Une AWS Instance Scheduler est un outil fourni par Amazon Web Services (AWS) qui permet de planifier et d'automatiser les arrêts et les démarrages des EC2 instances d'AWS. On peut configurer les horaires spécifiques pour l'arrêt et le démarrage des instances EC2 en fonction de nos besoins.

Comme Avantages de AWS instances Scheduler nous avons la réduction des couts des l’infrastructure, optimisation des ressources, gestion de la conformité mais aussi un gain en temps dans la mesure où c’est un processus automatisé.

Speach de transition entre les diapo: Instances scheduler n’est pas un service comme les autres dans la mesure ou sur la console AWS si on entre AWS instance scheduler on ne vera pas de service, car c’est une solution proposée par un ensemble de service.

EC2 Instances

AWS instances scheduler fonctionne avec les instances ec2. On peut planifier les heures de demarrage et d’arret des instances EC2 en utilisant instances scheduler.

AWS lambda

AWS Lambda est un service de (AWS) qui permet d'exécuter du code sans avoir à gérer les serveurs ou les infrastructures sous-jacentes. Elle s'exécute automatiquement en réponse à des événements déclenchés par d'autres services AWS. Elle est conçue pour faciliter le développement et le déploiement d'applications basées sur le cloud, en offrant une gestion automatisée des ressources.

AWS Lambda nous offre beaucoup d’avantages : Large prise en charge de langages de programmation, une Scalabilité automatique : AWS Lambda peut évoluer automatiquement pour répondre à la demande de traitement des fonctions Lambda, sans nécessiter de configuration manuelle de l'infrastructure. La facturation de AWs lambda est basee sur l’execution du code. La facturation depend de la quantite de mémoire allouee a la fonction, de la duree d’excution du code et le nombre d’invocation. Autres avantages integration avec d’autres services AWS.

Ces fonctionnalités rendent la fonction Lambda d'AWS très utile pour toute personne qui cherchent à créer des applications évolutives et flexibles, sans avoir à se soucier de la gestion de l'infrastructure sous-jacente.

CloudWatch

AWS CloudWatch est un service de monitoring. Il permet de collecter, stocker et surveiller les métriques, les journaux et les événements générés par les ressources AWS et les applications déployées sur AWS.

1. CloudWatch metriques : CloudWatch permet de collecter et de stocker des métriques de performance telles que l'utilisation du processeur, l'utilisation de la mémoire, la consommation de bande passante, etc.
2. CloudWatch alarmes : CloudWatch permet de configurer des alarmes pour surveiller les métriques de performance des ressources AWS et des applications déployées sur AWS. Les alarmes peuvent être configurées pour déclencher des notifications (par exemple, un e-mail ou un SMS) ou des actions automatiques (par exemple, le redémarrage d'une instance EC2) en cas de dépassement des seuils configurés.
3. CloudWatch logs : CloudWatch permet de collecter, stocker et analyser les logs générés par les instances EC2, les conteneurs, les services Lambda, et d'autres ressources AWS. Les logs peuvent être utilisés pour détecter les erreurs, les problèmes de performance et les tendances.
4. CloudWatch Events : CloudWatch Events est un service de CloudWatch qui permet de surveiller les événements générés par les ressources AWS et de déclencher des actions en réponse à ces événements.

En resume, les fonctionnalités de CloudWatch permettent de surveiller et d'analyser en temps réel les ressources AWS et les applications déployées sur AWS, de détecter les erreurs et les problèmes de performance, et de prendre des mesures automatiques

1. DynamoDB : DynamoDB est une base de données NoSQL entièrement gérée d'AWS. Il peut être utilisé pour stocker des informations sur les horaires planifiés pour le démarrage et l'arrêt des instances EC2. Les AWS Lambda fonction peuvent être configurées pour lire ces informations à partir de DynamoDB et executer le script de la lambda.

Pour permettre à chaque service de communiquer avec les autres, On doit également configurer les autorisations IAM appropriées. Par exemple, On doit donner à la fonction Lambda les autorisations nécessaires pour accéder à la table DynamoDB et à CloudWatch event.

