Université de Lille

POLYTECH LILLE

Rapport Stage

Sujet: Orchestration de machines virtuelles

Réalisé par:
Souleymane Sow

Encadrant:
Mr X. Redon

Année Universitaire: **2020 - 2021**

Durée: **17 Mai - 25 Juin**

Sommaire:

Introduction:

La virtualisation consiste à créer plusieurs machines virtuelles (aussi appelées ordinateurs virtuels, instances virtuelles, versions virtuelles, VM ou virtual machine) à partir d'une machine physique, à l'aide d'un logiciel appelé hyperviseur. Parce que ces machines virtuelles fonctionnent de la même manière que des machines physiques, mais ne s'appuient sur les ressources que d'un seul système informatique, la virtualisation permet aux services informatiques des entreprises d'exécuter plusieurs systèmes d'exploitation sur un seul serveur (connu également sous le nom d'hôte). Pendant ce temps, l'hyperviseur attribue des ressources informatiques à chaque ordinateur virtuel, en fonction des besoins. Les opérations informatiques deviennent ainsi beaucoup plus efficaces et rentables. L'attribution flexible des ressources physiques est l'une des raisons pour lesquelles la virtualisation est le fondement du cloud computing. L'orchestration correspond à la configuration, la gestion et la coordination automatisées des systèmes informatiques, applications et services. L'orchestration facilite la gestion des tâches complexes pour les services informatiques. Ces derniers sont en charge de nombreux serveurs et applications, dont la gestion manuelle limite les possibilités d'évolution.

C'est dans ce cadre que s'inscrit notre projet dont le but est de mettre en place une infrastructure permettant à des utilisateurs de gérer un ensemble de machines virtuelles qui seront stockées sur un système de stockage distribué.

Plusieurs composants seront nécessaires pour mettre en place une infrastructure d'orchestration de machines virtuelles: un système de fichiers distribué, un système de virtualisation et un orchestrateur de machines virtuelles.

Le démonstrateur devra utiliser des disques sur deux serveurs physiques et permettre de faire tourner des machines virtuelles sur ces deux serveurs. On essayera de mettre en place les mécanismes de haute disponibilité et de migrations sur notre infrastructure.

I. Préparation du projet: Choix techniques

1. Logiciels:

Pour le système de fichiers on va regarder du côté de CEPH. Pour l'orchestrateur, ProxMox semble incontournable. Pour le système de virtualisation on utilisera kvm.

- **CEPH**: Solution libre de stockage distribué (software-defined storage) très populaire qui propose trois protocoles en un avec : Bloc, Fichiers & Objet (S3). Les objectifs principaux de Ceph sont d'être complètement distribués sans point unique de défaillance, extensible jusqu'à l'exaoctet et librement disponible. Les données sont répliquées, permettant au système d'être tolérant aux pannes.
- **Proxmox**: Solution de virtualisation libre basée sur l'hyperviseur Linux KVM et offrant aussi une solution de containers avec LXC
- **KVM**: Hyperviseur libre de type I pour Linux. KVM est intégré dans le noyau Linux depuis la version 2.6.20. Il fonctionne originellement sur les processeurs à architectures x86 disposant des instructions de virtualisation Intel VT ou AMD-V.

2. Matériels:

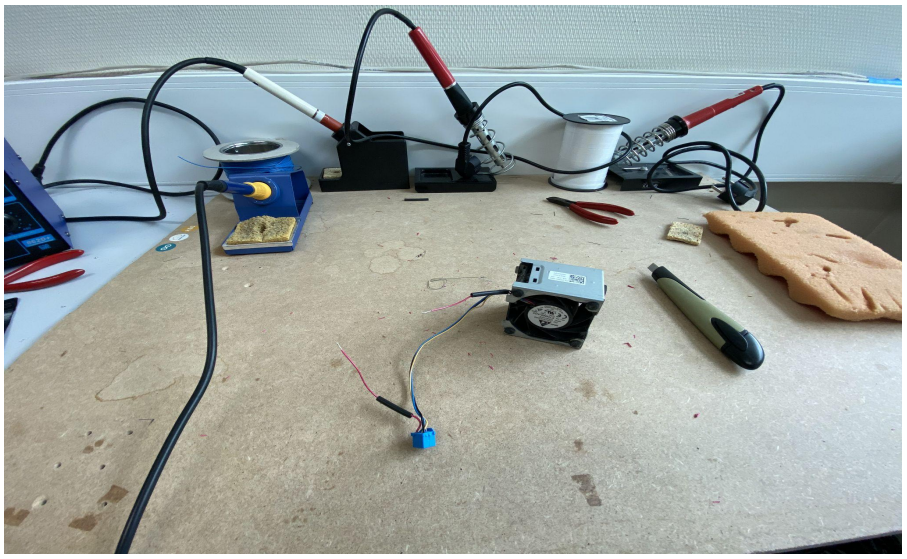
Pour le matériel, on aura à utiliser 2 serveurs physiques de la marque Dell.
Le premier est un PowerEdge 2950 bi-processeur avec 4Go de Ram disponible et 3 disques de stockage; un de 70Go et 2 de 300 Go. Il est dans un état fonctionnel, il ne reste qu'à installer un nouveau système d'exploitation dessus.
Le second est un PowerEdge R520 disposant également de 2 processeurs, avec 32Go de Ram disponible et un disque de 1 To de stockage. Mais dû à un problème avec ses capteurs de températures, le système de refroidissement s'active au maximum dès le démarrage du serveur provoquant un bruit énorme.
L'interface de gestion du serveur IDRAC mis en place par le fournisseur Dell étant inaccessible, notre premier objectif sera de résoudre ce problème.

