartiment (et sont accessibles uniquement par les utilisateurs autorisés à lire le compartiment). Si votre application nécessite le chiffrement au repos, vous pouvez configurer les compartiments de votre compte de sorte que le chiffrement soit activé par défaut. Pour de plus amples informations, consultez [Chiffrement par défaut Amazon S3 pour les compartiments S3](#) dans le Guide de l'utilisateur Amazon Simple Storage Service.

Pour en savoir plus sur la protection des données, consultez le billet de blog [Modèle de responsabilité partagée AWS et RGPD](#) sur le Blog sur la sécurité d'AWS.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

## Confidentialité du trafic inter-réseaux

Vous pouvez utiliser Amazon Virtual Private Cloud (Amazon VPC) pour créer des limites entre les ressources de votre application Elastic Beanstalk et contrôler le trafic entre celles-ci, votre réseau sur site et Internet. Pour plus d'informations, consultez [the section called "Amazon VPC" \(p. 1006\)](#).

Pour de plus amples informations sur la sécurité dans Amazon VPC, veuillez consulter [Sécurité](#) dans le Guide de l'utilisateur Amazon VPC.

Pour en savoir plus sur la protection des données, consultez le billet de blog [Modèle de responsabilité partagée AWS et RGPD](#) sur le Blog sur la sécurité d'AWS.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

## Gestion des identités et des accès pour Elastic Beanstalk

AWS Identity and Access Management (IAM) est un service AWS qui aide un administrateur à contrôler en toute sécurité l'accès aux ressources AWS. Des administrateurs IAM contrôlent les personnes qui s'authentifient (sont connectées) et sont autorisées (disposent d'autorisations) à utiliser des ressources AWS Elastic Beanstalk. IAM est un service AWS que vous pouvez utiliser sans frais supplémentaires.

Pour de plus amples informations sur l'utilisation d'IAM, veuillez consulter [Utilisation d'Elastic Beanstalk avec AWS Identity and Access Management \(p. 920\)](#).

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

## AWS Stratégies gérées par pour AWS Elastic Beanstalk

Pour ajouter des autorisations à des utilisateurs, des groupes et des rôles, il est plus facile d'utiliser des stratégies gérées par AWS que d'écrire des stratégies vous-même. Il faut du temps et de l'expertise pour [Créer des stratégies IAM gérées par le client](#) qui ne fournissent à votre équipe que les autorisations dont elle a besoin. Pour démarrer rapidement, vous pouvez utiliser nos stratégies gérées AWS. Ces stratégies couvrent des cas d'utilisation courants et sont disponibles dans votre Compte AWS . Pour de plus amples informations sur les stratégies gérées par AWS, veuillez consulter [Stratégies gérées par AWS](#) dans le Guide de l'utilisateur IAM.

Les services AWS assurent la maintenance et la mise à jour des stratégies gérées AWS. Vous ne pouvez pas modifier les autorisations définies dans les stratégies gérées par AWS. Les services ajoutent occasionnellement des autorisations à une stratégie gérée par AWS pour prendre en charge de nouvelles fonctions. Ce type de mise à jour affecte toutes les identités (utilisateurs, groupes et rôles) auxquelles la stratégie est attachée. Les services sont très susceptibles de mettre à jour une stratégie gérée AWS quand une nouvelle fonction est lancée ou quand de nouvelles opérations sont disponibles. Les services ne supprimant pas les autorisations d'une stratégie gérée AWS, les mises à jour de stratégie n'interrompent vos autorisations existantes.

En outre, AWS prend en charge des stratégies gérées pour des activités professionnelles couvrant plusieurs services. Par exemple, la politique ViewOnlyAccess gérée par AWS donne accès en lecture seule à beaucoup de services et ressources AWS. Quand un service lance une nouvelle fonction, AWS ajoute des autorisations en lecture seule pour les nouvelles opérations et ressources. Pour obtenir la liste des stratégies de fonctions professionnelles et leurs descriptions, consultez la page [Stratégies gérées par AWS pour les fonctions professionnelles](#) dans le Guide de l'utilisateur IAM.

## Mises à jour Elastic Beanstalk des stratégies gérées par AWS

Afficher des détails sur les mises à jour des stratégies gérées par AWS pour Elastic Beanstalk depuis le 1er mars 2021.

Modification	Description	Date
AWSElasticBeanstalkService – Obsolète	<p>Cette stratégie a été remplacée par <a href="#">AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy</a>.</p> <p>Cette stratégie sera progressivement supprimée à une date ultérieure. Dès que la date sera fixée, elle sera publiée dans ce tableau.</p> <p>Lorsque cette stratégie est progressivement supprimée, elle ne pourra plus être associée à de nouveaux utilisateurs, groupes ou rôles IAM.</p>	À déterminer

Modification	Description	Date
	<p>Pour plus d'informations, veuillez consulter <a href="#">Stratégies de rôles de service gérées (p. 926)</a>.</p>	
<b>AWSElasticBeanstalkReadOnlyAccess</b> – Obsolète GovCloud (US) Région AWS	<p>Cette stratégie a été remplacée par <a href="#">AWSElasticBeanstalkReadOnly</a>.            Cette stratégie sera progressivement supprimée dans la GovCloud (US) Région AWS .            Lorsque cette stratégie est progressivement supprimée, elle ne pourra plus être associée à de nouveaux utilisateurs, groupes ou rôles IAM après le 17 juin 2021.            Pour plus d'informations, veuillez consulter <a href="#">Stratégies utilisateur (p. 946)</a>.</p>	17 juin 2021
<b>AWSElasticBeanstalkManagedUpdate</b> – Mise à jour	<p><del>Les stratégies Read-Only à jour pour permettre à Elastic Beanstalk de lire les attributs des zones de disponibilité EC2.</del> Elle permet à Elastic Beanstalk de fournir une validation plus efficace de votre sélection de type d'instance dans les zones de disponibilité.            Pour plus d'informations, veuillez consulter <a href="#">Stratégies de rôles de service gérées (p. 926)</a>.</p>	16 juin 2021
<b>AWSElasticBeanstalkFullAccess</b> – Obsolète GovCloud (US) Région AWS	<p>Cette stratégie a été remplacée par <a href="#">AdministratorAccess-AWSElasticBeanstalk</a>.            Cette stratégie sera progressivement supprimée dans la GovCloud (US) Région AWS .            Lorsque cette stratégie est progressivement supprimée, elle ne pourra plus être associée à de nouveaux utilisateurs, groupes ou rôles IAM après le 10 juin 2021.            Pour plus d'informations, veuillez consulter <a href="#">Stratégies utilisateur (p. 946)</a>.</p>	10 juin 2021

Modification	Description	Date
<p>Les stratégies gérées suivantes sont devenues obsolètes dans toutes les Région AWS de Chine :</p> <ul style="list-style-type: none"> <li>• AWSElasticBeanstalkFullAccess</li> <li>• AWSElasticBeanstalkReadOnlyAccess</li> </ul>	<p>La stratégie <code>AWSElasticBeanstalkFullAccess</code> a été remplacée par <code>AdministratorAccess-AWSElasticBeanstalk</code>.</p> <p>La stratégie <code>AWSElasticBeanstalkReadOnlyAccess</code> a été remplacée par <code>AWSElasticBeanstalkReadOnly</code>.</p> <p>Ces stratégies ont été progressivement supprimées dans toutes les Région AWS de Chine.</p> <p>Ces stratégies ne pourront plus être associées à de nouveaux utilisateurs, groupes ou rôles IAM après le 3 juin 2021.</p> <p>Pour plus d'informations, veuillez consulter <a href="#">Stratégies utilisateur (p. 946)</a>.</p>	3 juin 2021
<p>Les stratégies gérées suivantes étaient obsolètes dans toutes les Région AWS , à l'exception de la Chine et de GovCloud (US) :</p> <ul style="list-style-type: none"> <li>• AWSElasticBeanstalkFullAccess</li> <li>• AWSElasticBeanstalkReadOnlyAccess</li> </ul>	<p>La stratégie <code>AWSElasticBeanstalkFullAccess</code> a été remplacée par <code>AdministratorAccess-AWSElasticBeanstalk</code>.</p> <p>La stratégie <code>AWSElasticBeanstalkReadOnlyAccess</code> a été remplacée par <code>AWSElasticBeanstalkReadOnly</code>.</p> <p>Ces stratégies ont été progressivement supprimées dans toutes les Région AWS , à l'exception de la Chine et de GovCloud (US).</p> <p>Ces stratégies ne pourront plus être associées à de nouveaux utilisateurs, groupes ou rôles IAM après le 16 avril 2021.</p> <p>Pour plus d'informations, veuillez consulter <a href="#">Stratégies utilisateur (p. 946)</a>.</p>	16 avril 2021

Modification	Description	Date
Les stratégies gérées suivantes ont été mises à jour :	<p>Ces deux stratégies prennent désormais en charge les autorisations PassRole dans les de Chin Régions AWS .</p> <p>Pour plus d'informations sur <code>AdministratorAccess-AWSElasticBeanstalk</code>, veuillez consulter <a href="#">Stratégies utilisateur (p. 946)</a>.</p> <p>Pour plus d'informations sur <code>AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy</code>, veuillez consulter <a href="#">Stratégies de rôles de service gérées (p. 926)</a>.</p>	9 mars 2021
<code>AWSElasticBeanstalkManagedUpdatesCustomerRolePolicy</code> – Nouvelle stratégie	<p><code>Elastic Beanstalk Policy</code> ajouté une nouvelle stratégie en remplacement de la stratégie gérée <code>AWSElasticBeanstalkService</code>.</p> <p>Cette nouvelle stratégie gérée améliore la sécurité de vos ressources en appliquant un jeu d'autorisations plus restrictif.</p> <p>Pour plus d'informations, veuillez consulter <a href="#">Stratégies de rôles de service gérées (p. 926)</a>.</p>	3 mars 2021
Elastic Beanstalk a commencé à assurer le suivi des modifications	Elastic Beanstalk a commencé à assurer le suivi des modifications pour les stratégies gérées par AWS.	1er mars 2021

## Journalisation et surveillance dans Elastic Beanstalk

La surveillance est importante pour assurer la fiabilité, la disponibilité et les performances d'AWS Elastic Beanstalk et de vos solutions AWS. Vous devez recueillir les données de surveillance de toutes les parties de votre solution AWS de manière à pouvoir déboguer plus facilement une éventuelle défaillance à plusieurs points. AWS fournit plusieurs outils pour surveiller vos ressources Elastic Beanstalk et répondre aux incidents potentiels, comme les suivants :

Pour de plus amples informations sur la surveillance, veuillez consulter [Surveillance d'un environnement \(p. 830\)](#).

