**Travail Synthèse**

**Par Raphael Nadeau**

**2054198**

Équipe 3 3

# Étape 1 – Déployer le serveur Proxmox (Documentation de l’équipe Gamma)

Sur le serveur Gamma correspondant à l’adresse 192.168.139.199

Pour commencer, nous avons dû installer de nouveau disque dur pour notre serveur physique puisque aucun disque ne permettait au serveur de démarrer. Après le démarrage, nous avons dû vérifier l’état des cœurs de processeur pour ainsi appréhender le nombre de cœur de processeur que nous allons attribuer à nos futur VM. Lors de notre analyse nous avons remarqué que notre serveur avait :

* 10 cœurs de processeur
* 20 cœur virtuel de processeur (thread)
* 32 Go de Ram
* 3 To d’espace mémoire

Pour la pérennité des données sur le serveur sur les différents disques de notre serveur nous avons décidé de mettre la structure en RAID 5. Ainsi, nous ne perdrons pas de données lorsqu’un et même potentiellement plusieurs disques durs lâcherons.

Images des paramètres de Proxmox
Ensuite, nous avons télécharger un fichier iso de Proxmox, un hypervisor open-source de premier niveau. Grace à celui-ci nous pourrons monitorer et créer des VMs à partir d’un dashboard. Les paramètres que nous lui avons mis sont :

Pour se connecter au serveur, il faut aller sur le web et aller à l’adresse web :

192.168.139.199 :8006

# Étape 2 – Création des Vm’s(Documentation de l’équipe 2)

## Étape 2.1 - Creation de 4 Vm (1 master, 3 worker)

1. Création de la E2-VM1(Master node) et E2-VM2(Worker node) sur le serveur Promox 03.

E2-VM1 possède :

* 250gb d’espace de stockage
* 3 CPU
* 6 gigabits de ram
* 192.168.139.211

E2-VM2 Possède :

* 200gb d’espace de stockage
* 2 CPU
* 4 gigabits de ram
* 192.168.139.212

1. Création de la E2-VM3(Worker Node) sur le serveur Promox 02

E2-VM3 Possède :

* 150gb d’espace de stockage
* 2 CPU
* 4 gigabits de ram
* 192.168.139.213

1. Création de la E2-VM4(Worker Node) sur le serveur Promox 01

E2-VM4 Possède :

* 150gb d’espace de stockage
* 2 CPU
* 4 gigabits de ram
* 192.168.139.214

## Étape 2.2 – Commande pour les vm

### Étape 2.2.1 Commande à faire pour le Master Node

* sudo kubeadm init --control-plane-endpoint=192.168.139.211
* mkdir -p $HOME/.kube
* sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
* sudo chown $(id -u):$(id -g) $HOME/.kube/config
* kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.25.0/manifests/calico.yaml

### Étape 2.2.2 Commande a faire sur chaque Worker Node plus Master node

* sudo swapoff -a && sudo sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab
* sudo tee /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF

sudo modprobe overlay && sudo modprobe br\_netfilter

* sudo tee /etc/sysctl.d/kubernetes.conf <<EOF

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

net.ipv4.ip\_forward = 1

EOF

sudo sysctl –system

* sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates
* sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmour -o /etc/apt/trusted.gpg.d/docker.gpg
* sudo add-apt-repository --yes "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"
* sudo apt update && sudo apt install -y containerd.io
* containerd config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1
* sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml
* sudo systemctl restart containerd && sudo systemctl enable containerd
* curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo gpg --dearmour -o /etc/apt/trusted.gpg.d/kubernetes-xenial.gpg
* sudo apt-add-repository --yes "deb http://apt.kubernetes.io/ kubernetes-xenial main"
* sudo apt update && sudo apt install -y kubelet kubeadm kubectl && sudo apt-mark hold kubelet kubeadm kubectl
* kubeadm join 192.168.139.211:6443 --token broykg.90yjjx2fx024kapw --discovery-token-ca-cert-hash sha256:c70a2094b83aef5bb28e18eb7d7bc7a46acf54949271fde98d620237267b09f6

