**Cahier des Charges avec Intégration de la Méthode Scrum**

**Introduction**

Ce projet vise à réaliser une infrastructure DevOps complète en utilisant Terraform pour provisionner une VM sur AWS, Docker pour la conteneurisation des applications, Ansible pour l'automatisation du déploiement, et Prometheus ainsi que Zabbix pour la supervision et la sécurité. La méthode Scrum sera utilisée pour gérer ce projet en sprints.

**Diagramme de Gantt**

Une image contenant texte, capture d’écran, nombre, Police

Description générée automatiquement

|----------------------------------- Diagramme de Gantt -----------------------------------|

| Tâches | Juin | Juillet | Août |

| | 1 | 2 | 3 | 4 | 1 | 2 | 3 | 4 | 1 | 2 |

|--------------------------------------|----|----|----|----|----|----|----|----|----|----|

| 1. Infrastructure | | | | | | | | | | |

| - Installer Terraform | XX | XX | | | | | | | | |

| - Configurer AWS | XX | XX | | | | | | | | |

| - Provisionner VM | | | XX | XX | | | | | | |

| - Installer Docker | | | XX | XX | | | | | | |

| - Télécharger images Docker | | | XX | XX | | | | | | |

|--------------------------------------|----|----|----|----|----|----|----|----|----|----|

| 2. Containers | | | | | | | | | | |

| - Déployer container Nginx | | | | | XX | XX | | | | |

| - Déployer container Git | | | | | XX | XX | | | | |

| - Déployer container Prometheus | | | | | XX | XX | | | | |

|--------------------------------------|----|----|----|----|----|----|----|----|----|----|

| 3. Automatisation | | | | | | | | | | |

| - Créer playbooks Ansible | | | | | | | XX | XX | | |

| - Automatiser configuration Prometheus| | | | | | | XX | XX | | |

|--------------------------------------|----|----|----|----|----|----|----|----|----|----|

| 4. Développement | | | | | | | | | | |

| - Créer site web HTML | | | | | | | | | XX | XX |

| - Déployer site web sur Nginx | | | | | | | | | XX | XX |

| - Configurer Git pour versionning | | | | | | | | | XX | XX |

|--------------------------------------|----|----|----|----|----|----|----|----|----|----|

| 5. Supervision et Sécurité | | | | | | | | | | |

| - Configurer Prometheus | | | | | | | | | XX | XX |

| - Mettre en place Zabbix | | | | | | | | | XX | XX

|--------------------------------------|----|----|----|----|----|----|----|----|----|----|

**Sprint 1 : Infrastructure**

* Installer Terraform.
* Configurer AWS.
* Provisionner une VM.
* Installer Docker.

**Sprint 2 : Containers**

* Télécharger les images Docker.
* Déployer les containers Nginx, Git et Prometheus.

**Sprint 3 : Automatisation**

* Créer les playbooks Ansible.
* Automatiser la configuration de Prometheus.

**Sprint 4 : Développement**

* Créer un site web en HTML.
* Déployer le site web sur Nginx.
* Configurer Git pour le versionning.

**Sprint 5 : Supervision et Sécurité**

* Configurer Prometheus.
* Mettre en place Zabbix pour la sécurité.

**Étapes de Réalisation du Projet**

**1. Installation de Terraform et Provisionnement de la VM**

1. **Installation de Terraform**
   * Télécharger et installer Terraform.

#sudo apt-get update && sudo apt-get install -y wget unzip

wget https://releases.hashicorp.com/terraform/1.0.0/terraform\_1.0.0\_linux\_amd64.zip

unzip terraform\_1.0.0\_linux\_amd64.zip

sudo mv terraform /usr/local/bin/

terraform -v

1. **Configuration AWS pour Terraform**
   * Configurer les identifiants AWS pour Terraform.

bash

Copier le code

export AWS\_ACCESS\_KEY\_ID="your-access-key-id"

export AWS\_SECRET\_ACCESS\_KEY="your-secret-access-key"

export AWS\_DEFAULT\_REGION="us-west-2"

1. **Provisionner une VM sur AWS**

# main.tf

provider "aws" {

region = "us-west-2"

}

resource "aws\_instance" "example" {

ami = "ami-0c55b159cbfafe1f0" # Amazon Linux 2 AMI

instance\_type = "t2.micro"

tags = {

Name = "ExampleInstance"

}

}

#terraform init

terraform apply

**2. Installation de Docker et Téléchargement des Images**

1. **Installer Docker sur la VM**

sudo apt-get update

sudo apt-get install -y docker.io

sudo systemctl start docker

sudo systemctl enable docker

1. **Télécharger les Images Docker**

sudo docker pull nginx

sudo docker pull git

sudo docker pull prom/prometheus

**3. Automatisation avec Ansible**

1. **Installation d'Ansible**
   * Installer Ansible sur votre machine locale.

sudo apt-get update

sudo apt-get install -y ansible

1. **Création des Playbooks Ansible**
   * Créer un playbook pour déployer les containers Docker.