1. Réduction des coûts : En planifiant l'arrêt des instances lorsqu'elles ne sont pas utilisées, les utilisateurs peuvent réduire les coûts d'infrastructure en évitant de payer pour des instances inutiles. Cela peut entraîner des économies significatives sur les coûts de fonctionnement d'une infrastructure de cloud computing.
2. Amélioration de la disponibilité : En planifiant la mise en marche des instances au moment opportun, les utilisateurs peuvent s'assurer que les instances nécessaires sont disponibles lorsque les utilisateurs en ont besoin. Cela peut aider à améliorer la disponibilité des applications et des services qui s'exécutent sur ces instances.
3. Automatisation : AWS Instance Scheduler permet aux utilisateurs de programmer les arrêts et les démarreurs des instances EC2 en fonction de règles prédéfinies, sans intervention manuelle. Cela peut aider à automatiser les tâches de gestion des instances, ce qui permet aux équipes informatiques de se concentrer sur des tâches plus importantes.
4. Flexibilité : AWS Instance Scheduler est flexible et peut être configuré pour répondre aux besoins spécifiques de chaque organisation. Les utilisateurs peuvent créer des règles personnalisées pour leurs instances EC2, en fonction de leur environnement et de leurs besoins.

En résumé, AWS Instance Scheduler est un outil important pour les utilisateurs d'AWS qui souhaitent optimiser les coûts et améliorer la disponibilité de leurs instances EC2, tout en automatisant la gestion de ces instances.

L'intégration de AWS Instance Scheduler, DynamoDB, AWS CloudWatch Events, IAM et Lambda permet de planifier des démarrages et des arrêts automatiques d'instances EC2 selon des horaires préconfigurés, ce qui peut aider à réduire les coûts d'infrastructure.

Voici les étapes pour configurer cette intégration:

1. Créez une table DynamoDB pour stocker les plannings de démarrage et d'arrêt des instances EC2. Cette table doit comporter au moins les champs "InstanceId", "InstanceName", "ScheduleStart" et "ScheduleStop".
2. Créez une fonction Lambda AWS pour traiter les événements CloudWatch et mettre à jour la table DynamoDB. Cette fonction doit avoir les autorisations IAM nécessaires pour accéder à la table DynamoDB.
3. Créez un événement CloudWatch qui déclenchera la fonction Lambda selon un calendrier défini. Par exemple, vous pouvez configurer l'événement pour se déclencher toutes les nuits à minuit.
4. Dans la fonction Lambda, vous pouvez utiliser le SDK AWS pour récupérer la liste des instances EC2 à arrêter ou à démarrer, puis mettre à jour la table DynamoDB avec les horaires de démarrage et d'arrêt correspondants.
5. Configurer AWS Instance Scheduler en utilisant la table DynamoDB comme source de données. Cela vous permettra de planifier les démarrages et les arrêts d'instances EC2 en fonction des horaires définis dans la table.

Pour permettre à chaque service de communiquer avec les autres, vous devez également configurer les autorisations IAM appropriées. Par exemple, vous devez donner à la fonction Lambda les autorisations nécessaires pour accéder à la table DynamoDB et à l'événement CloudWatch, et vous devez donner à Instance Scheduler les autorisations nécessaires pour accéder à la table DynamoDB.

En mettant en place cette intégration, vous pouvez réduire les coûts d'infrastructure en arrêtant les instances EC2 qui ne sont pas utilisées en dehors des heures de travail ou pendant les périodes de faible activité. Cela peut également aider à optimiser l'utilisation des ressources AWS et à améliorer l'efficacité opérationnelle.

"0/5 \* \* \* ? \*" est une expression de calendrier Cron utilisée pour planifier une tâche qui doit s'exécuter toutes les 5 minutes.

