****

**420-414 - Infonuagique**

**Examen final**

**Rapport d’implémentation**

**DA : 2387923**

**Nom, prénom : El Jal Yassine**

**/ 25**

## Table des matières

[**1 - Introduction 3**](#_67nisotpon4)

[**2 - Infrastructure 3**](#_j62286n7fjad)

[2.1 - Schéma de l’infrastructure réseau (AWS) 3](#_j6ma99rr0pvo)

[2.2 - Description des composants réseau 3](#_u1iicythxd2)

[2.3 - Groupes de sécurité 4](#_t8x2t6vlje6f)

[2.4 - Configuration de l’instance EC2 4](#_q90eecp77fws)

[**3 - Déploiement des services : Docker et Docker Compose 5**](#_ugbwfa41gfcq)

[3.1 - Architecture Docker 5](#_7jmlpcyqzfaa)

[3.2 - Services déployés 5](#_jrpsv5l9c7e9)

[3.3 - Fichier docker-compose.yml 5](#_b8kaa8lz1lz0)

[**4 - Implémentation d’un proxy inverse (Traefik) 5**](#_rlbo8nkihx44)

[**5 - Nom de domaine 5**](#_qclcglxxb2y8)

[**6 - Haute disponibilité et résilience 5**](#_fhaps1caarg5)

[**7 - Automation avec Terraform (optionnel) 6**](#_2uz4mklkfs7c)

[**8 - Problèmes rencontrés et solutions apportées 6**](#_3uc3rhl53fye)

[**9 - Conclusion 6**](#_89m3hbrc49ea)

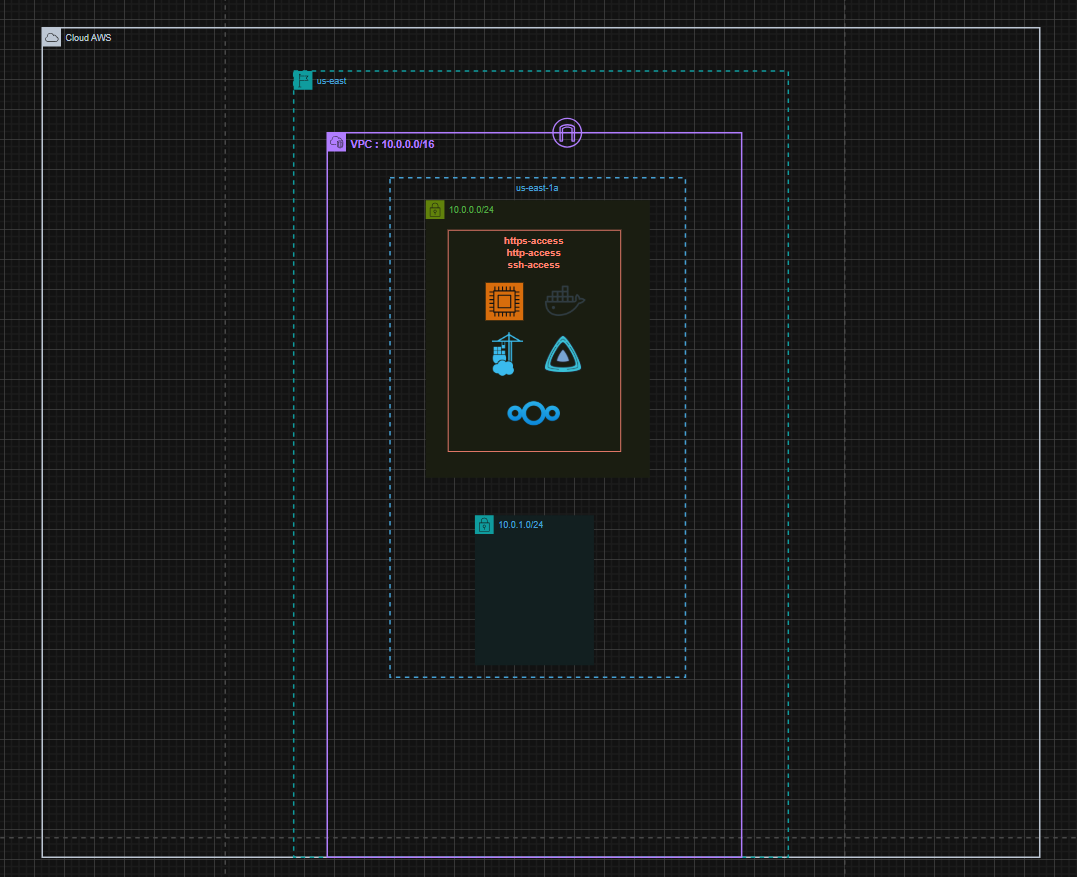
[**Annexes (facultatif) 6**](#_5rgqrmomrf0j)

