Projet Chef d'oeuvre

Introduction

Les administrateurs système DevOps déploient, supervisent et automatisent les infrastructures et applications en cloud privé, public ou hybride. Ils vont également mettre en œuvre des solutions de supervision et de sécurité au travers d’indicateurs et de scans qui remonteront des alertes ou anomalies.

Dans le but d’obtenir un titre professionnel « Administrateur système DevOps », il y a trois activités type à maîtriser :

- Automatiser le déploiement d’une infrastructure dans le cloud :

Création et configuration de serveurs virtuels à l’aide de scripts, puis à l'aide d'un outil de gestion des configurations de type Ansible ou Terraform ; automatisation du déploiement de l’infrastructure ; mise en place de toutes les mesures de sécurité, préparation d'un environnement de test et de pré-production afin de tester les mises à jour avant leur mise en production ; configuration de l’offre cloud sur laquelle l’infrastructure sera déployée ; déploiement de l’infrastructure sur un cloud privé, public ou hybride à l’aide d’un outil de gestion de configuration de type Ansible ; vérification que tous les services sont accessibles par les utilisateurs finaux.

- Déployer une application en continu :

Récupération des codes de l’application à déployer, puis préparation d'un environnement de test pour y déployer l’application avant sa mise en production ; exécution des premiers tests et remonte des dysfonctionnements aux développeurs ; préparation des différents serveurs de données et le stockage associé ; création et gestion des containers destinés à recevoir l’application ; migration des données et déploiement de l’application dans l’environnement de pré-production ; échange avec l’équipe des développeurs pour corriger les dysfonctionnements découverts lors des différentes phases de tests ; déploiement de l’application et de ses mises à jour successives sur l’environnement de production à l’aide d’une plateforme de type Kubernetes.

- Superviser les services déployés :

Définition des indicateurs à surveiller et mise en place d'outils pour le faire, puis installation et configuration d'une solution de supervision ; correction du problème si anomalie.

Un projet doit être réalisé en entreprise ou en centre de formation validant l’acquisition des compétences nécessaires pour réaliser ces activités types. Dans le cas présent, en raison de la complexité associée à la mise en œuvre d'un projet complet et exhaustif couvrant les

compétences demandées en entreprise, et étant donné que l'entreprise ne ressent pas le besoin de développer une telle infrastructure (car elle dispose déjà d'infrastructures établies nécessitant principalement de la maintenance et de la supervision), le projet a été mené en centre de formation. Dans ce cas là, le cahier de charge est rédigé par le candidat.

Ce projet vise à démontrer l'acquisition des compétences et connaissances nécessaires requises pour assumer efficacement le rôle d'un administrateur système DevOps. Il s'agit d'une démonstration pratique des savoir-faire et techniques nécessaires pour gérer et optimiser les systèmes et processus informatiques dans un environnement DevOps : développement de scripts, utilisation de différentes plateformes, automatisation de processus, administration, sécurisation et supervision de systèmes.

I. Cahier des charges

1. Objectif du projet

L'objectif de ce projet est d'automatiser le déploiement de WordPress, tout en sécurisant et supervisant l'infrastructure et l'application. Afin de réaliser cet objectif, différentes étapes devront être mises en application et diverses compétences devront être utilisées dans ce but :

- Automatisation et Déploiement :
  - Utilisation de Docker sur une VM Azure pour déployer Jenkins.
  - Déploiement du cluster Azure Kubernetes Service (AKS) en utilisant Terraform.
  - Configuration de Jenkins pour automatiser le déploiement de WordPress avec autoscaling et de sa base de données MariaDB.
  - Configuration de l'Ingress Traefik pour exposer WordPress au trafic entrant.
- Sécurité et Supervision :
  - Mise en place de tests de sécurité avec WPScan et Sonar Cloud.
  - Mise en place du chiffrement TLS pour sécuriser la communication avec WordPress.
  - Mise en place de Prometheus et Grafana pour superviser l'infrastructure et les performances de WordPress.
  - Mise en place de Log Analytics pour collecter et analyser les logs provenant de l'infrastructure Kubernetes
- Développement de compétences professionnelles :
  - Automatiser la création de serveurs à l'aide de scripts
  - Automatiser le déploiement d'une infrastructure
  - Sécuriser l'infrastructure
  - Mettre l'infrastructure en production dans le cloud
  - Gérer le stockage des données
  - Gérer des containers
  - Automatiser la mise en production d'une application avec une plateforme
  - Définir et mettre en place des statistiques de services
  - Exploiter une solution de supervision