Voici une explication détaillée de chaque partie de l'expression:

* Le premier champ "0/5" spécifie le nombre de minutes après l'heure où la tâche doit être exécutée. Dans cet exemple, la tâche doit être exécutée toutes les 5 minutes, donc "0/5" signifie que la tâche doit être exécutée lorsque la minuterie est à 0 seconde, 5e seconde, 10e seconde, etc.
* Le deuxième champ "\*" spécifie que la tâche doit être exécutée toutes les heures, sans distinction de l'heure spécifique.
* Le troisième champ "\*" spécifie que la tâche doit être exécutée tous les jours, sans distinction du jour spécifique.
* Le quatrième champ "\*" spécifie que la tâche doit être exécutée tous les mois, sans distinction du mois spécifique.
* Le cinquième champ "?" est utilisé pour spécifier un jour de la semaine ou pour ignorer la spécification du jour de la semaine. Dans cet exemple, la tâche doit être exécutée tous les jours de la semaine, donc le champ "?" est utilisé.
* Le sixième champ "\*" spécifie que la tâche doit être exécutée tous les jours de la semaine, sans distinction du jour spécifique.

En résumé, "0/5 \* \* \* ? \*" signifie que la tâche doit être exécutée toutes les 5 minutes, tous les jours de la semaine, à n'importe quelle heure, n'importe quel jour du mois et n'importe quel mois.

Les points d'interrogation ("?") sont utilisés dans les expressions cron pour représenter un caractère de "non-spécification" dans les champs du jour de la semaine et du jour du mois.

Dans l'expression cron "0/5 \* \* \* ? \*", les deux champs du jour de la semaine et du jour du mois sont remplacés par "?", ce qui signifie que ces champs ne sont pas spécifiés et la tâche sera exécutée tous les 5 minutes indépendamment du jour de la semaine ou du mois.

En résumé, les points d'interrogation sont utilisés pour indiquer que l'on ne spécifie pas de valeur pour un champ particulier, et qu'on veut donc exécuter la tâche à chaque occurrence possible pour les autres champs spécifiés.

AWS Lambda est un service de calcul sans serveur d'Amazon Web Services (AWS) qui permet d'exécuter du code sans avoir à gérer les serveurs ou les infrastructures sous-jacentes. Les fonctions Lambda sont des fonctions AWS sans serveur qui s'exécutent automatiquement en réponse à des événements déclenchés par d'autres services AWS ou des requêtes HTTP. AWS Lambda est un service d'informatique sans serveur proposé par Amazon Web Services (AWS) qui permet aux développeurs de créer des fonctions Lambda et de les déployer facilement dans le cloud AWS.

Pour créer une fonction Lambda dans AWS, vous devez d'abord créer un package de code qui contient votre fonction et toutes les dépendances nécessaires. Le package peut être téléchargé depuis votre ordinateur ou être stocké dans un référentiel Git.

Ensuite, vous pouvez créer une fonction Lambda en utilisant la console AWS ou l'API AWS. Lors de la création de la fonction, vous devez spécifier le package de code et la méthode de gestion des événements déclencheurs. Les événements peuvent être déclenchés par une variété de services AWS tels que S3, DynamoDB, API Gateway, CloudWatch, et plus encore.

Une fois que la fonction Lambda est créée, elle est automatiquement déployée et peut être exécutée en réponse à des événements déclencheurs. AWS Lambda gère automatiquement l'évolutivité et les ressources nécessaires pour exécuter les fonctions en parallèle, en fonction du nombre de demandes.

Le coût de la fonction Lambda dépend de la quantité de mémoire allouée à la fonction, de la durée d'exécution et du nombre d'invocations. Vous ne payez que pour le temps d'exécution réel de votre code et le nombre d'invocations, ce qui en fait une solution rentable pour les charges de

Le temps d'exécution d'une Lambda function dépend de plusieurs facteurs, tels que la taille du code de la fonction, la quantité de ressources allouées à la fonction, la complexité de la logique de la fonction, et la quantité de données traitées par la fonction.