## 1 - Introduction

*Le projet a pour but de déployer des services à l’aide de Docker et de AWS. Pour gérer ces services je vais utiliser docker compose. Les services que j’ai choisis sont : jellyfin, portainer et nextcloud*

## 2 - Infrastructure

##### 2.1 - Schéma de l’infrastructure réseau (AWS)



##### 2.2 - Description des composants réseau

*Remplissez le tableau suivant avec les détails de votre VPC :*

|  |  |
| --- | --- |
| ***Nom du VPC*** | VPC-tpFinal |
| ***Région*** | us-east-1 |
| ***Plage d’adresses IPv4*** | 10.0.0.0/16 |
| ***Configuration de la connectivité***  ***(Passerelle Internet ou autre)*** | igw |

*Remplissez le tableau suivant avec les détails de chaque table de routage (dupliquez le tableau pour chaque table de routage créé/utilisée :*

|  |  |
| --- | --- |
| ***Nom de la table de routage*** | tpFinal-rtb-public |
| ***Règles de routage*** | |
| ***Destination*** | ***Cible*** |
| local | tpFinal-public-1 |
| igw |  |

|  |  |
| --- | --- |
| ***Nom de la table de routage*** | tpFinal-rtb-private |
| ***Règles de routage*** | |
| ***Destination*** | ***Cible*** |
| local | tpFinal-private-1 |
|  |  |

*Remplissez le tableau suivant avec les détails de chaque sous-réseau (dupliquez le tableau pour chaque sous-réseau de votre architecture :*

|  |  |
| --- | --- |
| ***Nom du sous-réseau*** | tpFinal-public-1 |
| ***Zone de disponibilité*** | us-east-1a |
| ***Plage d’adresse IPv4*** | 10.0.0.0/24 |
| ***Public/Privé*** | Public |
| ***Table de routage associée*** | tpFinal-rtb-public |

|  |  |
| --- | --- |
| ***Nom du sous-réseau*** | tpFinal-private-1 |
| ***Zone de disponibilité*** | us-east-1a |
| ***Plage d’adresse IPv4*** | 10.0.1.0/24 |
| ***Public/Privé*** | Privé |
| ***Table de routage associée*** | tpFinal-rtb-private |

##### 2.3 - Groupes de sécurité

*Remplissez le tableau suivant pour chaque groupe de sécurité implémenté (dupliquez le tableau pour chaque groupe de sécurité créé :*

|  |  |  |  |
| --- | --- | --- | --- |
| ***Nom du groupe de sécurité*** | | ssh-access | |
| ***Description*** | | Allow SSH inbound traffic | |
| ***Règles entrantes*** | | | |
| ***Type*** | ***Protocole*** | ***Plage de ports*** | ***Source*** |
| SSH | tcp | 22 | 0.0.0.0/0 |
| ***Règles sortantes*** | | | |
| ***Type*** | ***Protocole*** | ***Plage de ports*** | ***Source*** |
| SSH | -1 | 0 | 0.0.0.0/0 |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Nom du groupe de sécurité*** | | http-access | |
| ***Description*** | | Allow HTTP inbound traffic | |
| ***Règles entrantes*** | | | |
| ***Type*** | ***Protocole*** | ***Plage de ports*** | ***Source*** |
| HTTP | tcp | 80 | 0.0.0.0/0 |
| ***Règles sortantes*** | | | |
| ***Type*** | ***Protocole*** | ***Plage de ports*** | ***Source*** |
| HTTP | -1 | 0 | 0.0.0.0/0 |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Nom du groupe de sécurité*** | | https-access | |
| ***Description*** | | Allow HTTPS inbound traffic | |
| ***Règles entrantes*** | | | |
| ***Type*** | ***Protocole*** | ***Plage de ports*** | ***Source*** |
| HTTPS | tcp | 443 | 0.0.0.0/0 |
| ***Règles sortantes*** | | | |
| ***Type*** | ***Protocole*** | ***Plage de ports*** | ***Source*** |
| HTTPs | -1 | 0 | 0.0.0.0/0 |

***Justifiez la raison du choix de chaque règle.***

##### 2.4 - Configuration de l’instance EC2

*Remplissez le tableau des caractéristiques techniques de l’instance EC2 suivant :*

|  |  |
| --- | --- |
| ***Nom de l’instance*** | public-instance |
| ***Type de l’instance*** | t2.large |
| ***AMI*** | Ubuntu 24.04 LTS |
| ***VPC*** | VPC-tpFinal |
| ***Sous-réseau*** | tpFinal-public-1 |
| ***Groupe(s) de sécurité*** | http-access, https-access, ssh-access |