II. Réalisation du projet:

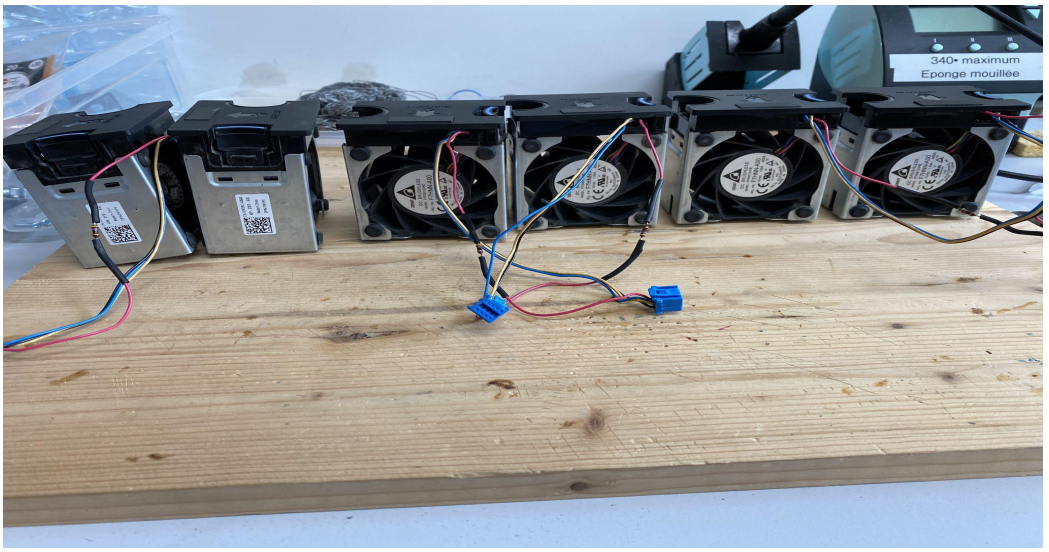
1. Semaine 1 :Remise en marche du PowerEdge R520

Après plusieurs essais de solutions software trouvées dans nos recherches pour pouvoir récupérer l'accès à IDRAC, on a pas réussi à accéder à la plateforme. On s'est tourné ainsi vers une solution hardware à savoir réduire le taux de ventilation du système.

Pour cela, il nous a fallu démonter un des ventilateurs pour tester une solution à savoir réduire la tension d'alimentation du ventilateur. Ainsi on a eu à chercher la référence du ventilateur pour retrouver la fonction de chaque fil. Après avoir retrouvé le fil d'alimentation on a pu avec le matériel de soudure disponible dans les salles, rajouter une résistance de 47 ohms et d'½ W de puissance pour plus de sécurité.



Après avoir réinstaller le ventilateur et mis en marche notre serveur, on a noté une nette différence dans le bruit . Donc on a fait la même chose pour les autres ventilateurs pour réduire leur tension d'alimentation et ainsi leur vitesse de rotation.



Après avoir appliqué la solution à tous les ventilateurs et les avoir remis en place, la ventilation du serveur a nettement baissé. Vient alors la question à savoir si la solution adoptée permet de refroidir assez les composants internes du serveur afin d'éviter une surchauffe.
Pour étudier cela on a installé d'abord un nouvel OS sur le serveur, un Devuan 3.0 sur une partition de 100Go.
Après avoir fini l'installation et les différentes configurations réseaux pour avoir accès au réseau de l'école et à Internet, on a installé des paquets à savoir

lm-sensors et **stress** qui permettent respectivement de connaître la température des cpu et des composants de notre serveur, imposer des charges à nos cpu pour qu'ils tournent à fond.
Ensuite on a mis en place un script shell qui permettait de stresser nos cpu pendant une durée de 2h sans interruption et d'afficher la température des éléments du serveur chaque 10 minutes pour qu'on puisse monitorer leur évolution.