2. Exigences fonctionnelles

Déploiement de l'Infrastructure

Dans ce projet, Terraform sera utilisé pour orchestrer le déploiement et la gestion d'un cluster Azure Kubernetes Service (AKS), constituant la base de l'infrastructure. Terraform est un outil d'infrastructure en tant que code qui permet de définir de manière déclarative l'infrastructure souhaitée. Cela facilite le déploiement et la gestion de l'infrastructure de manière reproductible, garantissant ainsi une mise en place uniforme et fiable du cluster tout en offrant une certaine flexibilité. Cette approche garantit également une intégration aisée avec d'autres services cloud d'Azure, tels que les groupes de sécurité et les comptes de stockage, assurant une configuration optimale et sécurisée du cluster.

Le rôle de Terraform dans ce projet va au-delà de la simple création du cluster AKS : il englobe également la configuration de l'ensemble de l'infrastructure nécessaire - comme les aspects réseau ou stockage. Cette approche garantit non seulement un déploiement automatisé et reproductible mais aussi une mise à jour efficace et maîtrisée de l'ensemble de l'infrastructure. De plus, Terraform jouera un rôle clé dans

l'intégration fluide avec des services Azure complémentaires, comme Log Analytics pour le monitoring et Azure Storage pour la gestion des données.

Le cluster AKS, ainsi déployé et géré, servira de plateforme pour l'hébergement et l'exécution de WordPress et de sa base de données MariaDB. La gestion et l'orchestration des conteneurs au sein du cluster seront facilitées, assurant un fonctionnement optimal des applications dans un environnement Kubernetes géré. L'utilisation de Terraform pour définir et déployer ces composants clés permettra une gestion précise et une évolution aisée de l'infrastructure, en phase avec les exigences dynamiques du projet.

Automatisation des Processus

La partie cruciale de l'automatisation des processus est gérée par Jenkins. Jenkins est un outil d'intégration continue et de déploiement continu (CI/CD) largement utilisé. Il sera déployé à l'aide de Docker sur une machine virtuelle (VM) Azure, offrant ainsi un environnement contrôlé et isolé pour la gestion des pipelines d'automatisation. Il sera configuré pour orchestrer et automatiser l'ensemble du processus d'infrastructure.

Le pipeline sera donc conçu pour prendre en charge le déploiement automatisé sur le cluster AKS de WordPress et sa base de données MariaDB, le scaling, les outils de sécurité et de supervision et le TLS, en utilisant des fichiers Helm stockés sur GitHub. Cette configuration permet de simplifier considérablement le processus de déploiement, en réduisant les interventions manuelles et en accélérant le déploiement. L'intégration de Jenkins avec GitHub assure également un suivi efficace des modifications apportées au code et aux configurations en assurant ainsi une synchronisation continue et en favorisant ainsi un processus de développement et de déploiement transparent et cohérent. Jenkins contiendra également les secrets nécessaires au déploiement, tels que les identifiants d'accès et les clés API, et des plugins et outils seront mis en place comme nécessaire pour étendre les capacités de Jenkins, notamment pour l'intégration avec Kubernetes, Terraform, et les outils de sécurité.

Gestion des Applications

La gestion des applications, en particulier WordPress et sa base de données MariaDB, est réalisée via Helm, un gestionnaire de packages pour Kubernetes. Helm facilite le déploiement et la gestion de WordPress et de MariaDB sur le cluster AKS. Avec Helm, les configurations complexes de WordPress et de sa base de données sont simplifiées en packages réutilisables, appelés charts, qui décrivent toutes les ressources nécessaires et leurs interdépendances. Cette approche permet une mise en œuvre rapide et facile de WordPress, tout en offrant la flexibilité nécessaire pour personnaliser la configuration en fonction des besoins spécifiques du projet.