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

## Création de rapports d'intégrité améliorée

La création de rapports d'intégrité améliorée est une fonctionnalité que vous pouvez activer sur votre environnement pour autoriser Elastic Beanstalk à collecter des informations complémentaires sur les ressources de votre environnement. Elastic Beanstalk analyse les informations recueillies pour fournir une meilleure image de l'intégrité globale de l'environnement et aider à l'identification des problèmes pouvant entraîner une indisponibilité de votre application. Pour plus d'informations, consultez [Surveillance et création de rapports d'intégrité améliorée \(p. 837\)](#).

## Journaux des instances Amazon EC2

Les instances Amazon EC2 de votre environnement Elastic Beanstalk génèrent des journaux que vous pouvez afficher pour résoudre les problèmes avec vos fichiers de configuration ou d'application. Les journaux créés par le serveur web, le serveur d'applications, les scripts de plateforme Elastic Beanstalk et AWS CloudFormation sont stockés localement sur des instances individuelles. Vous pouvez les récupérer facilement avec la [console de gestion d'environnement \(p. 429\)](#) ou l'interface de ligne de commande EB. Vous pouvez également configurer votre environnement pour diffuser en temps réel les journaux dans Amazon CloudWatch Logs. Pour de plus amples informations, veuillez consulter [Affichage des journaux des instances Amazon EC2 dans votre environnement Elastic Beanstalk \(p. 884\)](#).

## Notifications de l'environnement

Vous pouvez configurer votre environnement Elastic Beanstalk pour utiliser Amazon Simple Notification Service (Amazon SNS) et vous informer des événements importants qui affectent votre application. Spécifiez une adresse e-mail pendant ou après la création de l'environnement afin de recevoir des e-mails d'AWS lorsqu'une erreur se produit, ou lorsque l'état de votre environnement change. Pour de plus amples informations, veuillez consulter [Notifications d'environnement Elastic Beanstalk avec Amazon SNS \(p. 644\)](#).

## Alarmes Amazon CloudWatch

Les alarmes Amazon CloudWatch permettent de surveiller une métrique unique sur une période de temps que vous spécifiez. Si la métrique dépasse un seuil donné, une notification est envoyée à une rubrique Amazon SNS ou à une stratégie AWS Auto Scaling. Les alarmes CloudWatch ne déclenchent pas d'actions simplement parce qu'elles se trouvent dans un état particulier. Au lieu de cela, les alarmes appellent des actions lorsque l'état a changé et a été maintenu pendant un certain nombre de périodes. Pour de plus amples informations, veuillez consulter [Utilisation d'Elastic Beanstalk avec Amazon CloudWatch \(p. 894\)](#).

## Journaux AWS CloudTrail

CloudTrail fournit un enregistrement des actions réalisées par un utilisateur, un rôle ou un service AWS dans Elastic Beanstalk. Avec les informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été envoyée à Elastic Beanstalk, l'adresse IP à partir de laquelle la demande a été effectuée, l'auteur et la date de la demande, ainsi que d'autres détails. Pour de plus amples informations, veuillez consulter [Journalisation des appels d'API Elastic Beanstalk avec AWS CloudTrail \(p. 893\)](#).

## Débogage AWS X-Ray

X-Ray est un service AWS qui collecte des données sur les demandes traitées par votre application et les utilise pour construire une cartographie des services qui vous permet d'identifier les problèmes liés à votre application et les opportunités d'optimisation. Vous pouvez utiliser la console AWS Elastic Beanstalk ou un fichier de configuration pour exécuter le démon X-Ray sur les instances de votre environnement. Pour de plus amples informations, veuillez consulter [Configuration du débogage AWS X-Ray \(p. 638\)](#).

## Validation de la conformité pour Elastic Beanstalk

La sécurité et la conformité d'AWS Elastic Beanstalk sont évaluées par des auditeurs tiers dans le cadre de plusieurs programmes de conformité AWS. Il s'agit notamment des certifications SOC, PCI, FedRAMP, HIPAA et autres. AWS fournit une liste mise à jour régulièrement des services AWS concernés par des programmes de conformité spécifiques sur la page [Services AWS concernés par le programme de conformité](#).

Les rapports d'audit tiers sont disponibles au téléchargement à l'aide de AWS Artifact. Pour plus d'informations, consultez [Téléchargement des rapports dans AWS Artifact](#).

Pour de plus amples informations sur les programmes de conformité AWS, veuillez consulter [Programmes de conformité AWS](#).

Votre responsabilité de conformité lors de l'utilisation d'Elastic Beanstalk est déterminée par la sensibilité de vos données, les objectifs de conformité de votre organisation, ainsi que par la législation et la réglementation applicables. Si votre utilisation d'Elastic Beanstalk doit être conforme aux normes HIPAA, PCI, ou FedRAMP, AWS fournit des ressources pour vous aider :

- [Guides de démarrage rapide de la sécurité et de la conformité](#)— Guides de déploiement qui proposent des considérations architecturales et fournissent des procédures pour déployer des environnements de référence centrés sur la sécurité et la conformité sur AWS.
- [Livre blanc sur l'architecture pour la sécurité et la conformité HIPAA](#)— Livre blanc qui décrit comment les entreprises peuvent utiliser AWS pour créer des applications conformes à la loi HIPAA.
- [Ressources de conformité AWS](#) – Ensemble de manuels et de guides de conformité susceptibles de s'appliquer à votre secteur et à votre emplacement.
- [AWS Config](#) – Service qui permet d'évaluer comment les configurations de vos ressources se conforment aux pratiques internes, aux normes et aux directives industrielles.
- [AWS Security Hub](#) – Vue complète de l'état de votre sécurité au sein d'AWS, qui vous permet de vérifier votre conformité aux normes du secteur et aux bonnes pratiques de sécurité.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

## Résilience dans Elastic Beanstalk

L'infrastructure mondiale AWS repose sur les régions et les zones de disponibilité AWS.

Les régions AWS fournissent plusieurs zones de disponibilité physiquement séparées et isolées, reliées par des réseaux à faible latence, ainsi qu'à débit et à redondance élevés.

Avec les zones de disponibilité, vous pouvez concevoir et exploiter des applications et des bases de données qui basculent automatiquement d'une zone de disponibilité à l'autre sans interruption. Les zones de disponibilité sont plus hautement disponibles, tolérantes aux pannes et évolutives que les infrastructures traditionnelles à un ou plusieurs centres de données.

Pour plus d'informations sur les régions et les zones de disponibilité AWS, consultez [Infrastructure mondiale AWS](#).

AWS Elastic Beanstalk gère et automatise l'utilisation de l'infrastructure AWS globale en votre nom. Lorsque vous utilisez Elastic Beanstalk, vous bénéficiez des mécanismes de disponibilité et de tolérance aux pannes offerts par AWS.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

## Sécurité de l'infrastructure dans Elastic Beanstalk

En tant que service géré, AWS Elastic Beanstalk est protégé par les procédures de sécurité du réseau mondial AWS qui sont décrites dans le livre blanc [Amazon Web Services : Présentation des procédures de sécurité](#).

Vous utilisez les appels d'API publiés AWS pour accéder à Elastic Beanstalk via le réseau. Les clients doivent prendre en charge le protocole TLS (Transport Layer Security) 1.0 ou version ultérieure. Nous recommandons TLS 1.2 ou version ultérieure. Les clients doivent également prendre en charge les suites de chiffrement PFS (Perfect Forward Secrecy) comme Ephemeral Diffie-Hellman (DHE) ou Elliptic Curve Ephemeral Diffie-Hellman (ECDHE). La plupart des plateformes modernes telles que Java 7 et versions ultérieures prennent en charge ces modes.

En outre, les demandes doivent être signées à l'aide d'un ID de clé d'accès et d'une clé d'accès secrète associée à un mandataire IAM. Vous pouvez également utiliser [AWS Security Token Service](#) (AWS STS) pour générer des informations d'identification de sécurité temporaires afin de signer les demandes.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

## Configuration et analyse des vulnérabilités dans Elastic Beanstalk

AWS et nos clients partagent la responsabilité d'atteindre un niveau élevé de sécurité et de conformité des composants logiciels. AWS Elastic Beanstalk vous aide à assurer votre part du modèle de responsabilité partagée en fournissant une fonction de mises à jour gérées. Cette fonction applique automatiquement les mises à jour correctives et mineures correspondant à une version de plateforme prise en charge par Elastic Beanstalk.