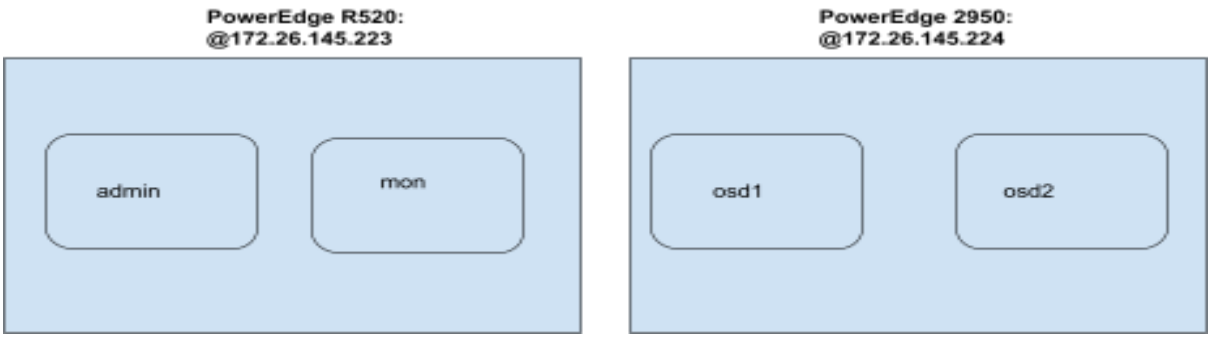
```
#!/bin/bash

stress --quiet --cpu  8 --timeout 7200 &
echo "Processus de stress des CPU lancé"
let "i=1";
let "b=12";
while (($i < $b)) ;
do
        sensors
        sleep 600
        let "i=i+1"
done
```

On a pu constater que nos températures ne dépassaient pas les 55°C loin des 97°C de seuil critique.
On en déduit que notre solution a permis de résoudre le problème de refroidissement et de nuisance sonore de notre serveur.
On peut ainsi pouvoir démarrer le projet proprement dit.

2. Semaine 2:

2.1. Mise en place du système de fichiers distribué CEPH:

Après l'installation d'un système et les configurations réseau de nos deux serveurs, on va mettre en place une infrastructure de stockage distribuée avec ceph.
Ceph est une plateforme libre de stockage distribuée. Les objectifs principaux de Ceph sont d'être complètement distribué sans point unique de défaillance, extensible jusqu'à l'exaoctet et librement disponible. Les données sont répliquées, permettant au système d'être tolérant aux pannes. Ceph fonctionne sur du matériel non spécialisé. Le système est conçu pour s'auto-réparer et automatiser au maximum ses tâches administratives afin de réduire les coûts d'exploitation.
On va essayer de mettre en place l'architecture ceph suivante:



On se place d'abord au niveau du répertoire **/etc/hosts** de chaque machine pour rajouter les machines du réseau comme dans la configuration suivante

```
172.26.145.224  osd1
172.26.145.224  osd2
172.26.145.223  admin
172.26.145.223  mon
```

Installation et configuration de CEPH:

Pour installer CEPH, nous allons avoir besoin d'un utilisateur dédié. Il va donc falloir créer un utilisateur commun sur nos 2 serveurs de notre cluster:

- **Création de l'utilisateur ceph:**

```
useradd -d /home/ceph -m ceph
passwd ceph (pasglop)
echo "ceph ALL = (root) NOPASSWD:ALL" | sudo tee /etc/sudoers.d/ceph
(Donner le droit sudo sans mot de passe pour tous les logiciels de la machine pour l'utilisateur ceph)
sudo chmod 0440 /etc/sudoers.d/ceph
```

- **Mise en place du ssh:**

Pour s'installer, CEPH utilise le protocole SSH et nécessite une authentification par paire de clé.

On va donc mettre en place une authentification par paire de clé entre notre serveur d'administration (**admin**) et les autres machines de notre cluster.

Sur **admin:** On bascule sur le user **ceph** et on crée la paire de clef

```
su - ceph
ssh keygen
```

On envoie la clé publique vers les autres machines de notre cluster

```
ssh-copy-id ceph@osd1
```

Enfin pour finir la configuration du SSH du serveur d'administration, il faut créer un fichier **config** dans **/home/ceph/.ssh**

```
Host osd1
Hostname osd1
User ceph
Host osd2
Hostname osd2
User ceph
Host mon
Hostname mon
User ceph
```

- **Configuration des dépôts:**

On va maintenant configurer les dépôts de notre serveur d'administration pour pouvoir installer la dernière version de CEPH. Les opérations suivantes sont réalisées sur le serveur d'administration uniquement en utilisateur ceph.

Mise en place des dépôts CEPH:

```
wget -q -O- 'https://download.ceph.com/keys/release.asc' | sudo apt-key add -
echo deb https://download.ceph.com/debian-giant/ jessie main | sudo tee /etc/apt/sources.list.d/ceph.list
sudo apt update
sudo apt install ceph-deploy ntp
```

On installe ntp pour éviter les dérives d'horloge dans les moniteurs ceph.

2.2. Mise en place du système de fichiers distribué CEPH(Suite):

- **Création du cluster ceph**

Nous allons maintenant installer *Ceph* sur chacune des machines pour en faire un cluster puis nous les spécialisons une à une. Sur la machine d'administration (**admin**) et en tant qu'utilisateur ***ceph***, il faut créer un dossier nommé **test-ceph** dans lequel les commandes ceph-deploy seront utilisées (ceph-deploy crée des fichiers localement là où il est exécuté):

```
mkdir /home/ceph/test-ceph
```

Un cluster Ceph ne peut fonctionner sans moniteur, aussi nous devons commencer par indiquer quelle machine aura ce rôle futur. Pour ce faire, la commande

```
ceph-deploy new mon
```

commence le déploiement d'un nouveau cluster en ajoutant une machine moniteur dans ce cluster. Elle crée également trois fichiers : un fichier de clefs, un fichier de log et un fichier de configuration.