Tests de sécurité et Sécurité avancée

Les outils de sécurité, tels que WPScan et Sonar Cloud, sont utilisés pour identifier les vulnérabilités et les problèmes de sécurité dans les composants logiciels utilisés. WPScan est un scanner de sécurité spécifiquement conçu pour WordPress. Il permet de détecter les vulnérabilités connues dans les thèmes, les plugins et la version même de WordPress. En identifiant ces failles, WPScan aide à prévenir les attaques et à renforcer la sécurité du site. Sonar Cloud examine le code pour détecter les bugs, les vulnérabilités de sécurité et les mauvaises pratiques de codage. Son utilisation contribue à maintenir un niveau élevé de qualité de code, réduisant ainsi le risque de failles de sécurité et améliorant la maintenabilité du projet. Ces tests seront intégrés dans le processus de déploiement de Jenkins afin d'exécuter automatiquement et quotidiennement ces tests sur WordPress.

Pour renforcer la sécurité de l'application WordPress, des mesures de sécurité avancées seront également mises en place. Cela inclut la configuration du chiffrement TLS pour sécuriser la communication entre les utilisateurs et WordPress.

Surveillance et Monitoring

Pour assurer une surveillance et un monitoring efficaces de l'infrastructure et des applications, Prometheus et Grafana sont mis en place. Prometheus est un système de surveillance utilisé pour collecter des métriques de performance et d'état de l'infrastructure Kubernetes et des applications déployées. Ces métriques sont ensuite visualisées dans Grafana, offrant une vue d'ensemble claire et détaillée de la santé et des performances du système.

Cette combinaison d'outils permet non seulement un monitoring efficace et une capacité d'intervention rapide en cas de problèmes, afin d'optimiser les performances de l'application WordPress et de l'infrastructure sous-jacente, mais aussi de planifier des améliorations proactives de l'infrastructure. La capacité de Prometheus à collecter des métriques à grande échelle, combinée à la puissance de visualisation de Grafana, fait de cette solution un choix idéal pour la supervision en temps réel de systèmes complexes.

En complément, Azure Log Analytics sera utilisé pour fournir une analyse plus approfondie et des aperçus sur le comportement de l'infrastructure en collectant et analysant les logs provenant de l'infrastructure Kubernetes.

3. Exigences non fonctionnelles

Sécurité

La sécurité est une préoccupation majeure dans la gestion de tout système informatique, et encore plus dans un environnement cloud où les ressources sont exposées sur Internet. Dans ce projet, plusieurs mesures sont prises pour assurer la sécurité de l'infrastructure et des applications :

- TLS pour le Chiffrement des Communications : L'utilisation de TLS (Transport Layer Security) est essentielle pour sécuriser les communications entre les clients et le serveur WordPress. En mettant en œuvre le chiffrement TLS, toutes les données transmises, y compris les informations sensibles comme les identifiants de connexion, sont protégées contre les écoutes indiscrètes. Cela est particulièrement important pour les sites Web qui gèrent des données personnelles ou confidentielles.
- WPScan pour les Scans de Sécurité : WPScan offre une analyse approfondie de la sécurité de WordPress en détectant les vulnérabilités connues dans les thèmes, les plugins et la plateforme elle-même. En intégrant WPScan dans le processus d'intégration continue, le système bénéficie d'une surveillance proactive contre les failles de sécurité, permettant une intervention rapide pour corriger toute vulnérabilité identifiée.
- Sonar Cloud pour l'Analyse de Code : Sonar Cloud joue un rôle clé dans la garantie de la qualité du code en analysant les sources pour détecter les bugs, les vulnérabilités de sécurité et les antipatterns. L'utilisation de Sonar Cloud contribue à maintenir un code propre et sécurisé, ce qui est essentiel pour la fiabilité à long terme de l'infrastructure.
- Log Analytics pour le Suivi et l'Analyse : Azure Log Analytics est un outil puissant dans le cadre de la surveillance et de l'analyse de la sécurité. Il collecte et analyse les données de journalisation provenant de diverses sources. Cette approche centralisée de gestion des journaux permet de détecter rapidement les activités suspectes, les tentatives d'intrusion ou les anomalies de comportement, offrant ainsi une visibilité complète sur l'état de sécurité de l'infrastructure.