Pour plus d'informations, consultez [Modèle de responsabilité partagée pour la maintenance de la plateforme Elastic Beanstalk \(p. 27\)](#).

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

## Bonnes pratiques de sécurité pour Elastic Beanstalk

AWS Elastic Beanstalk fournit différentes fonctionnalités de sécurité à prendre en compte lorsque vous développez et mettez en œuvre vos propres stratégies de sécurité. Les bonnes pratiques suivantes doivent être considérées comme des instructions générales et ne représentent pas une solution de sécurité complète. Étant donné que ces bonnes pratiques peuvent ne pas être appropriées ou suffisantes pour votre environnement, considérez-les comme des considérations utiles et non comme des recommandations.

Pour consulter d'autres rubriques de sécurité Elastic Beanstalk, veuillez vous reporter à [Sécurité AWS Elastic Beanstalk \(p. 1128\)](#).

### Bonnes pratiques de sécurité préventive

Les contrôles de sécurité préventifs tentent d'éviter les incidents avant qu'ils ne se produisent.

## Implémentation d'un accès sur la base du moindre privilège

Elastic Beanstalk fournit des stratégies gérées AWS Identity and Access Management (IAM) pour les profils d'instance (p. 921), les fonctions du service (p. 926) et les utilisateurs IAM (p. 946). Ces stratégies gérées spécifient toutes les autorisations qui peuvent être nécessaires au bon fonctionnement de votre environnement et de votre application.

Votre application peut ne pas exiger toutes les autorisations de nos stratégies gérées. Vous pouvez les personnaliser et accorder uniquement les autorisations requises pour que les instances de votre environnement, le service Elastic Beanstalk et vos utilisateurs puissent effectuer leurs tâches. C'est particulièrement pertinent pour les stratégies utilisateur, où différents rôles utilisateur peuvent avoir des besoins différents en matière d'autorisations. L'implémentation d'un accès sur la base du moindre privilège est fondamentale pour réduire les risques en matière de sécurité et l'impact que pourraient avoir des erreurs ou des actes de malveillance.

## Mise à jour régulière de vos plateformes

Elastic Beanstalk publie régulièrement de nouvelles versions de plateforme pour mettre à jour toutes ses plateformes. Les nouvelles versions de plateforme fournissent des mises à jour du système d'exploitation, de l'environnement d'exécution, du serveur d'applications et du serveur web, ainsi que des mises à jour des composants Elastic Beanstalk. Bon nombre de ces mises à jour de plateforme incluent des correctifs de sécurité importants. Assurez-vous que vos environnements Elastic Beanstalk s'exécutent sur une version de plateforme prise en charge (généralement la dernière version de votre plateforme). Pour plus d'informations, consultez [Mise à jour de la version de la plateforme de votre environnement Elastic Beanstalk \(p. 496\)](#).

Le moyen le plus simple de maintenir à jour la plateforme de votre environnement consiste à configurer l'environnement pour utiliser les [mises à jour gérées de la plateforme \(p. 501\)](#).

## Appliquer IMDsv2 sur les instances d'environnement

Les instances Amazon Elastic Compute Cloud (Amazon EC2) de vos environnements Elastic Beanstalk utilisent le service de métadonnées d'instance (IMDS), un composant sur instance, pour accéder en toute sécurité aux métadonnées d'instance. IMDS prend en charge deux méthodes d'accès aux données : IMDSv1 et IMDSv2. IMDSv2 utilise des requêtes orientées session et atténue plusieurs types de vulnérabilités qui pourraient être utilisées pour essayer d'accéder à l'IMDS. Pour de plus amples informations sur les avantages d'IMDsv2, veuillez consulter les [améliorations apportées pour ajouter une défense en profondeur au service de métadonnées d'instance EC2](#).

IMDSv2 est plus sécurisé, il est donc conseillé d'appliquer l'utilisation d'IMDSv2 sur vos instances. Pour appliquer IMDsv2, assurez-vous que tous les composants de votre application prennent en charge IMDSv2, puis désactivez IMDSv1. Pour plus d'informations, consultez [the section called "IMDS" \(p. 545\)](#).

## Bonnes pratiques de sécurité de détection

Les contrôles de sécurité de détection identifient les violations de sécurité après qu'elles se sont produites. Ils peuvent vous aider à détecter une menace ou un incident de sécurité potentiel.

## Mise en œuvre de la surveillance

La surveillance constitue une part importante de la gestion de la fiabilité, de la disponibilité et des performances de vos solutions Elastic Beanstalk. AWS fournit différents outils et services que vous pouvez utiliser pour surveiller vos services AWS.

Voici quelques exemples d'éléments à surveiller :

- Métriques Amazon CloudWatch pour Elastic Beanstalk : définissez les alarmes pour les principales métriques Elastic Beanstalk et pour les métriques personnalisées de votre application. Pour plus d'informations, consultez [Utilisation d'Elastic Beanstalk avec Amazon CloudWatch \(p. 894\)](#).
- Entrées AWS CloudTrail – Suivez les actions qui peuvent avoir un impact sur la disponibilité, comme `UpdateEnvironment` ou `TerminateEnvironment`. Pour plus d'informations, consultez [Journalisation des appels d'API Elastic Beanstalk avec AWS CloudTrail \(p. 893\)](#).

## Activer AWS Config

AWS Config fournit une vue détaillée de la configuration des ressources AWS de votre compte. Vous pouvez voir comment les ressources sont liées, obtenir un historique des changements de configuration, et voir comment les configurations et les relations changent au fil du temps.

Vous pouvez utiliser AWS Config pour définir des règles qui évaluent les configurations de ressources pour assurer la conformité des données. Les règles AWS Config représentent les paramètres de configuration idéaux pour vos ressources Elastic Beanstalk. Si une ressource enfreint une règle et est signalée comme non conforme, AWS Config peut vous alerter à l'aide d'une rubrique Amazon Simple Notification Service (Amazon SNS). Pour plus d'informations, consultez [Recherche et suivi des ressources Elastic Beanstalk avec AWS Config \(p. 913\)](#).

# Dépannage

Ce chapitre inclut un tableau des problèmes les plus courants dans Elastic Beanstalk et des solutions permettant de les résoudre ou de les contourner. Des messages d'erreur peuvent s'afficher comme des événements dans la page de gestion d'environnement de la console, dans les journaux ou sur la page Health.

Si l'intégrité de votre environnement passe au rouge, essayez les éléments suivants :

- Consultez les [événements \(p. 879\)](#) récents de l'environnement. Les messages d'Elastic Beanstalk sur les problèmes de déploiement, de charge et de configuration s'affichent souvent ici.
- Examinez l'[historique des modifications \(p. 877\)](#) d'environnement récents. L'historique des modifications répertorie toutes les modifications apportées à la configuration de vos environnements et inclut d'autres informations, telles que l'utilisateur IAM ayant apporté les modifications et les paramètres de configuration définis.
- [Extrayez des journaux \(p. 884\)](#) afin d'afficher de récentes entrées du fichier journal. Les journaux de serveur web contiennent des informations sur les erreurs et les demandes entrantes.
- [Connectez-vous à une instance \(p. 881\)](#) et vérifiez les ressources du système.
- [Restaurez \(p. 476\)](#) sur une version de travail précédente de l'application.
- Annulez les récentes modifications de configuration ou restaurez une [configuration sauvegardée \(p. 663\)](#).
- Déployez un nouvel environnement. Si l'environnement s'avère sain, effectuez un [échange CNAME \(p. 486\)](#) pour acheminer le trafic vers le nouvel environnement et continuer à déboguer l'ancien.

## Rubriques

- [Connectivity \(p. 1140\)](#)
- [Création de l'environnement et lancements d'instance \(p. 1141\)](#)
- [Déploiements \(p. 1142\)](#)
- [Santé \(p. 1142\)](#)
- [Configuration \(p. 1142\)](#)
- [Résolution des problèmes de conteneurs Docker \(p. 1143\)](#)
- [FAQ \(p. 1144\)](#)

## Connectivity

### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021. Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou mettez à jour vos plateformes manuellement ([p. 498](#)). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

Problème : Impossible de se connecter à Amazon RDS à partir d'Elastic Beanstalk.

Pour connecter RDS à votre application Elastic Beanstalk, procédez comme suit :

- Assurez-vous que RDS est dans la même région que votre application Elastic Beanstalk.
- Assurez-vous que le groupe de sécurité RDS de votre instance dispose d'une autorisation concernant le groupe de sécurité Amazon EC2 que vous utilisez pour votre environnement Elastic Beanstalk. Pour savoir comment trouver le nom de votre groupe de sécurité EC2 via la console de gestion AWS, consultez [Groupes de sécurité \(p. 542\)](#). Pour de plus amples informations sur la configuration de votre groupe de sécurité EC2, veuillez consulter la section « Autoriser un accès réseau à un groupe de sécurité Amazon EC2 » de la page [Utilisation de groupes de sécurité de base de données](#) dans le Guide de l'utilisateur Amazon Relational Database Service.
- Pour Java, assurez-vous que le fichier JAR MySQL se trouve dans votre répertoire WEB-INF/lib. Pour plus d'informations, consultez [Ajout d'une instance de base de données Amazon RDS à votre environnement d'application Java \(p. 136\)](#).