Nous allons ensuite installer l'ensemble des programmes et librairies nécessaires à Ceph sur l'autre serveur. À la fin, nos 2 serveurs seront tous identiques et leur rôle leur sera attribué par la suite.

```
ceph-deploy install admin mon osd1 osd2
--repo-url=https://download.ceph.com/debian-jewel/
ceph-deploy install osd1 osd2
--repo-url=https://download.ceph.com/debian-15.2.4/
```

L'erreur suivante a été rencontré lors de l'installation

```
[admin][ERROR ] RuntimeError: command returned non-zero exit status: 100
    [ceph_deploy][ERROR ] RuntimeError: Failed to execute command: env
DEBIAN_FRONTEND=noninteractive DEBIAN_PRIORITY=critical apt-get
--assume-yes -q --no-install-recommends install ceph ceph-osd ceph-mds
ceph-mon radosgw
```

Ce problème a pu être résolu en installant sur nos 2 serveurs les paquets **gnupg1** et **radosgw**

- **Spécialisation de la machine virtuelle moniteur**

Maintenant que Ceph est installé sur chacune de nos machines et que l'une d'entre elles est identifiée comme une machine moniteur nous devons la spécialiser. Pour ce faire, nous créons le moniteur à l'aide de la commande suivante toujours exécutée depuis la machine d'administration:

```
ceph-deploy mon create mon
```

Lors de cette étape, on rencontre une erreur de dépendances de packages. Pour pouvoir créer un nouveau moniteur, les scripts de commande de **ceph-deploy mon create** utilisent **systemctl** or notre OS Devuan 3.0 est **systemd-free** utilisant **sysv-init.** On a pas pu trouver d'alternative pour pouvoir lancer la commande avec un système sysv-init.

3. Semaine 3:

On a pas pu trouver une alternative d'installation de Ceph sans Systemd. Lors des recherches de solutions, il s'est trouvé que Proxmox était systemd dépendant donc ne pourrait être installé sur nos deux serveurs. On a décidé de mettre en place des machines virtuelles et y installer un Debian 10 pour pouvoir mettre en place nos solutions.

3.1. Installation machine virtuelle:

- **Installation des paquets requis:**

Pour vérifier que les microprocesseurs de nos machines permettent la virtualisation avec KVM on tape la commande suivante:

```
grep -E 'vmx|svm' /proc/cpuinfo &>/dev/null && echo "La virtualisation est possible sur cette machine." || echo "Le microprocesseur de cette machine ne permet pas d'utiliser la virtualisation avec KVM."
```

On installe ensuite les paquetages qemu-kvm et libvirt-daemon:

```
apt-get install qemu-system libvirt-clients libvirt-daemon-system virtinst
```

- **QEMU** est un logiciel libre de machine virtuelle, pouvant émuler un processeur et, plus généralement, une architecture différente si besoin. Il permet d'exécuter un ou plusieurs systèmes d'exploitation via les hyperviseurs KVM et Xen, ou seulement des binaires, dans l'environnement d'un système d'exploitation déjà installé sur la machine.
- l**ibvirt** est une bibliothèque, une API, un daemon et des outils en logiciel libre de gestion de la virtualisation. Elle est notamment utilisée par KVM, Xen, VMware ESX, QEMU et d'autres solutions de virtualisation. Elle est notamment utilisée par la couche d'orchestration des hyperviseurs.

Ensuite on utilise la commande adduser pour ajouter notre nom d'utilisateur aux groupes kvm et libvirt:

```
adduser pifou kvm
adduser pifou libvirt
```

- **Installation de notre VM Proxmox:**

Avant de lancer l'installation, il nous faut au préalable avoir notre fichier d'installation sur notre machine avant de lancer la commande d'installation suivante:

```
virt-install --name proxmox --ram 4096 --disk path=/var/lib/libvirt/images/proxmox.img,size=20 --vcpus 2 --os-type linux --os-variant debian10 --network network=default --graphics none --console
```

```
pty,target_type=serial --location
/var/lib/libvirt/boot/debian-10.9.0-amd64-netinst.iso --extra-args
'console=ttyS0,115200n8 serial'
```

- **--name:** Nom de la machine
- **--ram:** spécifie la quantité de mémoire qu'utilise la machine virtuelle en mégaoctets
- **--disk path:** montre le chemin d'accès au disque virtuel qui peut être un fichier, une partition ou un volume logique. Dans cet exemple, un fichier nommé proxmox.img dans le répertoire /var/lib/libvirt/images/, avec une taille de 20 gigaoctets
- **--location:** Fichier à utiliser comme CDROM d'installation virtuel
- **--os-type:** Optimise la configuration pour un type de système d'exploitation (Linux/windows)
- **--os-variant:** Optimise davantage la configuration pour une variante de système d'exploitation spécifique(Debian 10/ Ubuntu 20)
- **--vcpus:** Nombre de processeurs virtuels à configurer pour notre vm
- **--network:** Connectez la vm au réseau de la machine hôte

A la fin de l'installation on pourra gérer notre machine avec les commandes suivantes:

- Lister les machines virtuelles

```
virsh list --all
```

- Puis, pour démarrer, arrêter ou encore redémarrer une machine, nous pourrons utiliser les commandes suivantes :

```
virsh start proxmox
virsh shutdown proxmox
virsh reboot proxmox
```

- Se connecter à la console de notre VM:

```
virsh console proxmox
```

- D'autre part, pour forcer l'arrêt de notre VM :

```
virsh destroy proxmox
```

3.2. Installation de Proxmox:

Dans cette partie on va décrire les étapes pour l'installation de Proxmox sur un système Debian 10.