Le temps d'exécution peut varier en fonction de ces facteurs, mais en général, les fonctions Lambda sont conçues pour être très rapides et répondre rapidement aux événements déclencheurs. En général, le temps d'exécution d'une Lambda function peut aller de quelques millisecondes à plusieurs secondes.

Il est important de noter que Lambda facture les temps d'exécution en tranches de 100 millisecondes, donc même une différence de quelques millisecondes peut affecter le coût de l'exécution de la fonction.

Un AWS CloudWatch Event est un service géré par Amazon Web Services (AWS) qui permet de surveiller les ressources AWS et de déclencher des actions en réponse à des événements.

Les événements sont générés par les ressources AWS, tels que les instances EC2, les groupes de sécurité, les files d'attente SQS, etc. CloudWatch Event permet de détecter ces événements et de les traiter en temps réel, en utilisant des règles pour filtrer et router les événements vers des destinations telles que Lambda, SNS, SQS, etc.

Les règles de CloudWatch Event peuvent être configurées pour détecter des événements spécifiques et déclencher des actions en réponse. Par exemple, vous pouvez créer une règle pour surveiller les événements relatifs à une instance EC2 spécifique, puis déclencher une Lambda function pour effectuer une action, telle que l'arrêt ou le redémarrage de l'instance.

CloudWatch Event peut être utilisé pour automatiser les tâches et les workflows, pour répondre rapidement aux événements et pour surveiller les ressources AWS en temps réel.

AWS CloudWatch est un service de surveillance et d'analyse de logs proposé par Amazon Web Services (AWS). Il permet de collecter, stocker et surveiller les métriques, les journaux et les événements générés par les ressources AWS et les applications déployées sur AWS.

CloudWatch peut collecter et stocker des métriques telles que l'utilisation du processeur, l'utilisation de la mémoire, la consommation de bande passante, etc. Ces métriques peuvent être utilisées pour surveiller la performance et la disponibilité des ressources AWS, et pour déclencher des alarmes en cas de dépassement des seuils configurés.

CloudWatch permet également de collecter et de stocker des logs générés par les instances EC2, les conteneurs, les services Lambda, et d'autres ressources AWS. Ces logs peuvent être analysés pour détecter les erreurs, les problèmes de performance et les tendances.

Enfin, CloudWatch Events est un service de CloudWatch qui permet de surveiller les événements générés par les ressources AWS et de déclencher des actions en réponse à ces événements.

En somme, AWS CloudWatch est un service central de surveillance, de gestion et d'analyse des ressources et des applications déployées sur AWS.

AWS CloudWatch offre un large éventail de fonctionnalités de surveillance et d'analyse des ressources AWS et des applications déployées sur AWS. Voici quelques-unes des principales fonctionnalités de CloudWatch :

1. Collecte et stockage de métriques : CloudWatch permet de collecter et de stocker des métriques de performance telles que l'utilisation du processeur, l'utilisation de la mémoire, la consommation de bande passante, etc.
2. Surveillance et alarmes : CloudWatch permet de configurer des alarmes pour surveiller les métriques de performance des ressources AWS et des applications déployées sur AWS. Les alarmes peuvent être configurées pour déclencher des notifications (par exemple, un e-mail ou un SMS) ou des actions automatiques (par exemple, le redémarrage d'une instance EC2) en cas de dépassement des seuils configurés.
3. Collecte et analyse de logs : CloudWatch permet de collecter, stocker et analyser les logs générés par les instances EC2, les conteneurs, les services Lambda, et d'autres ressources AWS. Les logs peuvent être utilisés pour détecter les erreurs, les problèmes de performance et les tendances.
4. CloudWatch Events : CloudWatch Events est un service de CloudWatch qui permet de surveiller les événements générés par les ressources AWS et de déclencher des actions en réponse à ces événements.

En somme, les fonctionnalités de CloudWatch permettent de surveiller et d'analyser en temps réel les ressources AWS et les applications déployées sur AWS, de détecter les erreurs et les problèmes de performance, et de prendre des mesures automatiques en cas de dépassement des seuils configurés.