Problème : Les serveurs créés dans la console Elastic Beanstalk n'apparaissent pas dans Toolkit for Eclipse

Vous pouvez importer des serveurs manuellement en suivant les instructions figurant sur la page [Importation d'environnements existants dans Eclipse \(p. 142\)](#).

## Création de l'environnement et lancements d'instance

Événement : Échec du lancement de l'environnement

Cet événement se produit lorsque Elastic Beanstalk tente de lancer un environnement et rencontre des défaiillances ce faisant. Les événements précédents sur la page Événements vous alertent de la cause racine de ce problème.

Événement : L'opération de création de l'environnement est terminée, mais avec des délais de commande. Essayez d'augmenter le délai d'expiration.

Le déploiement de votre application peut prendre beaucoup de temps si vous utilisez des fichiers de configuration qui exécutent des commandes sur l'instance, téléchargent des fichiers volumineux ou installent des packages. Augmentez le [délai de commande \(p. 480\)](#) afin que votre application dispose de plus de temps pour commencer son exécution pendant les déploiements.

Événement : Échec de la création des ressources suivantes : [AWSEBInstanceLaunchWaitCondition]

Ce message indique que les instances Amazon EC2 de votre environnement n'ont pas communiqué à Elastic Beanstalk qu'elles ont été lancées avec succès. Cette erreur peut se produire si ces instances n'ont pas de connectivité Internet. Si vous avez configuré votre environnement pour lancer des instances sur un sous-réseau VPC privé, [vérifiez que ce sous-réseau possède un NAT \(p. 1006\)](#) pour permettre aux instances de se connecter à Elastic Beanstalk.

Événement : Un rôle de service est requis dans cette région. Ajoutez une option Service Role à l'environnement.

Elastic Beanstalk utilise un rôle de service pour surveiller les ressources de votre environnement et prendre en charge [les mises à jour de la plateforme gérée \(p. 501\)](#). Pour de plus amples informations, veuillez consulter [Gestion des rôles de service Elastic Beanstalk \(p. 926\)](#).

## Déploiements

Problème : L'application devient indisponible pendant les déploiements

Un temps d'arrêt de quelques secondes est possible, car Elastic Beanstalk utilise un processus de mise à niveau par remplacement direct. Utilisez des [propagations de déploiements \(p. 479\)](#) pour minimiser l'impact des déploiements sur vos environnements de production.

Événement : Impossible de créer la version de l'application AWSElastic Beanstalk

L'ensemble de sources de votre application est peut-être trop volumineux ou vous avez peut-être atteint le [quota des versions d'application \(p. 409\)](#).

Événement : L'opération de mise à jour de l'environnement est terminée, mais avec des délais de commande. Essayez d'augmenter le délai d'expiration.

Le déploiement de votre application peut prendre beaucoup de temps si vous utilisez des fichiers de configuration qui exécutent des commandes sur l'instance, téléchargent des fichiers volumineux ou installent des packages. Augmentez le [délai de commande \(p. 480\)](#) afin que votre application dispose de plus de temps pour commencer son exécution pendant les déploiements.

## Santé

Événement : L'utilisation de l'UC dépasse 95 %

Essayez [d'exécuter davantage d'instances \(p. 547\)](#) ou [choisissez un type d'instance différent \(p. 538\)](#).

Événement : Elastic Load Balancer awseb-*myapp* a zéro instances saines

Si votre application semble fonctionner, assurez-vous que l'URL de vérification de l'état de votre application est correctement configurée. Sinon, consultez l'écran Health et les journaux de l'environnement pour obtenir plus d'informations.

Événement : Elastic Load Balancer awseb-*myapp* introuvable

L'équilibrEUR de charge de votre environnement a peut-être été supprimé hors bande. Ne modifiez les ressources de votre environnement qu'avec les options de configuration et l'[extensibilité \(p. 737\)](#) fournies par Elastic Beanstalk. Recréez votre environnement ou lancez-en un nouveau.

Événement : Échec du lancement de l'instance EC2. Lancement d'une nouvelle instance EC2 en attente...

La disponibilité du type d'instance de votre environnement peut être faible ou vous avez peut-être atteint le quota d'instances de votre compte. Vérifiez le [tableau de bord de l'état des services](#) pour vous assurer que le service Elastic Compute Cloud (Amazon EC2) est vert ou [demandez une augmentation de quota](#).

## Configuration

### Important

Le certificat Let's Encrypt à signature croisée DST Root CA X3 a expiré le 30 septembre 2021. Pour cette raison, les environnements Beanstalk exécutés sur les systèmes d'exploitation

Amazon Linux 2 et Amazon Linux AMI peuvent ne pas être en mesure de se connecter à des serveurs à l'aide des certificats Let's Encrypt.

Le 3 octobre 2021, Elastic Beanstalk a publié de nouvelles versions de plateforme pour Amazon Linux AMI et Amazon Linux 2 et mis à jour les certificats d'autorité de certification. Pour recevoir ces mises à jour et résoudre ce problème, activez [Mises à jour gérées \(p. 501\)](#) ou [mettez à jour vos plateformes manuellement \(p. 498\)](#). Pour plus d'informations, consultez les [notes de mise à jour de plateforme](#) dans les Notes de mise à jour AWS Elastic Beanstalk.

Vous pouvez également appliquer les solutions de contournement manuelles décrites dans cet [article du Centre de connaissances AWS](#). Étant donné qu'Elastic Beanstalk propose des AMI avec des GUID verrouillés, nous vous recommandons d'utiliser la commande sudo yum install reprise dans l'article. Vous pouvez également utiliser la commande sudo sed reprise dans l'article si vous préférez modifier manuellement le système en place.

Événement : Vous ne pouvez pas configurer un environnement Elastic Beanstalk avec des valeurs pour les options Elastic Load Balancing Target (Cible Elastic Load Balancing) et Application Healthcheck URL (URL de vérification de l'état de l'application)

L'option `Target` dans l'espace de noms `aws:elb:healthcheck` est obsolète. Supprimez l'option `Target` de votre environnement et essayez à nouveau de mettre à jour.

Événement : Impossible d'attacher ELB à plusieurs sous-réseaux de la même zone de disponibilité.

Ce message s'affiche si vous tentez de déplacer un équilibrEUR de charge entre des sous-réseaux d'une même zone de disponibilité. La modification de sous-réseaux sur l'équilibrEUR de charge nécessite de le faire sortir en dehors des zones de disponibilité, puis de le réintégrer dans la zone de disponibilité d'origine avec les sous-réseaux souhaités. Au cours du processus, toutes vos instances sont migrées entre les zones de disponibilité, ce qui entraîne un temps d'arrêt important. Pensez plutôt à créer un nouvel environnement et [effectuez un échange CNAME \(p. 486\)](#).

## Résolution des problèmes de conteneurs Docker

Événement : Échec de l'extraction de l'image Docker : la plus récente : Nom de référentiel () non valide, seulement [a-z0-9-\_] sont autorisés. Consultez les journaux pour plus d'informations.

Vérifiez la syntaxe du fichier `dockerrun.aws.json` à l'aide d'un validateur JSON. Vérifiez également le contenu `Dockerfile` par rapport aux conditions décrites dans [Configuration Docker \(p. 53\)](#)

Événement : Aucune directive EXPOSE trouvée dans le `Dockerfile`, abandon du déploiement

Le `Dockerfile` ou le fichier `dockerrun.aws.json` ne déclare pas le port de conteneur. Utilisez l'instruction `EXPOSE` (`Dockerfile`) ou le bloc `Ports` (fichier `dockerrun.aws.json`) pour exposer un port pour le trafic entrant.

Événement : Échec du téléchargement des informations d'authentification `référentiel` à partir de `nom du compartiment`

Le fichier `dockerrun.aws.json` fournit une paire de clés EC2 non valide et/ou un compartiment S3 pour le fichier `.dockercfg`. Ou, le profil d'instance n'a pas d'autorisation `GetObject` pour le compartiment S3. Vérifiez que le fichier `.dockercfg` contient un compartiment S3 et une paire de clés EC2 valides. Accordez des autorisations pour l'action `s3:GetObject` au rôle IAM dans le profil d'instance. Pour obtenir des détails, consultez [Gestion des profils d'instance Elastic Beanstalk \(p. 921\)](#)

Événement : L'exécution de l'activité a échoué, car : AVERTISSEMENT : Fichier de configuration d'authentification non valide

Le format de votre fichier d'authentification (`config.json`) n'est pas correct. Veuillez consulter [Utilisation d'images à partir d'un référentiel privé \(p. 90\)](#).

## FAQ

Question : Comment remplacer l'URL de mon application myapp.us-west-2.elasticbeanstalk.com par www.myapp.com ?

Dans un serveur DNS, enregistrez un enregistrement CNAME tel que **www.mydomain.com CNAME mydomain.elasticbeanstalk.com**.

Question : Comment spécifier une zone de disponibilité spécifique pour mon application Elastic Beanstalk ?

Vous pouvez choisir une zone de disponibilité spécifique via les API, l'interface de ligne de commande, le plug-in Eclipse ou le plug-in Visual Studio. Pour de plus amples informations sur l'utilisation de la console Elastic Beanstalk pour spécifier une zone de disponibilité, veuillez consulter [Groupe Auto Scaling pour votre environnement Elastic Beanstalk \(p. 547\)](#).