- **Ajout d'une entrée dans /etc/hosts pour notre @ip:**

Le nom d'hôte de notre machine doit pouvoir être résolu avant de commencer l'installation de Proxmox sur notre Debian

```
127.0.0.1       localhost.localdomain localhost
192.168.122.10   proxmox1.plil.info proxmox1
```

On teste notre configuration avec la commande hostname --ip-address qui nous renvoie notre adresse IP si tout est ok!

- **Ajout du repos Proxmox:**

```
echo "deb http://download.proxmox.com/debian/pve buster
pve-no-subscription" > /etc/apt/sources.list.d/pve-install-repo.list
```

- **Et sa clé** :

```
wget http://download.proxmox.com/debian/proxmox-ve-release-6.x.gpg -O
/etc/apt/trusted.gpg.d/proxmox-ve-release-6.x.gpg
```

On la rend exécutable :

```
chmod +r /etc/apt/trusted.gpg.d/proxmox-ve-release-6.x.gpg
```

On met à jour nos dépôts:

```
sudo apt update && sudo apt full-upgrade
```

- **Installation des packages de Proxmox VE**

Pour commencer l'installation on lance :

```
apt install proxmox-ve postfix open-iscsi
```

A la fin de l'installation, le système reboot, et démarre automatiquement sur le kernel Proxmox. Pour accéder à la page d'administration de notre Proxmox à partir du navigateur d'une machine de TP de la salle, on va mettre en place un Reverse Proxy au niveau de nos serveurs pour faciliter l'administration. On rappelle que l'interface d'administration de Proxmox est disponible à l'adresse IP de la machine hôte (notre VM) dans le port 8006.

3.3. Mise en place d'un Reverse Proxy Apache pour l'accès à l'interface de gestion de Proxmox:

Le **reverse proxy** sera une passerelle permettant aux hôtes appartenant au réseau 172.26.145.0/24 d'accéder au réseau virtuel interne de nos serveurs Cordouan 192.168.122.0/24 et Container 192.168.123.0/24. Il nous faudra spécifier à notre machine d'administration appartenant au réseau 172.26.145.0 la route lui permettant d'accéder au réseau virtuel de nos Vms grâce à la commande ip route add.

- **Installation Apache:** Dans un premier temps il nous faut installer Apache2

```
apt-get install apache2
```

- **Activation des modules utilisés:**

Pour utiliser Apache en mode proxy inverse, il nous faut activer les 2 modules proxy et proxy_http via la commande a2enmod et les modules ssl pour une connexion sécurisée https

```
a2enmod ssl proxy rewrite proxy_connect proxy_http
```

Le module **proxy** permet l'implémentation d'un serveur mandataire / passerelle et le module **proxy_http** apporte le support des requêtes HTTP et HTTPS au module proxy. Pour activer ces services, il est nécessaire de redémarrer apache.

```
service apache2 restart
```

- **Mise en place du Virtualhost:** On modifie ensuite le fichier de configuration de notre proxy en éditant le fichier **/etc/apache2/sites-available/proxmox.polytech-lille.fr.conf**

```
<IfModule mod_ssl.c>
        <VirtualHost *:443>
            ServerName proxmox.polytech-lille.fr
            ProxyPreserveHost On
            ProxyRequests Off
            ProxyErrorOverride Off
            ProxyPass       / https://192.168.122.10:8006/
            ProxyPassReverse    / https://192.168.122.10:8006/===Mise
en place d'un Reverse Proxy Apache pour l'accès à l'interface de
gestion de Proxmox===
            SSLEngine on
            SSLCertificateFile    /etc/ssl/server.crt
```

```
        SSLCertificateKeyFile    /etc/ssl/server.key
        SSLProxyEngine On
        SSLProxyCheckPeerCN off
        SSLProxyCheckPeerName off
        SSLProxyCheckPeerExpire off
        SSLProxyVerify none
    </VirtualHost>
</IfModule>
```

ServerName permet de spécifier le nom de domaine voulu;**ProxyPass** et **ProxyPassReverse** donnent la correspondance entre le path voulu et l'adresse du serveur destinataire et ajustent l'URL dans les en-têtes HTTP. **ProxyRequests** permet d'activer et de désactiver la fonctionnalité de serveur mandataire. Pour la configuration du SSL, il nous faut créer un certificat SSL.

- **Création d'un certificat SSL:**

Prérequis: Installation du paquet openssl:

```
apt-get install openssl
```

Création de la clef: On se place au niveau du répertoire /etc/ssl/ avant de créer la clef

```
sudo openssl genrsa -out server.key 2048
```

Cette commande va créer la clé privée avec l'algorithme RSA 2048 bits.