# deploy\_containers.yml

- hosts: all

become: yes

tasks:

- name: Ensure Docker is installed

apt:

name: docker.io

state: present

- name: Start Docker service

systemd:

name: docker

state: started

enabled: yes

- name: Run Nginx container

docker\_container:

name: nginx

image: nginx

state: started

ports:

- "80:80"

- name: Run Git container

docker\_container:

name: git

image: git

state: started

- name: Run Prometheus container

docker\_container:

name: prometheus

image: prom/prometheus

state: started

ports:

- "9090:9090"

* + Exécuter le playbook Ansible.

ansible-playbook -i inventory deploy\_containers.yml

**4. Développement et Configuration**

1. **Création d'un Site Web en HTML**
   * Créer un fichier index.html.

<!-- index.html -->

<!DOCTYPE html>

<html>

<head>

<title>Page de Test</title>

</head>

<body>

<h1>Bienvenue sur notre site web!</h1>

</body>

</html>

* + Copier le fichier index.html dans le container Nginx.

docker cp index.html nginx:/usr/share/nginx/html/index.html

1. **Configuration de Git pour le Versionning**
   * Initialiser un dépôt Git.

docker exec -it git git init /path/to/repo

docker exec -it git git config --global user.email "you@example.com"

docker exec -it git git config --global user.name "Your Name"

* + Ajouter et commiter le site web.

docker exec -it git git -C /path/to/repo add index.html

docker exec -it git git -C /path/to/repo commit -m "Ajout du site web initial"

**5. Supervision et Sécurité**

1. **Configuration de Prometheus**
   * Créer un fichier de configuration Prometheus.

bash

Copier le code

cat <<EOF > prometheus.yml

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'docker'

static\_configs:

- targets: ['localhost:9090']

EOF

* + Copier le fichier de configuration dans le container Prometheus.

bash

Copier le code

docker cp prometheus.yml prometheus:/etc/prometheus/prometheus.yml

docker restart prometheus

1. **Mise en Place de Zabbix pour la Sécurité**
   * Installer Zabbix sur la VM.
   * Configurer Zabbix pour monitorer les containers.

Une image contenant texte, capture d’écran, logiciel, Logiciel multimédia

Description générée automatiquement

**Schéma de l'Architecture Globale**

+-------------------+

| AWS EC2 |

|-------------------|

| Terraform |

+--------+----------+

|

+---------v---------+

| Docker |

+---------+----------+

|

+---------+---------+---------+---------+

| | | | |

+--v--+ +--v--+ +--v--+ +--v--+ +--v--+

| Nginx| | Git | |Prometheus| | Zabbix |

+------+ +------+ +----------+ +------+

**Compétences Acquises**

1. **Automatiser le Déploiement d'une Infrastructure Cloud**
   * **Automatiser la création de serveurs à l'aide de scripts** : Utilisation de Terraform pour provisionner des VMs sur AWS.
   * **Automatiser le déploiement d'une infrastructure** : Utilisation d'Ansible pour déployer et configurer les containers.
   * **Sécuriser l'infrastructure** : Mise en place de Wazuh pour la sécurité des containers.
   * **Mettre l'infrastructure en production dans le cloud** : Déploiement des services sur AWS.
2. **Déployer en Continu une Application**
   * **Préparer un environnement de test** : Utilisation de Docker pour isoler les applications dans des containers.
   * **Gérer le stockage des données** : Utilisation de Git pour le versionning du site web.
   * **Gérer des containers** : Utilisation de Docker pour le déploiement des services.
   * **Automatiser la mise en production d'une application avec une plateforme** : Utilisation d'Ansible pour automatiser le déploiement.
3. **Superviser les Services Déployés**
   * **Définir et mettre en place des statistiques de services** : Configuration de Prometheus pour la supervision des containers.
   * **Superviser les services déployés** : Utilisation de Prometheus pour monitorer les performances des containers.

**Intégration de la Méthode Scrum**

**1. Définition des Rôles**

* **Scrum Master (SM)** : Facilite le processus Scrum, aide l'équipe à travailler efficacement et supprime les obstacles.
* **Équipe de Développement** : DevOps étudiante, responsables de la réalisation des tâches. Responsable de la vision du projet et la gestion du backlog.

**2. Artefacts Scrum**

* **Product Backlog** : Liste des fonctionnalités, tâches et améliorations à réaliser.
* **Sprint Backlog** : Ensemble des éléments du Product Backlog sélectionnés pour un sprint et un plan pour les délivrer.
* **Incrément** : Version fonctionnelle du produit à la fin de chaque sprint.