Question : Comment modifier le type d'instance de mon environnement ?

Pour modifier le type d'instance de votre environnement, accédez à la page de configuration de l'environnement et sélectionnez Edit (Modifier) dans la catégorie de configuration Instances. Ensuite, sélectionnez un nouveau type d'instance, puis choisissez Apply (Appliquer) pour mettre à jour votre environnement. Après cela, Elastic Beanstalk résilie toutes les instances en cours d'exécution et les remplace par de nouvelles.

Question : Comment déterminer si quelqu'un a apporté des modifications de configuration à un environnement ?

Pour afficher ces informations, dans le volet de navigation de la console Elastic Beanstalk, sélectionnez Change history (Historique des modifications) pour afficher la liste des modifications de configuration de tous les environnements. Cette liste inclut la date et l'heure de la modification, le paramètre de configuration modifié et sa nouvelle valeur, ainsi que l'utilisateur IAM qui a effectué la modification. Pour plus d'informations, consultez [Historique des modifications \(p. 877\)](#).

Question : Puis-je éviter que les volumes EBS soient supprimés lorsque les instances sont mises hors service ?

Les instances de votre environnement utilisent Amazon EBS pour le stockage. Toutefois, le volume racine est supprimé lorsqu'une instance est résiliée par Auto Scaling. Nous vous recommandons de ne pas stocker les données d'état ou d'autres informations dans vos instances. Si nécessaire, vous pouvez éviter la suppression des volumes via l'AWS CLI : \$ aws ec2 modify-instance-attribute -b '/dev/sdc=<vol-id>:false', comme indiqué dans la [référence de l'AWS CLI](#).

Question : Comment supprimer des informations personnelles de mon application Elastic Beanstalk ?

Les ressources AWS que votre application Elastic Beanstalk utilise peuvent stocker des informations personnelles. Lorsque vous arrêtez un environnement, Elastic Beanstalk arrête les ressources qu'il a créées. Les ressources que vous avez ajoutées à l'aide de [fichiers de configuration \(p. 737\)](#) sont également résiliées. Toutefois, si vous avez créé des ressources AWS en dehors de votre environnement Elastic Beanstalk et que vous les avez associées à votre application, il se peut que vous ayez besoin de vérifier manuellement que les informations personnelles stockées par votre application ne sont pas conservées. Tout au long de ce manuel du développeur, chaque fois que nous abordons la création de ressources supplémentaires, nous mentionnons également le moment auquel vous devez envisager de les supprimer.

# Ressources Elastic Beanstalk

Les ressources connexes suivantes peuvent s'avérer utiles lors de l'utilisation de ce service.

- [Référence d'API Elastic Beanstalk](#) Description complète de toutes les API SOAP et Query. Cette référence contient également une liste de tous les types de données SOAP.
- [elastic-beanstalk-samples sur GitHub](#) – Référentiel GitHub avec des exemples de fichiers de configuration (.ebextensions). Le fichier `README.md` du référentiel possède des liens vers d'autres référentiels GitHub avec des exemples d'application.
- [FAQ technique Elastic Beanstalk](#) – Les principales questions posées par les développeurs à propos de ce produit.
- [Notes de mise à jour d'AWS Elastic Beanstalk](#) – Détails sur les nouvelles fonctionnalités, mises à jour et correctifs liés au service Elastic Beanstalk, à la plateforme, à la CLI EB.
- [Formations et ateliers](#) – Liens vers des formations spécialisées et basées sur les rôles, en plus des ateliers d'autoformation pour améliorer vos compétences AWS et acquérir une expérience pratique.
- [Outils pour développeur AWS](#) : liens vers des outils pour développeur, kits SDK, boîtes à outils IDE et outils de ligne de commande pour développer et gérer des applications AWS.
- [Livres blancs AWS](#) : liens vers une liste complète des livres blancs techniques AWS, couvrant des sujets tels que l'architecture, la sécurité et l'économie, créés par des architectes de solutions AWS ou d'autres experts techniques.
- [Centre AWS Support](#) : hub pour la création et la gestion de vos cas AWS Support. Inclut également des liens vers d'autres ressources utiles, telles que des forums, des FAQ techniques, l'état de santé d'un service et AWS Trusted Advisor.
- [AWS Support](#) : principale page web d'informations à propos d'AWS Support, un canal d'assistance technique individuelle rapide pour vous aider à développer et à exécuter des applications dans le cloud.
- [Contactez-nous](#) : point de contact central pour toute question relative à la facturation AWS, à votre compte, aux événements, à des abus ou à d'autres problèmes.
- [AWSConditions d'utilisation du site](#) : informations détaillées sur nos droits d'auteur et notre marque, sur votre compte, licence et accès au site, et sur d'autres sujets.

## Exemples d'applications

Vous trouverez ci-dessous des liens permettant de télécharger les exemples d'application déployés dans le cadre de [Mise en route avec Elastic Beanstalk \(p. 3\)](#).

### Note

Certains exemples utilisent des fonctions qui ont peut-être été publiées après le lancement de la plateforme que vous utilisez. Si l'exécution de l'exemple échoue, tentez de mettre à jour votre plateforme vers une version actuelle, comme décrit dans [the section called “Plateformes prises en charge” \(p. 31\)](#).

- Docker – [docker.zip](#)
- Docker multi-conteneurs – [docker-multicontainer-v2.zip](#)
- Docker avec plateforme préconfigurée (Glassfish) – [docker-glassfish-v1.zip](#)
- Go – [go.zip](#)
- Corretto – [corretto.zip](#)

- Tomcat – [tomcat.zip](#)
- .NET Core sous Linux – [dotnet-core-linux.zip](#)
- .NET – [dotnet-asp-v1.zip](#)
- Node.js – [nodejs.zip](#)
- PHP – [php.zip](#)
- Python – [python.zip](#)
- Ruby – [ruby.zip](#)

# Historique de la plateforme

L'historique de la plateforme AWS Elastic Beanstalk a été déplacé. Consultez [Historique de la plateforme](#) dans le document Plateformes AWS Elastic Beanstalk.

## Rubriques

- [Plateformes personnalisées Elastic Beanstalk \(p. 1147\)](#)

## Plateformes personnalisées Elastic Beanstalk

AWS Elastic Beanstalk prend en charge les plateformes personnalisées. Une plateforme personnalisée est à plusieurs titres une personnalisation plus avancée qu'une [image personnalisée \(p. 787\)](#). Une plateforme personnalisée vous permet de développer une plateforme entièrement nouvelle avec un système d'exploitation personnalisé, des logiciels supplémentaires et des scripts exécutés par Elastic Beanstalk sur les instances de la plateforme. Cette souplesse vous permet de construire une plateforme pour une application qui utilise un langage ou d'autres logiciels d'infrastructure pour lesquels Elastic Beanstalk ne fournit pas de plateforme gérée. Cela peut se comparer aux images personnalisées où une AMI est modifiée pour être utilisée avec une plateforme Elastic Beanstalk existante, et Elastic Beanstalk fournit toujours les scripts de plateforme et contrôle la pile logicielle de la plateforme. Par ailleurs, avec les plateformes personnalisées, vous créez et tenez à jour vos personnalisations de façon automatisée et scriptée, alors qu'avec les images personnalisées, vous apportez les modifications manuellement sur une instance en cours d'exécution.

### Note

Les plateformes personnalisées Elastic Beanstalk prennent uniquement en charge la création d'une AMI à partir de l'AMI Amazon Linux, de RHEL 7, RHEL 6 ou des AMI de base Ubuntu 16.04. La fonctionnalité des plateformes personnalisées ne prend pas en charge Amazon Linux 2 ou d'autres systèmes d'exploitation.

Pour créer une plateforme personnalisée, vous devez générer une AMI à partir de l'un des systèmes d'exploitation pris en charge, Ubuntu, RHEL ou Amazon Linux (consultez l'entrée `flavor` dans [Format de fichier platform.yaml \(p. 1155\)](#) pour connaître les numéros de version exacts), puis ajouter d'autres personnalisations. Vous créez votre propre plateforme Elastic Beanstalk à l'aide de [Packer](#), qui est un outil open source permettant de créer des images de machines pour de nombreuses plateformes, y compris des AMI à utiliser avec Amazon Elastic Compute Cloud (Amazon EC2). Une plateforme Elastic Beanstalk comprend une AMI configurée pour exécuter un ensemble de logiciels prenant en charge une application, et des métadonnées qui peuvent inclure des options de configuration personnalisées et des paramètres d'option de configuration par défaut.

Elastic Beanstalk gère Packer comme une plateforme intégrée distincte. Autrement dit, vous n'avez pas besoin de vous soucier de la configuration et des versions de Packer.

Vous créez une plateforme en fournissant un modèle Packer à Elastic Beanstalk, ainsi que les scripts et les fichiers que le modèle appelle pour créer une AMI. Ces composants sont fournis avec un [fichier de définition de plateforme \(p. 1148\)](#), qui spécifie le modèle et les métadonnées, dans une archive ZIP appelée [archive de définition de plateforme \(p. 1153\)](#).

Lorsque vous créez une plateforme personnalisée, vous lancez un environnement d'instance unique sans aucune adresse IP élastique pour exécuter Packer. Packer lance alors une autre instance pour générer

une image. Vous pouvez réutiliser cet environnement pour plusieurs plateformes et plusieurs versions de chaque plateforme.