Demande de signature du certificat: Ensuite il faut générer un fichier de « demande de signature de certificat », en anglais CSR : Certificate Signing Request

```
sudo openssl req -new -key server.key -out server.csr
```

On répond à un certain nombre de questions. On veille surtout à mettre le nom du serveur tel qu'il est appelé de l'extérieur dans le champ « **Common Name** » (ici : "**proxmox.polytech-lille.fr**").

Pour visualiser le contenu du fichier généré:

```
sudo openssl req -text -noout -in server.csr
```

Signature du certificat: Enfin, on génère le certificat signé au format x509:
(certificat auto signé pour 365 jours)

```
sudo openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

Installation du certificat:

```
cp /etc/ssl/server.crt /etc/ssl/certs/
cp /etc/ssl/server.key /etc/ssl/private/
```

Maintenant que notre certificat SSL est généré et notre fichier de configuration virtualhost défini ,nous devons l'activer puis recharger Apache

```
a2ensite proxmox.polytech-lille.fr.conf
service apache2 reload
```

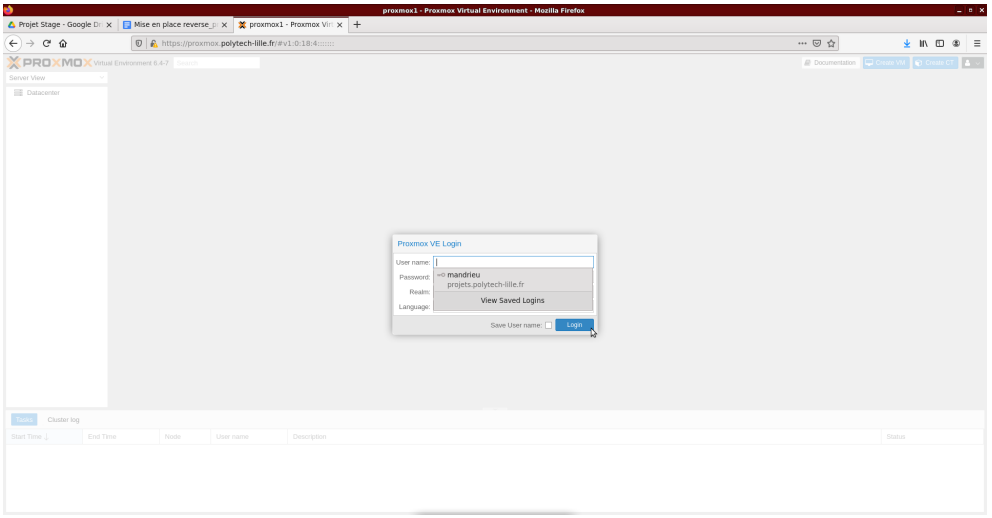
On met aussi en place une **masquerade** sur nos 2 serveurs pour permettre à nos VMs d'accéder au réseau externe en utilisant l'adresse IP de leur serveur hôte sur le réseau 172.26.145.0/24.

```
iptables -t nat -A POSTROUTING -j MASQUERADE -s 192.168.122.10/32 (Sur Cordouan)
    iptables -t nat -A POSTROUTING -j MASQUERADE -s 192.168.123.10/32 (Sur Container)
```

Pour tester si notre configuration fonctionne à partir d'un navigateur de notre machine d'administration, après avoir désactivé les proxy pour les réseaux de nos machines proxmox et rajouté le nom de nos hôtes dans le fichier /etc/hosts comme suit:

```
172.26.145.223 cordouan proxmox.polytech-lille.fr
172.26.145.224 container proxmox2.polytech-lille.fr
```

On arrive à accéder aux interfaces d'administration de nos proxmox via les adresses proxmox.polytech-lille.fr et proxmox2.polytech-lille.fr



Interface de gestion du Proxmox

3.4. Mise en place de notre cluster Proxmox:

On va mettre en place un cluster de 2 noeuds:

Cordouan----> Name: proxmox1
@IP: 172.26.145.223
Container----> Name: proxmox2
@IP: 172.26.145.224

Vu que nos proxmox sont installés sur des machines virtuelles qui n'ont pas accès à l'espace de stockage des serveurs hôtes, on doit rajouter de l'espace de stockage sur nos Vms. Pour cela on va utiliser la commande:

Sur Cordouan:
virsh attach-disk proxmox1 /dev/sda3 vdb
Sur Container:
virsh attach-disk proxmox2 /dev/sdb vdb
virsh attach-disk proxmox2 /dev/sdc vdc

- **Création du cluster:** On va créer le cluster sur Promox1

```
pvecm create mycluster
```

- **Vérification du statut du cluster:**

```
pvecm status
```

```
[root@proxmox1:~]# pvecm status
Cluster information
-------------------
Name:             mycluster
Config Version:   1
Transport:        knet
Secure auth:      on

Quorum information
------------------
Date:             Fri Jun  4 16:16:59 2021
Quorum provider:  corosync_votequorum
Nodes:            1
Node ID:          0x00000001
Ring ID:          1.5
Quorate:          Yes