## Étape 2.3 – Installation d’un loadbalancer metalb

* kubectl apply -f <https://raw.githubusercontent.com/metallb/metallb/v0.13.9/config/manifests/metallb-native.yaml>
* nano metallb-config.yaml
* Appliquer le manifest suivant :

apiVersion: metallb.io/v1beta1

kind: IPAddressPool

metadata:

name: first-pool

namespace: metallb-system

spec:

addresses:

- 192.168.139.135-192.168.139.136

---

apiVersion: metallb.io/v1beta1

kind: L2Advertisement

metadata:

name: example

namespace: metallb-system

* kubectl apply -f metallb-config.yaml

## Étape 2.4 – Installation d’un Ingress Controller

* kubectl apply -f https://raw.githubusercontent.com/kubernetes/ingress-nginx/controller-v1.7.0/deploy/static/provider/cloud/deploy.yaml

## Étape 2.5 – Commande Pour installer Rancher

### Étape 2.4.1 Installer docker

* sudo apt install docker-ce -y

### Étape 2.4.2 Installer rancher

* sudo docker run --privileged -d --restart=unless-stopped -p 80:80 -p 443:443 rancher/rancher
* Suite a l’installation , se connecter sur le UI de rancher, en tapant l’adresse du master node sur le port 80 : 192.168.139.211 :80
* Afin de retrouver l’id du container : sudo docker ps
* Retrouver le container de rancher et retrouver le mot de passe : docker logs bf60ee2f07ae 2>&1 | grep "Bootstrap Password:"
* Entrer le mot de passe temporaire générer dans le UI de rancher : t4lrr2h4drmxlnwchtptgdkhgzgzw8pfkqlxgkpzhxgm8lb76bsw5q
* Changer le mot passe pour celui choisi : !abc\_def\_123\_456!

## Étape 2.6 – Installation de Longhorn

* Application du manifest de longhorn : kubectl apply -f <https://raw.githubusercontent.com/longhorn/longhorn/v1.4.1/deploy/longhorn.yaml>
* Regarder si le déploiement a bien fonctionner : kubectl get pods \  
  --namespace longhorn-system \  
  --watch

## Étape 2.7 – Import le cluster dans rancher, pour pouvoir monitorer le cluster

* Nom du cluster : clustere2
* curl --insecure -sfL <https://192.168.139.211/v3/import/bhwfd6jtk6w79lbcqd8vp6667b8896vx4hqzlxjpbclwfnmkz5xzwt_c-m-ptwhzl5t.yaml> | kubectl apply -f –

# Étape 3 – Création du manifeste

## Étape 3.1 - Dockerfile

* Création d’un fichier docker afin de créer un image partant de php:8.0.3-apache-buster et copier le contenue du dossier sans les fichier index.php.PV-NFS\_CLOUD, Dockerfile et README.md renseigner dans le .dockerignore

## Étape – 3.2 Manifeste (2054198-tp03.yaml)

### Étape 3.2.1- Serveur web / php

* Création du déploiement des 10 pods avec l’image personnaliser préalablement pousser sur dockerhub. (Aucun problème rencontrer)
* Création du service des serveurs web php-service afin de pouvoir atteindre les pods sur le port 80 (Aucun problème rencontrer)
* Création de la règles ingress(nom : ingress-php ) afin de pouvoir rediriger sur les pods servant le seveur php, host : accueil.2054198.tp03 (Aucun problème rencontrer)
* Modification du fichiers hosts pour rediriger accueil.2054198.tp03 vers l’IP du reverse Proxy 192.68.139.215 (Problème, parce que j’ai mal écrit accueil dans le fichier host )
* Création du PV PVC liée a un serveur AWS externe 3.140.197.109, path : /var/nfs/tp03 (nom : pv-nfs et pvc-nfs)
  + Difficulté avec les règles entrantes du serveur AWS
* Monter le claim du volume persistant au pods php avec le mountPath: /var/www/html/index.php et un subPath : index.php afin de juste changer le fichier index.php provenant du serveur AWS sur les pods.
  + Difficulté rencontrer avec les subpath 🡪 solution est venu de professeur.