#### Note

Les plateformes personnalisées sont spécifiques à une région AWS. Si vous utilisez Elastic Beanstalk dans plusieurs régions, vous devez créer vos plateformes séparément dans chaque région.

Dans certaines circonstances, les instances lancées par Packer ne sont pas nettoyées et doivent être mises hors service manuellement. Pour savoir comment nettoyer manuellement ces instances, veuillez consulter [Nettoyage des instances Packer \(p. 1154\)](#).

Les utilisateurs de votre compte peuvent utiliser vos plateformes personnalisées en spécifiant un [ARN de plateforme \(p. 951\)](#) lors de la création de l'environnement. Ces ARN sont renvoyés par la commande eb platform create que vous avez utilisée pour créer la plateforme personnalisée.

Chaque fois que vous créez votre plateforme personnalisée, Elastic Beanstalk crée une nouvelle version de la plateforme. Les utilisateurs peuvent spécifier une plateforme par son nom pour obtenir uniquement la dernière version de la plateforme, ou inclure un numéro de version pour obtenir une version spécifique.

Par exemple, pour déployer la dernière version de la plateforme personnalisée ayant l'ARN **MyCustomPlatformARN**, qui peut être la version 3.0, la ligne de commande de votre interface de ligne de commande EB se présente comme suit :

```
eb create -p MyCustomPlatformARN
```

Pour déployer la version 2.1, la ligne de commande de votre interface de ligne de commande EB se présente comme suit :

```
eb create -p MyCustomPlatformARN --version 2.1
```

Vous pouvez appliquer des balises à une version de plateforme personnalisée lorsque vous la créez, et modifier des balises de versions de plateforme personnalisée existantes. Pour plus d'informations, consultez [Balisage des versions de plateforme personnalisée \(p. 1157\)](#).

## Création d'une plateforme personnalisée

Pour créer une plateforme personnalisée, la racine de votre application doit inclure un fichier de définition de plateforme `platform.yaml`, qui définit le type de générateur utilisé pour la création de la plateforme personnalisée. Le format de ce fichier est décrit dans [Format de fichier platform.yaml \(p. 1155\)](#). Vous pouvez créer une plateforme personnalisée entièrement nouvelle ou reposant sur l'un des [exemples de plateforme personnalisée \(p. 1148\)](#).

## Utilisation d'un exemple de plateforme personnalisée

Une alternative à la création de votre propre plateforme personnalisée consiste à utiliser l'un des exemples d'archive de définition de plateforme pour amorcer votre plateforme personnalisée. Une AMI source et une région constituent les seuls éléments que vous devez configurer dans les exemples avant de pouvoir les utiliser.

#### Note

N'utilisez pas d'exemple de plateforme personnalisée non modifié en production. L'objectif des exemples vise à illustrer certaines des fonctionnalités disponibles pour une plateforme personnalisée, mais qui n'ont pas été renforcées à des fins de production.

[NodePlatform\\_Ubuntu.zip](#)

Cette plateforme personnalisée est basée sur Ubuntu 16.04 et prend en charge Node.js 4.4.4. Nous allons utiliser cette plateforme personnalisée pour les exemples de cette section.

[NodePlatform\\_RHEL.zip](#)

Cette plateforme personnalisée est basée sur RHEL 7.2 et prend en charge Node.js 4.4.4.

[NodePlatform\\_AmazonLinux.zip](#)

Cette plateforme personnalisée est basée sur Amazon Linux 2016.09.1 et prend en charge Node.js 4.4.4.

[TomcatPlatform\\_Ubuntu.zip](#)

Cette plateforme personnalisée est basée sur Ubuntu 16.04 et prend en charge Tomcat 7/Java 8.

[CustomPlatform\\_NodeSampleApp.zip](#)

Exemple Node.js qui utilise express et ejs pour afficher une page web statique.

[CustomPlatform\\_TomcatSampleApp.zip](#)

Exemple Tomcat qui affiche une page web statique lors de son déploiement.

Téléchargez l'exemple d'archive de définition de plateforme : [NodePlatform\\_Ubuntu.zip](#). Ce fichier contient un fichier de définition de plateforme, un modèle Packer, des scripts exécutés par Packer lors de la création de l'image, et des scripts et des fichiers de configuration que Packer copie sur l'instance de générateur lors de la création de la plateforme.

[Example NodePlatform\\_Ubuntu.zip](#)

```
|-- builder                         Contains files used by Packer to create the custom platform
|-- custom_platform.json            Packer template
|-- platform.yaml                  Platform definition file
|-- ReadMe.txt                      Briefly describes the sample
```

Le fichier de définition de plateforme, `platform.yaml`, indique à Elastic Beanstalk le nom du modèle Packer, `custom_platform.json`.

```
version: "1.0"

provisioner:
  type: packer
  template: custom_platform.json
  flavor: ubuntu1604
```

Le modèle Packer indique à Packer comment créer les AMI pour la plateforme, en utilisant une [AMI Ubuntu](#) comme AMI de base pour l'image de plateforme pour les types d'instances HVM. La section `provisioners` demande à Packer de copier tous les fichiers dans le dossier `builder` au sein de l'archive vers l'instance et d'exécuter le script `builder.sh` sur l'instance. Lorsque les scripts sont terminés, Packer crée une image à partir de l'instance modifiée.

Elastic Beanstalk crée trois variables d'environnement qui peuvent être utilisées pour baliser des AMI dans Packer :

`AWS_EB_PLATFORM_ARN`

L'ARN de la plateforme personnalisée.

#### AWS\_EB\_PLATFORM\_NAME

Le nom de la plateforme personnalisée.

#### AWS\_EB\_PLATFORM\_VERSION

La version de la plateforme personnalisée.

L'exemple de fichier `custom_platform.json` utilise ces variables pour définir les valeurs suivantes qu'il utilise dans les scripts :

- `platform_name`, qui est définie par `platform.yaml`
- `platform_version`, qui est définie par `platform.yaml`
- `platform_arn`, qui est définie par le script de génération principal, `builder.sh`, qui s'affiche à la fin de l'exemple de fichier `custom_platform.json`.

Le fichier `custom_platform.json` contient deux propriétés pour lesquelles vous devez préciser une valeur : `source_ami` et `region`. Pour de plus amples informations sur le choix des valeurs appropriées d'AMI et de région, veuillez consulter [Updating Packer template](#) dans le référentiel GitHub `eb-custom-platforms-samples`.

#### Example `custom_platform.json`

```
{  
  "variables": {  
    "platform_name": "{{env `AWS_EB_PLATFORM_NAME`}}",  
    "platform_version": "{{env `AWS_EB_PLATFORM_VERSION`}}",  
    "platform_arn": "{{env `AWS_EB_PLATFORM_ARN`}}"  
  },  
  "builders": [  
    {  
      ...  
      "region": "",  
      "source_ami": "",  
      ...  
    }  
  ],  
  "provisioners": [  
    {...},  
    {  
      "type": "shell",  
      "execute_command": "chmod +x {{ .Path }}; {{ .Vars }} sudo {{ .Path }}",  
      "scripts": [  
        "builder/builder.sh"  
      ]  
    }  
  ]  
}
```

Les scripts et les autres fichiers que vous incluez dans votre archive de définition de plateforme varient considérablement selon les modifications que vous souhaitez apporter à l'instance. L'exemple de plateforme inclut les scripts suivants :

- `00-sync-apt.sh` – Mis en commentaire : `apt -y update`. Nous avons mis la commande en commentaire parce qu'elle demande à l'utilisateur une entrée, ce qui rompt la mise à jour automatique du package. Cela peut être un problème Ubuntu. Cependant, l'exécution d'`apt -y update` est toujours recommandée comme bonne pratique. Pour cette raison, nous avons laissé la commande dans l'exemple de script à titre de référence.
- `01-install-nginx.sh` – Installe nginx.

- `02-setup-platform.sh` – Installe `wget`, `tree` et `git`. Copie les hooks et les [configurations de journalisation \(p. 884\)](#) sur l'instance, et crée les répertoires suivants :
  - `/etc/SampleNodePlatform` – Emplacement de chargement du fichier de configuration de conteneur pendant le déploiement.
  - `/opt/elasticbeanstalk/deploy/appsource/` – Emplacement où le script `00-unzip.sh` charge le code source de l'application pendant le déploiement (pour de plus amples informations sur ce script, veuillez consulter la section [Outils de script de plateforme \(p. 43\)](#)).
  - `/var/app/staging/` – Emplacement de traitement du code source de l'application pendant le déploiement.
  - `/var/app/current/` – Emplacement d'exécution du code source de l'application après le traitement.
  - `/var/log/nginx/healthd/` – Emplacement où l'[agent amélioré de vérification de l'état \(p. 840\)](#) écrit les journaux.
  - `/var/nodejs` – Emplacement de chargement des fichiers Node.js pendant le déploiement.

Utilisez l'interface de ligne de commande EB pour créer votre première plateforme personnalisée avec l'exemple d'archive de définition de plateforme.

Pour créer une plateforme personnalisée

1. [Installez l'interface de ligne de commande EB \(p. 1028\)](#).
2. Créez un répertoire dans lequel l'exemple de plateforme personnalisée sera extrait.

```
~$ mkdir ~/custom-platform
```

3. Extrayez `NodePlatform_Ubuntu.zip` dans le répertoire, puis modifiez le répertoire extrait.

```
~$ cd ~/custom-platform
~/custom-platform$ unzip ~/NodePlatform_Ubuntu.zip
~/custom-platform$ cd NodePlatform_Ubuntu
```

4. Modifiez le fichier `custom_platform.json` et fournissez les valeurs des propriétés `source_ami` et `region`. Pour de plus amples informations, veuillez consulter [Updating Packer template](#).
5. Exécutez [eb platform init \(p. 1107\)](#) et suivez les instructions pour initialiser un référentiel de plateforme.