Votequorum information
----------------------
Expected votes:   1
Highest expected: 1
Total votes:      1
Quorum:           1
Flags:            Quorate

Membership information
----------------------
    Nodeid      Votes Name
0x00000001          1 192.168.122.10 (local)
[root@proxmox1:~]#
```

Statut du cluster:

- **Adhésion au cluster de proxmox2:** On va demander à proxmox2 de rejoindre le cluster **mycluster**

pvecm add 172.26.145.223

On peut vérifier sur l'interface d'administration de Proxmox si notre cluster a été créé et que nos noeuds ont été ajouté avec succès

Notre cluster Proxmox

- **Éviter la problématique du Quorum:** Très important pour la suite, nous allons effectuer une manipulation pour éviter de se retrouver avec un hyperviseur en "lecture seule" dans le cas ou le second nœud viendrait à rencontrer un problème (réseau, matériel ...). Afin de maintenir la synchronisation entre les nœuds, une exigence Proxmox est qu'au moins trois nœuds doivent être ajoutés au cluster. Dans notre architecture, cela n'est pas faisable. Dans les configurations à deux nœuds, ce qui se passe, c'est que les deux nœuds doivent toujours être opérationnels pour que toute modification, telle que le démarrage, l'arrêt ou la création de machines

virtuelles et de conteneurs, soit effectuée. Afin de résoudre ce problème, nous pouvons utiliser un QDevice externe dont le seul but est de régler les votes pendant les périodes de panne de nœud. Ce QDevice ne sera pas un composant visible du cluster Proxmox et ne pourra pas y exécuter de machine virtuelle ou de conteneur.

On installe le paquet **corsync-qdevice** sur nos deux noeuds:

```
apt install corosync-qdevice
```

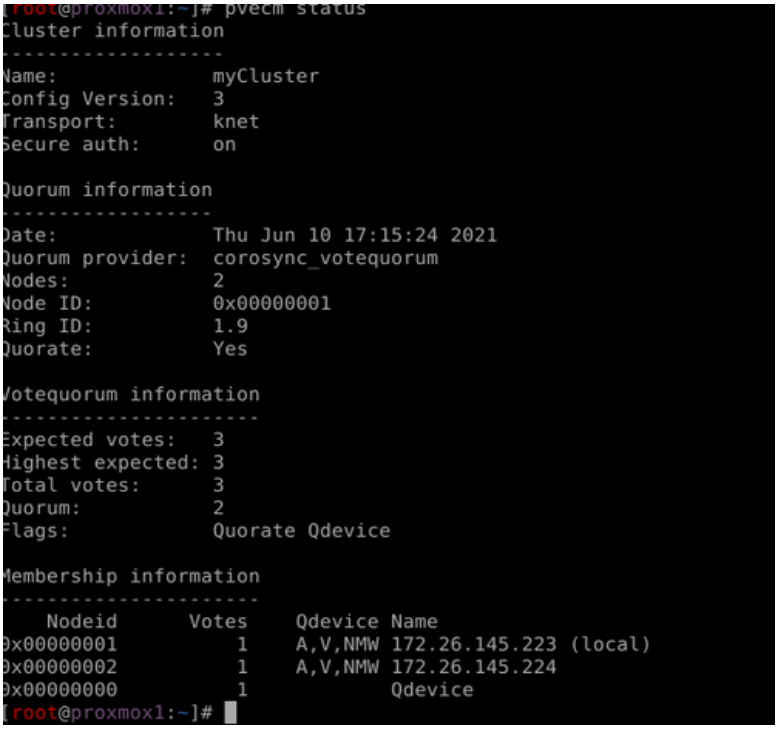
Ensuite sur notre machine d'administration qui va jouer le rôle du Qdevice on installe les paquets suivants:

```
apt install corosync-qnetd corosync-qdevice
```

Enfin à partir d'un noeud, on joint notre Qdevice à notre cluster via la commande (@IP Qdevice:172.26.145.70)

```
pvecm qdevice setup 172.26.145.70 -f
```

Pour vérifier si notre configuration a bien été prise en compte, on regarde l'état de notre cluster via la commande pvecm status, on observe bien que le nombre de votes est passé de 2 à 3 et que notre Qdevice a rejoint le cluster.



Notre cluster Proxmox avec 3 votes grâce au Qdevice

4. Semaine 4:

4.1. Mise en place d'une solution de stockage distribué avec Proxmox et Ceph:

Ceph c'est quoi? Ceph est une plateforme libre de stockage distribuée. Les objectifs principaux de Ceph sont d'être complètement distribué sans point unique de défaillance, extensible jusqu'à l'exaoctet et librement disponible. Les données sont répliquées, permettant au système d'être tolérant aux pannes. Ceph fonctionne sur du matériel non spécialisé. Le système est conçu pour s'auto-réparer et automatiser au maximum ses tâches administratives afin de réduire les coûts d'exploitation.

Pour faire simple, Ceph nous permet d'utiliser les disques de plusieurs hôtes pour réaliser du stockage distribué, permettant par exemple de réaliser des migrations de machines virtuelles à chaud, ou encore d'éviter les pannes étant donné qu'il n'y aura pas de SPOF, c'est-à-dire de « Single Point of Failure », pas de point unique de défaillance. Dans le cas des VMs toujours, si nous les stockons sur notre cluster Ceph, celles-ci seront donc répliquées sur chacun des hôtes.

Installation de Ceph: Alternativement à l'assistant d'installation recommandé de Proxmox VE Ceph disponible dans l'interface Web, nous pouvons utiliser la commande suivante sur chaque nœud de notre cluster :