Performance et Disponibilité

La performance et la disponibilité sont critiques pour assurer une expérience utilisateur optimale et maintenir la confiance des utilisateurs :

- Configuration de l'Autoscaling pour WordPress : Pour gérer efficacement la charge et assurer une haute disponibilité, un système d'autoscaling est mis en place pour WordPress. Ce système ajuste automatiquement le nombre de réplicas en fonction de la charge de trafic, garantissant ainsi que le site reste disponible et réactif même pendant les pics de fréquentation. Cela aide également à optimiser les coûts en ne consommant des ressources supplémentaires que lorsque cela est nécessaire.

Maintenabilité et Évolutivité

La capacité à maintenir et à faire évoluer l'infrastructure est essentielle pour répondre aux besoins changeants et pour intégrer les nouvelles technologies :

- Conception Modulaire et Évolutive : L'infrastructure est conçue de manière modulaire, ce qui permet d'ajouter, de modifier ou de retirer des composants sans perturber le reste du système. Cette approche facilite les mises à jour, les tests et l'intégration de nouvelles fonctionnalités ou services. L'infrastructure est également conçue pour être évolutive, permettant d'augmenter facilement les ressources en fonction de l'augmentation des besoins.
- Automatisation et Intégration Continue : L'automatisation des processus de déploiement, de test et de surveillance à travers des outils comme Jenkins, Terraform et Helm contribue à une maintenabilité accrue. L'intégration continue garantit que les modifications apportées au code ou à la configuration sont automatiquement testées et déployées, réduisant ainsi les erreurs humaines et accélérant le cycle de développement.

4. Contraintes

Contraintes Temporelles

La gestion efficace du temps est un aspect critique dans le cadre de ce projet. Les contraintes temporelles imposent un calendrier strict pour la mise en œuvre et le déploiement de l'infrastructure et des applications. La planification détaillée des tâches et le suivi rigoureux du

progrès sont essentiels pour respecter les délais fixés. Cela implique une coordination étroite entre les différentes phases du projet, notamment la configuration de l'infrastructure, le déploiement des applications, les tests, la mise en place de la sécurité et la surveillance. L'adoption de méthodes agiles et l'utilisation d'outils d'automatisation comme Jenkins et Terraform jouent un rôle clé pour optimiser le temps de développement et garantir une livraison dans les délais.

Contraintes de Localisation

L'obligation de l'utilisation de la région de déploiement de l'infrastructure sur Azure, spécifiquement en Europe de l'Ouest (westeurope), peut influencer plusieurs aspects du projet. Cette contrainte de localisation peut impacter la latence, la disponibilité des services et la conformité réglementaire. Il est donc crucial de s'assurer que tous les services et ressources nécessaires sont disponibles et performants dans cette région spécifique. De plus, la localisation doit respecter les exigences légales et réglementaires en matière de protection des données et de confidentialité, notamment le RGPD (Règlement Général sur la Protection des Données).

Contraintes Technologiques

La compatibilité et l'intégration des différentes technologies et outils constituent une contrainte significative. Chaque composant, qu'il s'agisse de Docker, Kubernetes, Terraform, Jenkins, ou d'autres outils de sécurité et de surveillance, doit être soigneusement choisi et configuré pour fonctionner harmonieusement au sein de l'écosystème Azure. Cette intégration nécessite une bonne compréhension des caractéristiques techniques de chaque outil ainsi que des meilleures pratiques pour leur intégration. La compatibilité des versions, la gestion des dépendances et la configuration réseau sont des aspects particulièrement importants à surveiller pour éviter les conflits et garantir une intégration fluide.