Vous pouvez raccourcir `eb platform` en `ebp`.

#### Note

Windows PowerShell utilise `ebp` comme alias de commande. Si vous exécutez l'interface de ligne de commande EB dans Windows PowerShell, utilisez la forme longue de cette commande : `eb platform`.

```
~/custom-platform$ ebp platform init
```

Cette commande crée également le répertoire `.elasticbeanstalk` dans le répertoire actuel et ajoute le fichier de configuration `config.yml` au répertoire. Ne modifiez ou ne supprimez pas ce fichier, car Elastic Beanstalk l'utilise pour créer la plateforme personnalisée.

Par défaut, `eb platform init` utilise le nom du dossier actuel comme nom de plateforme personnalisée, `custom-platform` dans cet exemple.

6. Exécutez la commande [eb platform create \(p. 1105\)](#) pour lancer un environnement Packer et obtenir l'ARN de la plateforme personnalisée. Vous aurez besoin de cette valeur plus tard lorsque vous créerez un environnement en utilisant la plateforme personnalisée.

```
~/custom-platform$ ebp platform create
```

...

Par défaut, Elastic Beanstalk crée le profil d'instance `aws-elasticbeanstalk-custom-platform-ec2-role` pour les plateformes personnalisées. Si vous préférez utiliser un profil d'instance existant, ajoutez l'option `-i` `INSTANCE_PROFILE` à la commande `eb platform create` (p. 1105).

Note

Packer ne parvient pas à créer de plateforme personnalisée si vous utilisez le profil d'instance Elastic Beanstalk par défaut, `aws-elasticbeanstalk-ec2-role`.

L'interface de ligne de commande EB affiche l'événement généré par l'environnement Packer jusqu'à ce que la génération soit terminée. Vous pouvez quitter la vue des événements en appuyant sur Ctrl +C.

7. Vous pouvez consulter les erreurs d'utilisation de la commande `eb platform logs` (p. 1108) dans les journaux.

```
~/custom-platform$ eb platform logs  
...
```

8. Vous pouvez vérifier le processus ultérieurement avec `eb platform events` (p. 1106).

```
~/custom-platform$ eb platform events  
...
```

9. Vérifiez l'état de votre plateforme avec `eb platform status` (p. 1108).

```
~/custom-platform$ eb platform status  
...
```

Lorsque l'opération est terminée, vous disposez d'une plateforme que vous pouvez utiliser pour lancer un environnement Elastic Beanstalk.

Vous pouvez utiliser la plateforme personnalisée lors de la création d'un nouvel environnement à partir de la console. Voir [Assistant de création d'un environnement](#) (p. 442).

Pour lancer un environnement sur votre plateforme personnalisée

1. Créez un nouveau répertoire pour votre application.

```
~$ mkdir custom-platform-app  
~$ cd ~/custom-platform-app
```

2. Initialisez un référentiel d'application.

```
~/custom-platform-app$ eb init  
...
```

3. Téléchargez l'exemple d'application [NodeSampleApp.zip](#).

4. Procédez à l'extraction de l'exemple d'application.

```
~/custom-platform-app$ unzip ~/NodeSampleApp.zip
```

5. Pour lancer un environnement exécutant votre plateforme personnalisée, exécutez la commande `eb create -p CUSTOM-PLATFORM-ARN`, où `CUSTOM-PLATFORM-ARN` est l'ARN renvoyé par une commande `eb platform create`.

```
~/custom-platform-app$ eb create -p CUSTOM-PLATFORM-ARN  
...
```

## Contenu de l'archive de définition de plateforme

Une archive de définition de plateforme est l'équivalent plateforme d'un [bundle de fichiers source d'application \(p. 415\)](#). L'archive de définition de plateforme est un fichier ZIP qui contient un fichier de définition de plateforme, un modèle Packer, ainsi que les scripts et les fichiers utilisés par le modèle Packer pour créer votre plateforme.

### Note

Lorsque vous utilisez l'interface de ligne de commande EB pour créer une plateforme personnalisée, une archive de définition de plateforme est créée à partir des fichiers et des dossiers de votre référentiel de plateforme. Vous n'avez donc pas besoin de créer l'archive manuellement.

Le fichier de définition de plateforme est un fichier au format YAML qui doit être nommé `platform.yaml` et se trouver à la racine de votre archive de définition de plateforme. Pour obtenir la liste des clés obligatoires et facultatives prises en charge dans un fichier de définition de plateforme, veuillez consulter [Création d'une plateforme personnalisée \(p. 1148\)](#).

Vous n'avez pas besoin d'attribuer un nom spécifique au modèle Packer, mais le nom du fichier doit correspondre au modèle de fournisseur spécifié dans le fichier de définition de plateforme. Consultez la [documentation Packer](#) officielle pour plus d'informations sur la création de modèles Packer.

Les autres fichiers que contient votre archive de définition de plateforme sont des scripts et des fichiers utilisés par le modèle pour personnaliser une instance avant de créer une AMI.

## Hooks de plateforme personnalisée

Elastic Beanstalk utilise une structure de répertoires normalisée pour les hooks sur les plateformes personnalisées. Il s'agit de scripts exécutés au cours d'événements du cycle de vie et en réponse à des opérations de gestion : lorsque des instances dans votre environnement sont lancées, ou lorsqu'un utilisateur initie un déploiement ou utilise la fonctionnalité de redémarrage de serveur d'application.

Placez les scripts que vous souhaitez voir déclenchés par les hooks dans l'un des sous-dossiers du dossier `/opt/elasticbeanstalk/hooks/`.

### Warning

L'utilisation de hooks de plateforme personnalisée sur des plateformes gérées n'est pas prise en charge. Les hooks de plateforme personnalisée sont conçus pour les plateformes personnalisées. Sur les plateformes Elastic Beanstalk gérées, ils peuvent fonctionner différemment ou présenter quelques problèmes, et le comportement peut varier d'une plateforme à l'autre. Sur les plateformes de l'AMI Amazon Linux (anciennement Amazon Linux 2), ils peuvent toujours s'avérer utiles dans certains cas ; utilisez-les avec prudence.

Les hooks de plateforme personnalisés sont une fonctionnalité héritée qui existe sur les plateformes de l'AMI Amazon Linux. Sur les plateformes Amazon Linux 2, les hooks de plateforme personnalisés qui figurent dans le dossier `/opt/elasticbeanstalk/hooks/` sont complètement obsolètes. Elastic Beanstalk ne les lit ou ne les exécute pas. Les plateformes Amazon Linux 2 prennent en charge un nouveau type de hook de plateforme, spécialement conçu pour étendre les plateformes Elastic Beanstalk gérées. Vous pouvez ajouter des scripts et des programmes personnalisés directement dans un répertoire hooks de votre bundle de fichiers source d'application. Elastic Beanstalk les exécute au cours des différentes étapes de

provisionnement d'instance. Pour plus d'informations, développez la section Platform Hooks dans [the section called “Extension des plateformes Linux” \(p. 34\)](#).

Les hooks sont organisés dans les dossiers suivants :

- `appdeploy` — Scripts exécutés lors d'un déploiement d'application. Elastic Beanstalk effectue un déploiement d'application lorsque de nouvelles instances sont lancées et quand un client initie un déploiement de nouvelle version.
- `configdeploy` — Scripts exécutés lorsqu'un client effectue une mise à jour de configuration qui affecte la configuration logicielle sur l'instance, par exemple, en définissant des propriétés d'environnement ou en activant la rotation des journaux sur Amazon S3.
- `restartappserver` — Scripts exécutés lorsqu'un client effectue une opération de redémarrage de serveur d'application.
- `preinit` — Scripts exécutés lors de l'amorçage d'une instance.
- `postinit` — Scripts exécutés après l'amorçage d'une instance.

Les dossiers `appdeploy`, `configdeploy` et `restartappserver` contiennent les sous-dossiers `pre`, `enact` et `post`. Lors de chaque phase d'une opération, tous les scripts du dossier `pre` sont exécutés par ordre alphabétique, puis viennent ceux du dossier `enact`, puis ceux du dossier `post`.

Lorsqu'une instance est démarrée, Elastic Beanstalk exécute `preinit`, `appdeploy` et `postinit`. Sur les déploiements suivants sur des instances en cours d'exécution, Elastic Beanstalk exécute les hooks `appdeploy`. Les hooks `configdeploy` sont exécutés lorsqu'un utilisateur met à jour des paramètres de configuration de logiciel d'instance. Les hooks `restartappserver` sont exécutés uniquement lorsque l'utilisateur lance un redémarrage de serveur d'application.

Lorsque vos scripts détectent des erreurs, ils peuvent quitter avec un état différent de zéro et écrire sur `stderr` pour faire échouer l'opération. Le message que vous écrivez dans `stderr` apparaîtra dans l'événement qui est généré lorsque l'opération échoue. Elastic Beanstalk capture également ces informations dans le fichier journal `/var/log/eb-activity.log`. Si vous ne voulez pas que l'opération échoue, renvoyez 0 (zéro). Les messages que vous écrivez dans `stderr` ou `stdout` apparaissent dans les [journaux de déploiement \(p. 884\)](#), mais pas dans le flux d'événements, sauf si l'opération échoue.