*Dans l’instance je vais installer docker, faire un fichier compose.yml, faire un fichier .env et je vais lancer mes conteneurs avec docker compose up -d*

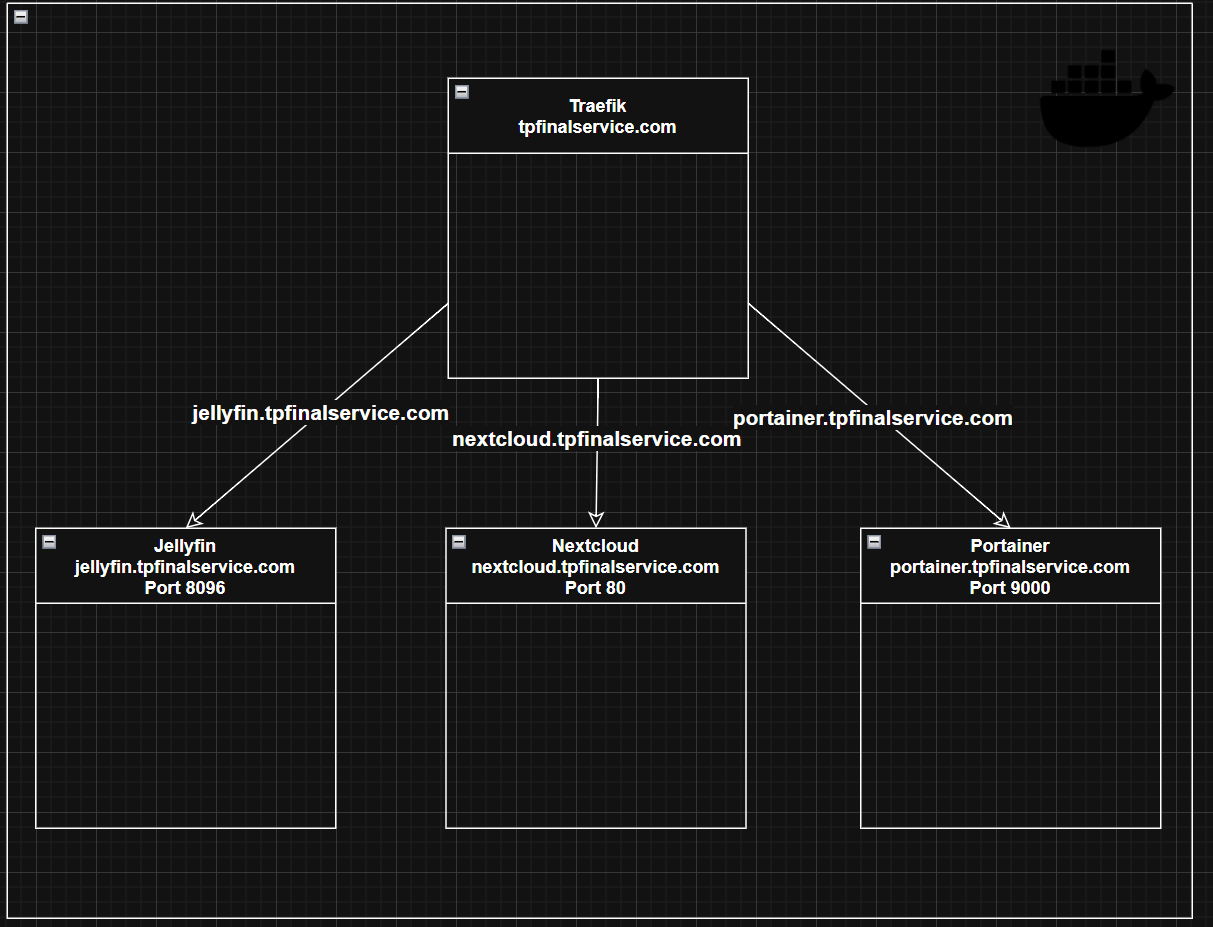
*Mon fichier compose :* [*Compose*](https://github.com/yassineeljal/tp-final/blob/main/note/compose.yml)

*Mon fichier .env :* [*.env*](https://github.com/yassineeljal/tp-final/blob/main/note/.env)

*Le README des commandes dans l’instance :* [*README*](https://github.com/yassineeljal/tp-final/blob/main/note/README.md)

## 3 - Déploiement des services : Docker et Docker Compose

##### 3.1 - Architecture Docker



##### 3.2 - Services déployés

*Listez et décrivez les services choisis parmi ceux proposés.*

*Pour chaque service précisez :*

* *Son rôle/sa description*
* *Port(s) utilisé(s)*
* *Configuration spécifique (volumes, variables d’environnement, dépendances…)*

##### 3.3 - Fichier *docker-compose.yml*

*Mon fichier compose :* [*Compose*](https://github.com/yassineeljal/tp-final/blob/main/note/compose.yml)