Contraintes de Sécurité

La sécurité est une préoccupation primordiale dans tous les aspects du projet. Les exigences minimales de sécurité incluent le chiffrement des communications via TLS, la protection contre les attaques par force brute, la réalisation de scans de vulnérabilité avec WPScan et Sonar Cloud, et la mise en place d'une surveillance efficace avec Prometheus, Grafana, et Azure Log Analytics. Ces mesures doivent être mises en place tout en équilibrant la facilité d'utilisation et la performance. Il est également important de se conformer aux normes de sécurité du secteur, ainsi qu'aux politiques et pratiques recommandées par Azure pour garantir la sécurité des données et des applications dans le cloud

II. Document d'architecture technique

1. Besoin fonctionnels

Description des fonctionnalités du système, telles que le déploiement automatisé, la gestion et la supervision d'une infrastructure WordPress sur AKS.

- Que fait cette partie ?
- Historique du projet et évolution des besoins.
- Attentes principales : efficacité, facilité de gestion, etc.
- Contraintes métiers impactant l'architecture, telles que la nécessité d'une haute disponibilité ou la conformité aux normes de sécurité.

2. Besoins non fonctionnels

Performances attendues du système, comme le temps de réponse, la capacité de charge, etc.

- Disponibilité et fiabilité requises pour l'infrastructure et les applications.
- Spécifications techniques détaillées : utilisation de langages, infrastructures, nombre d'utilisateurs, localisation géographique, etc.

3. Représentation fonctionnelle

Schémas illustrant le flux de travail, les interactions entre les différents composants (Jenkins, Terraform, Kubernetes, WordPress, etc.).

- Description de la façon dont chaque composant contribue aux objectifs du projet.

4. Représentation applicative

Détails des applications utilisées (WordPress, Jenkins, etc.), y compris leur configuration, leur intégration et leur rôle dans l'architecture globale.

- Diagrammes illustrant l'architecture applicative et les interactions entre les différentes applications.

5. Représentation infrastructure

Description de l'infrastructure cloud (Azure) et du cluster Kubernetes.

- Détails sur le déploiement, la configuration et la gestion des ressources cloud.

6. Représentation opérationnelle

Description des processus opérationnels, y compris le déploiement continu, la supervision et la maintenance.

- Protocoles de gestion des incidents, de mise à jour, et de récupération après sinistre.

7. Coûts

Estimation détaillée des coûts associés à l'infrastructure, aux outils, et à la maintenance.

III. Réalisations

1. Mise en place de Jenkins

Jenkins - Docker-compose - Github - Secrets - Plugins et tools

2. Déploiement de l’infrastructure

Terraform - Helm et kubernetes – Wordpress - MariaDB

3. Mise en place de la sécurité

WPScan – Sonarcloud – TLS

4. Supervision de l’infrastructure

Prometheus - Grafana - Log ANALYTICS

IV. Recherche

Conclusion

Le projet comprend 9 des 10 compétences professionnelles à couvrir dans le but de l’acquisition du titre professionnel “Administrateur système DevOps“ :

Compétence	Action
Automatiser la création de serveurs à l’aide de scripts	Pipeline d’automatisation sur Jenkins

Automatiser le déploiement d'une infrastructure	Déploiements avec Terraform (Resource Group, AKS, Log Analytics, Machine Virtuelle)
Sécuriser l’infrastructure	Mise en place du TLS, de WPScan et de Sonar Cloud
Mettre l’infrastructure en production dans le cloud	Infrastructure en production sur Azure
Gérer le stockage des données	MariaDB comme base de données de Wordpress, Stockage Azure pour stocker les résultats de WPScan
Gérer des containers	Container Docker pour le déploiement de Jenkins
Automatiser la mise en production d’une application avec une plateforme	Déploiements avec Kubernetes et Helm (Wordpress, MariaDB, autoscaling, Prometheus, Grafana, Loki, Certmanager, Traefik)
Définir et mettre en place des statistiques de services	Utilisation de Log Analytics
Exploiter une solution de supervision	Utilisation de Prometheus et Grafana

Par manque de temps, la 10e compétence (Préparer un environnement de test) n’a pas pu être intégrée au projet.

Annexes

Outils utilisés