Plan de Déploiement – Projet Blockchain

En collaboration avec l'équipe d'exploitation, nous procédons à un déploiement de test de notre projet selon les étapes suivantes :

1. Création de la machine virtuelle

Responsable : Administrateur Réseau

Cette tâche consiste à créer une VM avec des ressources suffisantes pour héberger le backend, Hyperledger Fabric, et d'autres services nécessaires. La VM est configurée comme suit :

- RAM : 8 Go (fréquence 3600 MHz)
- CPU : 4 cœurs (fréquence > 2.4 GHz)
- Threads : 8 threads
- Disque : 200 Go SSD M.2
- Système d'exploitation : Ubuntu Server 24.04 LTS Lien de téléchargement : https://ubuntu.com/download/server/thank-you?version=24.04.2&architecture=amd64&lts=true
- Identifiants de connexion : `pub-ns1` / `esss2025`

2. Intégration de la VM dans un VLAN isolé

Responsable : Administrateur Système

Cette tâche a pour but de sécuriser la machine en l'intégrant dans un VLAN isolé avec l'adresse IP : `10.10.29.180`. Tous les ports sont ouverts pour permettre la communication avec les services.

3. Connexion à la VM via SSH

Responsable : Nous

Connexion sécurisée à la VM via la commande :

```
ssh pub-ns1@10.10.29.180
```

4. Clonage du dépôt Git du projet

Responsable : Nous

Téléchargement du code source via GitHub :

```
git clone https://github.com/ZebdaYacine/cnr-payment-blockchain.git
```

5. Installation de Hyperledger Fabric et dépendances

Responsable : Nous

Installation des outils nécessaires : Go, jq, curl, Hyperledger Fabric :

```
cd cnr-payment-blockchain/script
sudo chmod +x ./install.sh
sudo ./install.sh
```

6. Configuration des variables d'environnement

Responsable : Nous

Ajout des variables d'environnement dans `~/.bashrc` ou `/etc/environment` pour rendre Go, jq, et Fabric accessibles système-wide.

7. Déploiement du smart contract

Responsable : Nous

Copie du smart contract dans le dossier approprié de Fabric pour le déploiement :

```
cd cnr-payment-blockchain/script
cp -r smart-contract.go
~/fabric-samples/asset-transfer-basic/chaincode-go/chaincode
```

8. Déploiement du réseau Hyperledger Fabric

Responsable : Nous

Lancement du réseau Fabric localement :

```
cd fabric-simple/test-network
sudo chmod +x deploy-cc.sh
sudo ./deploy-cc.sh
```

9. Configuration de la boîte mail de test

Responsable : Administrateur Système

Définir les paramètres SMTP dans l'environnement de l'API Go :

```
SMTP_HOST=10.10.7.132
SMTP_PORT=587
SMTP_USER=echeance@cnr-dz.com
SMTP_PASS=@zerty2468++
```

Cela permet à l'API d'envoyer des notifications par e-mail.

10. Configuration des noms de domaine

Responsable : Sous-Directeur d'exploitation

Cette tâche permet d'accéder à l'application via des noms de domaines personnalisés. Les pare-feux (firewall) et WAF sont configurés pour autoriser le trafic :

- Backend : `te.cnr.api`
- Frontend : `tecnr-dz.com`

11. Exécution via Docker Compose

Responsable : Nous

Lancement de l'application à l'aide de Docker Compose :

```
cd cnr-payment-blockchain
sudo docker-compose up --build
```

Ce fichier `docker-compose.yml` contient les services suivants :

- **backend** : application API développée en Go, exposée sur le port `9000:3000`, utilisant un fichier `.env`, et dépendant de MongoDB et SFTP. Il accède également aux fichiers Fabric montés via volume.
- **mongodb** : base de données MongoDB, exposée sur le port `27017`, avec initialisation depuis `./mongo-init`.
- **sftp** : serveur SFTP basé sur l'image `atmoz/sftp`, exposé sur `2222:22`, pour le transfert de fichiers.
- **frontend** : application React, exposée sur le port `3000:80`, dépendante du backend.

12. Tests fonctionnels & validation

Responsable : Nous et le Sous-Directeur d'exploitation

Cette étape consiste à effectuer une batterie de tests afin de valider le bon fonctionnement de l'ensemble du système. Cela inclut :

- Vérifier que les **API Go** fonctionnent correctement
- Confirmer le **déploiement réussi du réseau blockchain**
- Tester la **réception des e-mails** envoyés par le backend
- **Téléverser des fichiers** vers le serveur **SFTP**
- **Détecter toute alerte** liée à la **perte d'intégrité des fichiers** présents dans le SFTP