*Présentez et commentez votre fichier docker-compose.yml :*

* *Organisation en services*
* *Images utilisées*
* *Réseaux et volumes*
* *Redémarrage automatique (restart)*
* *Dépendances (depends\_on)*
* *etc…*

*J’ai mis 3 services :*

* *portainer*
* *nextcloud*
* *jellyfin*

*Images utilisées : Les images officielles des services*

*Réseau : cloud 🡪 permettre d’avoir tout dans le même réseau*

*Volume : Comme des backups. Si je perds les données ou quoi que ce soit, ils ne sont pas réellement perdus parce que j’ai mis des volumes qui stock.*

*Redémarrage : Si les conteneurs crash, ils restart*

*Depends\_on : je l’ai mis qu’a nextcloud parce je veux qu’il se lance après avoir setup la base de données pour ne pas avoir de bug*

## 4 - Implémentation d’un proxy inverse (*Traefik*)

*Expliquez l’intégration de Traefik : Comment avez-vous configuré les règles de redirection pour chaque service ? Comment avez-vous implémenté le chiffrement SSL/TLS pour mettre en place le protocole HTTPS ?*

## 5 - Nom de domaine

*Remplissez les tableaux suivants pour les détails du nom de domaine :*

|  |  |
| --- | --- |
| ***Fournisseur de nom de domaine utilisé*** | DuckDNS |

|  |  |
| --- | --- |
| ***Service*** | ***Adresse complète*** |
| Traefik | tpfinalservice.com |
| Jellyfin | jellyfin.tpfinalservice.com |
| Portainer | portainer.tpfinalservice.com |
| Nextcloud | nextcloud.tpfinalservice.com |

## 6 - Haute disponibilité et résilience

*Décrivez les mécanismes mis en place pour assurer la disponibilité : Utilisation de plusieurs zones de disponibilité (si applicable), redémarrage automatique des conteneurs ou autre.*

*Les conteneurs redémarre automatiquement s’il y a un crash.*

*Si j’ai un problème avec les composants réseau en un clique je peux tout refaire.*

## 7 - Automation avec Terraform (*optionnel*)

*Si vous avez utilisé Terraform :*

* *Expliquez comment vous avez structuré vos fichiers .tf*
* *Montrez des extraits pertinents ou donnez un lien vers le dépôt de votre projet.*
* *Décrivez les avantages de l’IaC dans ce contexte (reproductibilité, versionnement, simplicité de déploiement).*

*Dossier Terraform :[Terraform](https://github.com/yassineeljal/tp-final/tree/main/tpFinal-terraform)*

*J’ai un seul fichier .tf qui s’appelle main.tf et qui configure toutes les composantes dont j’ai besoin.*

*Si j’ai besoin de créer des milliers de VPC ou n’importe quelle composant réseau, je n’ai pas besoin de tout faire manuellement. Je dois créer juste un script et c’est généralement du copier-coller avec des petites modifications.*

## 8 - Problèmes rencontrés et solutions apportées

*Listez les principaux obstacles rencontrés (techniques, de configuration, etc.). Pour chacun :*

* *Description du problème*
* *Démarche de résolution*
* *Outils ou ressources consultés*

*J’ai d’abord créé une EC2 mais j’ai utilisé t2.micro donc sa boguait beaucoup. J’ai donc modifié cette instance par une large. Ensuite j’ai remarqué que mon instance n’avait pas assez de stockage et je ne savais pas comment changer sa donc j’ais regarder une vidéo YouTube et j’ai finalement mis le volume size et le type*

*Outils : Conseil du prof et Youtube*

## 9 - Conclusion

*Faites un retour global sur l’expérience du projet et du cours dans son ensemble :*

* *Ce que vous avez appris*
* *Ce que vous feriez différemment*
* *L’intérêt du projet dans une optique professionnelle / DevOps / Cloud*

*J’ai appris 3 grandes choses. L’utilisation de composant réseau qui peut servir vraiment à tout comme le déploiement de site web. Docker qui est vraiment indispensable pour ne pas se tromper sur les différentes versions de dépendance, sa facilite surtout le bon déroulement entre la personne qui fait le site et celui qui le réutilise pour faire autre chose (il ne pourra pas se mêler sur quelle version il doit installer). Finalement Terraform qui est très utiles pour faire des scripts qu’on peut toujours réutiliser et qui fait gagner un temps fou si on veut créer une grande infrastructure.*

## Annexes (facultatif)

*Logs, commandes, liens vers le dépôt GitHub, autres schémas, configuration complète de Traefik, fichiers Terraform, etc.*

*Lien du projet complet :* [*Tp Final*](https://github.com/yassineeljal/tp-final/tree/main/tpFinal-terraform)