## Nettoyage des instances Packer

Dans certaines conditions, par exemple la suppression du processus du générateur Packer avant sa fin, les instances lancées par Packer ne sont pas nettoyées. Ces instances n'appartiennent pas à l'environnement Elastic Beanstalk et peuvent être affichées et résiliées uniquement à l'aide du service Amazon EC2.

Pour nettoyer manuellement ces instances

1. Ouvrez la [console Amazon EC2](#).
2. Vérifiez que vous êtes dans la même région AWS que celle dans laquelle vous avez créé l'instance avec Packer.
3. Sous Ressources, sélectionnez **N** Instances en cours d'exécution, où **N** indique le nombre d'instances en cours d'exécution.
4. Dans la zone de texte de la requête.
5. Sélectionnez la balise Name.
6. Saisissez packer.

La requête doit se présenter comme suit : tag:Name: packer

7. Sélectionnez les instances qui correspondent à la requête.
8. Si État de l'instance est en cours d'exécution, sélectionnez Actions, État de l'instance, Arrêter, puis Actions, État de l'instance, Résilier.

## Format de fichier platform.yaml

Le fichier `platform.yaml` a le format suivant :

```
version: "version-number"

provisioner:
    type: provisioner-type
    template: provisioner-template
    flavor: provisioner-flavor

metadata:
    maintainer: metadata-maintainer
    description: metadata-description
    operating_system_name: metadata-operating_system_name
    operating_system_version: metadata-operating_system_version
    programming_language_name: metadata-programming_language_name
    programming_language_version: metadata-programming_language_version
    framework_name: metadata-framework_name
    framework_version: metadata-framework_version

option_definitions:
    - namespace: option-def-namespace
        option_name: option-def-option_name
        description: option-def-description
        default_value: option-def-default_value

option_settings:
    - namespace: "option-setting-namespace"
        option_name: "option-setting-option_name"
        value: "option-setting-value"
```

Remplacez les espaces réservés par les valeurs suivantes :

`version-number`

Obligatoire. La version de la définition YAML. Doit indiquer **1.0**.

`provisioner-type`

Obligatoire. Le type de générateur utilisé pour créer la plateforme personnalisée. Doit indiquer **packer**.

`provisioner-template`

Obligatoire. Le fichier JSON contenant les paramètres correspondant à `provisioner-type`.

`provisioner-flavor`

Facultatif. Le système d'exploitation de base utilisé pour l'AMI. L'un des éléments suivants :

amazon (valeur par défaut)

Amazon Linux. Si la version n'est pas spécifiée, il s'agit de la dernière version d'Amazon Linux disponible au moment de la création de la plateforme.

Amazon Linux 2 n'est pas une version de système d'exploitation prise en charge.

ubuntu1604

Ubuntu 16.04 LTS

rhel7

RHEL 7

rhel6

RHEL 6

*metadata-maintainer*

Facultatif. Informations de contact pour la personne qui possède la plateforme (100 caractères).

*metadata-description*

Facultatif. Description de la plateforme (2 000 caractères).

*metadata-operating\_system\_name*

Facultatif. Nom du système d'exploitation de la plateforme (50 caractères). Cette valeur est disponible lors du filtrage de la sortie pour l'API [ListPlatformVersions](#).

*metadata-operating\_system\_version*

Facultatif. Version du système d'exploitation de la plateforme (20 caractères).

*metadata-programming\_language\_name*

Facultatif. Langage de programmation pris en charge par la plateforme (50 caractères).

*metadata-programming\_language\_version*

Facultatif. Langue de la version de la plateforme (20 caractères).

*metadata-framework\_name*

Facultatif. Nom de l'infrastructure web utilisée par la plateforme (50 caractères).

*metadata-framework\_version*

Facultatif. Version de l'infrastructure web de la plateforme (20 caractères).

*option-def-namespace*

Facultatif. Un espace de noms sous `aws:elasticbeanstalk:container:custom` (100 caractères)

*option-def-option\_name*

Facultatif. Nom de l'option (100 caractères). Vous pouvez définir jusqu'à 50 options de configuration personnalisées que la plateforme fournit aux utilisateurs.

*option-def-description*

Facultatif. Description de l'option (1 024 caractères).

*option-def-default\_value*

Facultatif. Valeur par défaut utilisée lorsque l'utilisateur n'en spécifie pas.

L'exemple suivant crée l'option **NPM\_START**.

```
options_definitions:
  - namespace: "aws:elasticbeanstalk:container:custom:application"
    option_name: "NPM_START"
    description: "Default application startup command"
    default_value: "node application.js"
```

*option-setting-namespace*

Facultatif. Espace de noms de l'option.

*option-setting-option\_name*

Facultatif. Nom de l'option. Vous pouvez spécifier jusqu'à 50 [options fournies par Elastic Beanstalk](#) (p. 679).

#### **option-setting-value**

Facultatif. Valeur utilisée lorsque l'utilisateur n'en spécifie pas.

L'exemple suivant crée l'option **TEST**.

```
option_settings:  
  - namespace: "aws:elasticbeanstalk:application:environment"  
    option_name: "TEST"  
    value: "This is a test"
```

## Balisage des versions de plateforme personnalisée

Vous pouvez appliquer des balises aux versions de votre plateforme AWS Elastic Beanstalk personnalisée. Les balises sont des paires clé-valeur associées aux ressources AWS. Pour de plus amples informations sur le balisage des ressources Elastic Beanstalk, les cas d'utilisation, les contraintes de clé et de valeur de balise, et les types de ressources pris en charge, veuillez consulter [Balisage des ressources d'application Elastic Beanstalk \(p. 423\)](#).

Vous pouvez spécifier des balises lorsque vous créez une version de plateforme personnalisée. Dans une version de plateforme personnalisée, vous pouvez ajouter ou supprimer des balises, ainsi que mettre à jour les valeurs des balises existantes. Vous pouvez ajouter jusqu'à 50 balises à chaque version de plateforme personnalisée.

### Ajout de balises lors de la création de versions de plateforme personnalisée

Si vous utilisez l'interface de ligne de commande EB pour créer votre version de plateforme personnalisée, utilisez l'option **--tags** avec [eb platform create \(p. 1105\)](#) pour ajouter des balises.

```
~/workspace/my-app$ eb platform create --tags mytag1=value1,mytag2=value2
```

Avec l'AWS CLI ou les autres clients basés sur l'API, ajoutez des balises en utilisant le paramètre **--tags** sur la commande [create-platform-version](#).

```
$ aws elasticbeanstalk create-platform-version \  
  --tags Key=mytag1,Value=value1 Key=mytag2,Value=value2 \  
  --platform-name my-platform --platform-version 1.0.0 --platform-definition-bundle  
  S3Bucket=DOC-EXAMPLE-BUCKET,S3Key=sample.zip
```

### Gestion des balises d'une version de plateforme personnalisée existante

Vous pouvez ajouter, mettre à jour et supprimer des balises dans une version existante de la plateforme personnalisée Elastic Beanstalk.

Si vous utilisez l'interface de ligne de commande EB pour mettre à jour votre version de plateforme personnalisée, utilisez [eb tags \(p. 1118\)](#) pour ajouter, mettre à jour, supprimer ou répertorier des balises.

Par exemple, la commande suivante répertorie les balises dans une version de plateforme personnalisée.

```
~/workspace/my-app$ eb tags --list --resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:platform/my-platform/1.0.0"
```

La commande suivante met à jour la balise mytag1 et supprime la balise mytag2.

```
~/workspace/my-app$ eb tags --update mytag1=newvalue --delete mytag2 \
--resource "arn:aws:elasticbeanstalk:us-east-2:my-account-id:platform/my-
platform/1.0.0"
```

Pour obtenir une liste complète des options et d'autres exemples, consultez [eb tags \(p. 1118\)](#).

Avec l'AWS CLI ou d'autres clients basés sur l'API, utilisez la commande [list-tags-for-resource](#) pour afficher les balises d'une version de plateforme personnalisée.

```
$ aws elasticbeanstalk list-tags-for-resource --resource-arn "arn:aws:elasticbeanstalk:us-
east-2:my-account-id:platform/my-platform/1.0.0"
```

Utilisez la commande [update-tags-for-resource](#) pour ajouter, mettre à jour ou supprimer des balises dans une version de plateforme personnalisée.

```
$ aws elasticbeanstalk update-tags-for-resource \
--tags-to-add Key=mytag1,Value=newvalue --tags-to-remove mytag2 \
--resource-arn "arn:aws:elasticbeanstalk:us-east-2:my-account-id:platform/my-
platform/1.0.0"
```

Spécifiez les balises à ajouter et les balises à mettre à jour dans le paramètre --tags-to-add de update-tags-for-resource. Une balise inexistante est ajoutée et la valeur d'une balise existante est mise à jour.

#### Note

Pour utiliser certaines commandes de l'interface de ligne de commande EB et de l'AWS CLI avec une version de la plateforme personnalisée Elastic Beanstalk, vous avez besoin de l'ARN de la version de la plateforme personnalisée. Vous pouvez extraire l'ARN à l'aide de la commande suivante.

```
$ aws elasticbeanstalk list-platform-versions
```

Utilisez l'option --filters pour filtrer la sortie vers le nom de votre plateforme personnalisée.