TÉLÉCOM SUDPARIS

Département : Informatique et Réseaux

———————————————————————————————

RAPPORT DE TRAVAUX PRATIQUES

NET8552 – Orchestration de Services Réseau

COMPTE RENDU DU TP

Déploiement et orchestration d'une chaîne de fonctions réseau : Firewall → Load Balancer → Web Server

———————————————————————————————

Outils et technologies : Docker, Docker Compose, TOSCA, xOpera, BPMN (Camunda Modeler).

Environnement d'exécution : VM Ubuntu 20.04 LTS.

Images Docker utilisées :

- Firewall : wallarm/api-firewall
- Load Balancer : haproxytech/haproxy-alpine
- Web Server : nginx

Année universitaire : 2025 – 2026

Encadré par : Pr Walid GAALOUL

Réalisé par : .....

Groupe : .....

1. Introduction

- Présenter brièvement le contexte de l'orchestration de services réseau.
- Définir ce qu'est une **chaîne de fonctions réseau (VNF chain)**.
- Préciser les objectifs du TP :

  - Déployer une chaîne de services composée de trois VNFs (Firewall, Load Balancer, Web Server).
  - Utiliser **Docker** pour le déploiement.
  - Orchestrer les services via **TOSCA** et **xOpera**.
  - Modéliser le workflow de déploiement avec **BPMN (Camunda)**.

2. Objectifs du TP

Objectif	Description
1	Comprendre la notion de VNF et de chaîne de services
2	Déployer les VNFs sous forme de conteneurs Docker
3	Décrire les interconnexions avec TOSCA
4	Orchestrer le déploiement avec xOpera
5	Modéliser et exécuter le workflow BPMN avec Camunda
6	Vérifier le flux réseau complet

3. Environnement de travail

- **Système d'exploitation :** Ubuntu 20.04 LTS
- **Outils installés :**
  - Docker & Docker Compose
  - Python 3 + xOpera
  - Camunda Modeler
  - (Optionnel : Ansible)
- **Images Docker utilisées :**
  - Firewall : `wallarm/api-firewall`
  - Load Balancer : `haproxytech/haproxy-alpine`
  - Web Server : `nginx`

4. Étapes de réalisation

Décrire les étapes :

4.1 Déploiement conteneurs Docker

- Insérer ton fichier `docker-compose.yml`.

- Expliquer brièvement le rôle de chaque service (Firewall, LB, WebServer).

- Capturer les conteneurs en exécution :

```
docker ps
```

(Insère la capture d'écran ici)

4.2 Modélisation TOSCA

- Inclure le fichier `service-chain.yaml` (copie du code ou capture d'écran).
- Expliquer les dépendances entre VNFs et comment xOpera les interprète.
- Capture du graphe TOSCA (optionnel).

4.3 Orchestration avec xOpera

- Commande de déploiement :
- `opera deploy service-chain.yaml`
- Résultats observés (logs, messages de succès).
- Vérification que les 3 conteneurs sont bien déployés (`docker ps`).

(Insère une capture du terminal ici).

- Commande de nettoyage (`opera undeploy`)

(Insère une capture du terminal ici).

4.4 Workflow BPMN (Camunda)

- Insérer une capture d'écran du fichier `workflow.bpmn` (Camunda Modeler).

- Expliquer brièvement le rôle de chaque tâche :

  - Deploy Firewall
  - Deploy Load Balancer
  - Deploy Web Server
  - Test Network Flow
- Explication du fonctionnement du moteur BPMN

- Mentionner le lien avec les scripts shell/Ansible.

4.5 Vérification du flux réseau

- Montre la sortie de :

```
curl http://localhost:8080

curl http://localhost:9090

curl http://localhost:8081.
```

- Logs de conteneurs :

```
docker logs vnf-firewall
docker logs vnf-lb
docker logs vnf-web
```

- Interprète les résultats (si le flux traverse bien les VNFs).

- Ajoute les captures des réponses HTTP 200 OK.

5. Résultats et observations

Présenter les résultats obtenus pour chaque étape sous forme de tableau et commenter les écarts éventuels.

Étape	Résultat attendu	Résultat obtenu	Commentaire
Déploiement Docker	3 conteneurs actifs		
Orchestration TOSCA	Déploiement automatique		
Workflow BPMN	Exécution séquentielle correcte		
Test réseau	Flux traversant les 3 VNFs		

6. Analyse et discussion

- Quelle est la différence entre **orchestration TOSCA** et **workflow BPMN** ?

- Pourquoi séparer la logique de déploiement (xOpera) de la logique métier (Camunda) ?

- Quels sont les avantages et limites de cette approche ?

7. Conclusion

- Résumer le travail accompli.

- **Synthétiser** les principales compétences acquises (Docker, TOSCA, BPMN, orchestration).

- Proposer des pistes d'amélioration (ex. ajout d'un IDS, monitoring, automatisation CI/CD, etc).

8. Annexes

- Fichiers de configuration :
  - `docker-compose.yml`
  - `service-chain.yaml`
  - `network-service-chain.bpmn`
  - Scripts `*.sh ou playbooks *.yaml`
- Captures d'écran d'exécution (conteneurs, workflows, tests).
- Dépôt GitHub exigé : (**URLici**)

```
/network-orchestration/
├── docker-compose.yml
├── service-chain.yaml
├── network-service-chain.bpmn
├── scripts/
│   ├── deploy_firewall.sh
│   ├── deploy_loadbalancer.sh
│   ├── deploy_webserver.sh
│   └── test_flow.sh
├── playbooks/
│   ├── deploy_firewall.yaml
│   ├── deploy_loadbalancer.yaml
│   └── deploy_webserver.yaml
└── screenshots/
    ├── docker_ps.png
    ├── curl_tests.png
    └── workflow_bpmn.png
```

9. Références / Ressources

- https://github.com/xlab-si/xopera-opera
- https://camunda.com/download/modeler/
- https://hub.docker.com/r/wallarm/api-firewall
- https://hub.docker.com/r/haproxytech/haproxy-alpine
- https://hub.docker.com/_/nginx
- https://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.3/