**3. Événements Scrum**

* **Sprint Planning** : Réunion pour planifier le travail du sprint, définir le Sprint Goal et sélectionner les items du Product Backlog.
* **Daily Scrum** : Réunion quotidienne de 15 minutes pour synchroniser les activités et planifier les 24 prochaines heures.
* **Sprint Review** : Réunion à la fin du sprint pour inspecter l'incrément et adapter le projet Backlog si nécessaire.
* **Sprint Retrospective** : Réunion après le Sprint Review pour discuter de ce qui a bien fonctionné et ce qui peut être amélioré.

**4. Planification des Sprints**

Chaque sprint durera deux semaines.

Stop….

Projet try2

**Introduction**

Le projet DevOps vise à déployer une infrastructure automatisée utilisant Terraform, Docker, Ansible, et des outils de supervision comme Prometheus et Wazuh. L'objectif est de provisionner une VM gratuite sur le cloud AWS ou une VM Linux, d'y installer Docker, et de télécharger et déployer les images Docker nécessaires. L'ensemble du processus sera automatisé avec Ansible, incluant la configuration d'un serveur de supervision pour monitorer les containers. En plus, la sécurité de l'infrastructure sera renforcée en utilisant Wazuh.

**Cahier des Charges**

**Objectifs**

* Provisionner une VM sur AWS ou Linux avec Terraform.
* Installer Docker et télécharger les images Docker (Nginx, Git, Prometheus).
* Automatiser l'installation et le déploiement des images Docker avec Ansible.
* Créer un site web HTML hébergé sur le container Nginx.
* Utiliser le container Git pour le versionning du site web.
* Monitorer les containers avec Prometheus et automatiser la configuration avec un script Bash.
* Sécuriser l'infrastructure globale avec Wazuh.

**Diagramme de Gantt**

|---------------------------- Diagramme de Gantt ----------------------------|

| Tâches | Semaines |

|--------------------------------------|--------------------------------------|

| 1. Infrastructure | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

| - Installer Terraform | XX | | | | | | | |

| - Configurer AWS | XX | | | | | | | |

| - Provisionner VM | | XX | | | | | | |

| - Installer Docker | | XX | | | | | | |

| - Télécharger images Docker | | XX | | | | | | |

|--------------------------------------|--------------------------------------|

| 2. Containers | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

| - Déployer container Nginx | | | XX | | | | | |

| - Déployer container Git | | | XX | | | | | |

| - Déployer container Prometheus | | | XX | | | | | |

|--------------------------------------|--------------------------------------|

| 3. Automatisation | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

| - Créer playbooks Ansible | | | | XX | | | | |

| - Automatiser configuration Prometheus| | | | XX | | | | |

|--------------------------------------|--------------------------------------|

| 4. Développement | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

| - Créer site web HTML | | | | | XX | | | |

| - Déployer site web sur Nginx | | | | | XX | | | |

| - Configurer Git pour versionning | | | | | XX | | | |

|--------------------------------------|--------------------------------------|

| 5. Supervision et Sécurité | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

| - Configurer Prometheus | | | | | | XX | | |

| - Mettre en place Wazuh | | | | | | XX | | |

|--------------------------------------|--------------------------------------|

**Étapes de Réalisation du Projet**

**1. Installation de Terraform et Provisionnement de la VM**

1. **Installation de Terraform**
   * Télécharger et installer Terraform.

bash

Copier le code

sudo apt-get update && sudo apt-get install -y wget unzip

wget https://releases.hashicorp.com/terraform/1.0.0/terraform\_1.0.0\_linux\_amd64.zip

unzip terraform\_1.0.0\_linux\_amd64.zip

sudo mv terraform /usr/local/bin/

terraform -v

1. **Configuration AWS pour Terraform**
   * Configurer les identifiants AWS pour Terraform.

bash

Copier le code

export AWS\_ACCESS\_KEY\_ID="your-access-key-id"

export AWS\_SECRET\_ACCESS\_KEY="your-secret-access-key"

export AWS\_DEFAULT\_REGION="us-west-2"

1. **Provisionner une VM sur AWS**

hcl

Copier le code

# main.tf

provider "aws" {

region = "us-west-2"

}

resource "aws\_instance" "example" {

ami = "ami-0c55b159cbfafe1f0" # Amazon Linux 2 AMI

instance\_type = "t2.micro"

tags = {

Name = "ExampleInstance"

}

}

bash

Copier le code

terraform init

terraform apply

**2. Installation de Docker et Téléchargement des Images**

1. **Installer Docker sur la VM**

bash

Copier le code

sudo apt-get update

sudo apt-get install -y docker.io

sudo systemctl start docker

sudo systemctl enable docker

1. **Télécharger les Images Docker**

bash

Copier le code

sudo docker pull nginx

sudo docker pull git

sudo docker pull prom/prometheus

**3. Automatisation avec Ansible**

1. **Installation d'Ansible**
   * Installer Ansible sur votre machine locale.