### Étape 3.2.2 - Serveur AWS

* sudo apt update
* sudo apt install nfs-kernel-server
* sudo mkdir /var/nfs/tp03 -p
* sudo chown nobody:nogroup /var/nfs/tp03
* sudo nano /etc/exports

/var/nfs/tp03\*(rw,sync,no\_subtree\_check,all\_squash,anonuid=65534,anongid=65534,insecure)

* sudo systemctl restart nfs-kernel-server
* Ouvrir les ports suivant UDP (2049 et 111) et TCP(2049 et 111)

### Étape 3.2.3 – MATTERMOST

* Création du déploiement de mattermost en 1 réplicas avec l’image mattermost/mattermost-preview ouvert sur le port 8065
* Création du service de mattermost-service qui permet d’atteindre le port 8065 interne et de le rediriger sur le port externe 80
* Modification de la règles ingress pour ajouter l’host de mattermost : mattermost.2054198.tp03
  + Difficulté rencontrer : Je croyais qu’il fallait recréer un ingress , mais avec la documentation du cours, j’ai comprit qu’il fallait simplement l’ajouter au ingress déjà présent.
* Ajout au fichier hosts sur la même adresse IP 192.168.139.215

### Étape 3.2.4 - MariaDB

* Création du déploiement en 1 réplicas avec l’image mariadb ouvert sur le port 3306
* Création du service de mariadb-service qui permet d’atteindre le port 3306 interne et de le rediriger sur le port externe 3306
* Création d’un secret mariadb-secret permettant de renseigner les variable d’environnement en base 64 suivante a mariadb :
  + MYSQL\_ROOT\_PASSWORD: c2VjcmV0 // secret
  + MYSQL\_USER: bmV4dGNsb3Vk // nextcloud
  + MYSQL\_PASSWORD: c2VjcmV0 // secret
  + MYSQL\_DATABASE: bmV4dGNsb3Vk // nextcloud

### Étape 3.2.5 – Nextcloud

* Création du déploiement en 1 réplicas avec l’image nextcloud ouvert sur le port 80
* Ajour des variables d’environnements suivantes :
  + name: NEXTCLOUD\_TRUSTED\_DOMAINS
  + value: 192.168.139.215 nextcloud.2054198.tp03 \*.\*.\*.\*
  + name: MYSQL\_HOST
  + value: mariadb-service
* Ajout d’un secret nextcloud-secret permettant de renseigner au déploiement les variable d’environnement suivante :
  + NEXTCLOUD\_ADMIN\_USER: YWRtaW4= # admin
  + NEXTCLOUD\_ADMIN\_PASSWORD: YWRtaW4= # admin
  + MYSQL\_USER: bmV4dGNsb3Vk # nextcloud
  + MYSQL\_PASSWORD: c2VjcmV0 # secret
  + MYSQL\_DATABASE: bmV4dGNsb3Vk # nextcloud
* Création d’un volume Longhorn sur le UI de Longhorn et création du pc pvc qui lui sont relié
* Liée le volume longhorn a nextcloud avec un VolumeMounts sur le path : /var/www/html/data et un volumes liée au PVC de longhorn persistentVolumeClaim: claimName: 2054198-nextcloud
* Création du service de nextcloud-service qui permet d’atteindre le port 80 interne et de le rediriger sur le port externe 80
* Modification de la règles ingress afin d’ajouter l’host nextcloud.2054198.tp03
* Difficulté rencontrer avec nextcloud :
  + Serveur ne répondant pas : changé le sélecteur du service de nextcloud a app et non mon-app