bash

Copier le code

sudo apt-get update

sudo apt-get install -y ansible

1. **Création des Playbooks Ansible**
   * Créer un playbook pour déployer les containers Docker.

yaml

Copier le code

# deploy\_containers.yml

- hosts: all

become: yes

tasks:

- name: Ensure Docker is installed

apt:

name: docker.io

state: present

- name: Start Docker service

systemd:

name: docker

state: started

enabled: yes

- name: Run Nginx container

docker\_container:

name: nginx

image: nginx

state: started

ports:

- "80:80"

- name: Run Git container

docker\_container:

name: git

image: git

state: started

- name: Run Prometheus container

docker\_container:

name: prometheus

image: prom/prometheus

state: started

ports:

- "9090:9090"

* + Exécuter le playbook Ansible.

bash

Copier le code

ansible-playbook -i inventory deploy\_containers.yml

**4. Développement et Configuration**

1. **Création d'un Site Web en HTML**
   * Créer un fichier index.html.

html

Copier le code

<!-- index.html -->

<!DOCTYPE html>

<html>

<head>

<title>Page de Test</title>

</head>

<body>

<h1>Bienvenue sur notre site web!</h1>

</body>

</html>

* + Copier le fichier index.html dans le container Nginx.

bash

Copier le code

docker cp index.html nginx:/usr/share/nginx/html/index.html

1. **Configuration de Git pour le Versionning**
   * Initialiser un dépôt Git.

bash

Copier le code

docker exec -it git git init /path/to/repo

docker exec -it git git config --global user.email "you@example.com"

docker exec -it git git config --global user.name "Your Name"

* + Ajouter et commiter le site web.

bash

Copier le code

docker exec -it git git -C /path/to/repo add index.html

docker exec -it git git -C /path/to/repo commit -m "Ajout du site web initial"

**5. Supervision et Sécurité**

1. **Configuration de Prometheus**
   * Créer un fichier de configuration Prometheus.

bash

Copier le code

cat <<EOF > prometheus.yml

global:

scrape\_interval: 15s

scrape\_configs:

- job\_name: 'docker'

static\_configs:

- targets: ['localhost:9090']

EOF

* + Copier le fichier de configuration dans le container Prometheus.

bash

Copier le code

docker cp prometheus.yml prometheus:/etc/prometheus/prometheus.yml

docker restart prometheus

1. **Mise en Place de Wazuh pour la Sécurité**
   * Installer Wazuh sur la VM.

bash

Copier le code

wget -qO - https://packages.wazuh.com/key/GPG-KEY-WAZUH | sudo apt-key add -

echo "deb https://packages.wazuh.com/4.x/apt/ stable main" | sudo tee /etc/apt/sources.list.d/wazuh.list

sudo apt-get update

sudo apt-get install -y wazuh-manager

* + Configurer Wazuh pour monitorer les containers.

bash

Copier le code

# Configurer Wazuh

**Schéma de l'Architecture Globale**

+-------------------+

| AWS EC2 |

| +---------------+ |

| | Docker | |

| | +-----------+ | |

| | | Nginx | | |

| | +-----------+ | |

| | +-----------+ | |

| | | Git | | |

| | +-----------+ | |

| | +-----------+ | |

| | |Prometheus | | |

| | +-----------+ | |

| | +-----------+ | |

| | | Wazuh | | |

| | +-----------+ | |

| +---------------+ |

+-------------------+

**Compétences Intégrées**

1. **Automatisation de l’infrastructure dans le cloud**
   * Utilisation de Terraform pour provisionner des instances AWS.
   * Automatisation de la création de serveurs avec des scripts.
   * Déploiement de l’infrastructure avec des outils de gestion de configuration comme Ansible.
   * Sécurisation de l'infrastructure avec Wazuh.
   * Mise en production de l’infrastructure dans le cloud.
2. **Déploiement continu d’une application**
   * Utilisation de Docker pour containeriser les applications.
   * Création d’un environnement de test avec des containers Docker.
   * Gestion des données avec Git pour le versionning du site web.
   * Automatisation du déploiement d’applications avec Ansible.
3. **Supervision des services déployés**
   * Utilisation de Prometheus pour superviser les services déployés.
   * Définition et mise en place des statistiques de services avec Prometheus.
   * Utilisation de Wazuh pour la sécurité et la supervision des containers.

En suivant ces étapes et en respectant ce cahier des charges, vous serez en mesure de démontrer vos compétences en automatisation, déploiement, supervision, et sécurisation d’une infrastructure DevOps.