```
pveceph install
```

Cela configure un référentiel de packages apt dans **/etc/apt/sources.list.d/ceph.list** et installe le logiciel requis.

Configuration initiale de Ceph: On exécute la commande suivante sur un des noeuds

```
pveceph init --network 192.168.122.0/24
```

Cela crée une configuration initiale dans **/etc/pve/ceph.conf** avec un réseau dédié pour Ceph. Ce fichier est automatiquement distribué à tous les nœuds Proxmox VE de notre cluster, à l'aide de **pmxcfs**. La commande crée également un lien symbolique dans **/etc/ceph/ceph.conf,** qui pointe vers ce fichier. Ainsi, nous pouvons simplement exécuter les commandes Ceph sans avoir besoin de spécifier un fichier de configuration.

Ceph monitor: Le moniteur Ceph conserve une copie principale de la carte de cluster. Un minimum de trois monitor nodes est recommandé pour garantir la

fiabilité. Mais dans notre architecture, on n'a que 2 nœuds disponibles dans notre cluster.

- **Création de moniteurs:** Sur chaque nœud où l'on souhaite placer un moniteur, on exécute la commande

```
pveceph mon create
```

- **Suppression de moniteurs:** Pour supprimer un moniteur Ceph, on se connecte d'abord au nœud sur lequel le MON est en cours d'exécution. Ensuite on exécute la commande suivante

```
pveceph mon destroy
```

Ceph manager: Ces nodes gèrent l'état de l'utilisation de la mémoire, de la charge du système et de l'exploitation des nœuds.

- **Création de manager:**

```
pveceph mgr create
```

- **Suppression de manager:**

```
pveceph mgr destroy
```

Ceph OSDs: Physiquement, les données sont stockées sur des disques ou SSD formatés avec un système de fichiers comme ext et que Ceph baptisé Ceph OSD (Ceph Object Storage Device). À chaque OSD correspond un démon chargé de stocker les données, de les répliquer ou de les redistribuer en cas de défaillance d'un équipement. Chaque démon OSD fournit aussi des informations de monitoring et de santé aux moniteurs Ceph. Un cluster Ceph doit à minima disposer de deux démons OSD (3 sont recommandés) pour démarrer.Il est recommandé d'utiliser un OSD par disque physique.

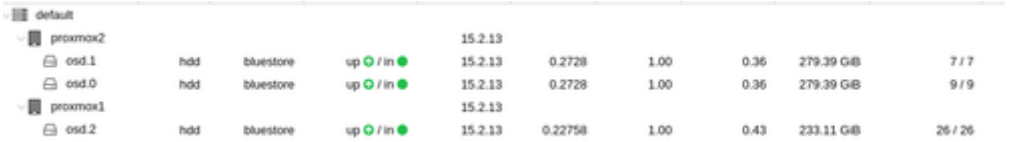
- **Création d'OSD:** Sur chaque noeud ou on veut créer un OSD on exécute la commande suivante en spécifiant le disque à utiliser comme OSD(sdb et sdc pour le proxmox2 et vdb pour le proxmox1)

```
pveceph osd create /dev/sdb
```

- **Suppression d'OSD:**

```
ceph osd out <ID>
systemctl stop ceph-osd@<ID>.service
```

La première commande indique à Ceph de ne pas inclure l'OSD dans la distribution des données. La deuxième commande arrête le service OSD. Jusqu'à ce moment, aucune donnée n'est perdue.



OSDs de notre cluster Ceph

Ceph Pool: Un cluster Ceph stocke les données sous forme d'objets stockés dans des partitions logiques baptisées "pools". À chaque pool Ceph correspond un ensemble de propriétés définissant les règles de réplications (le nombre de copie pour chaque donnée inscrite) ou le nombre de groupes de placement (placement groups) dans le pool. Les pools peuvent être répliqués ou protégés par l'utilisation de code à effacement (erasure coding). Dans un pool répliqué, le système stocke autant de copies que spécifié de chaque donnée (le coût de stockage augmente linéairement avec le nombre de copies spécifiées). Par exemple, si l'on a spécifié trois copies et que le cluster dispose de trois nœuds, la triple réplication permet de survivre à deux pannes de nœuds ou à la panne de deux disques. Lorsqu'aucune option n'est donnée, nous définissons une valeur par défaut de 128 PG, une taille de 3 répliques et une taille_min de 2 répliques, pour garantir qu'aucune perte de données ne se produise en cas d'échec d'un OSD.

- **Création du pool:** Sur un des noeuds de notre cluster on tape la commande suivante pour créer notre pool nommé **ceph-pool**

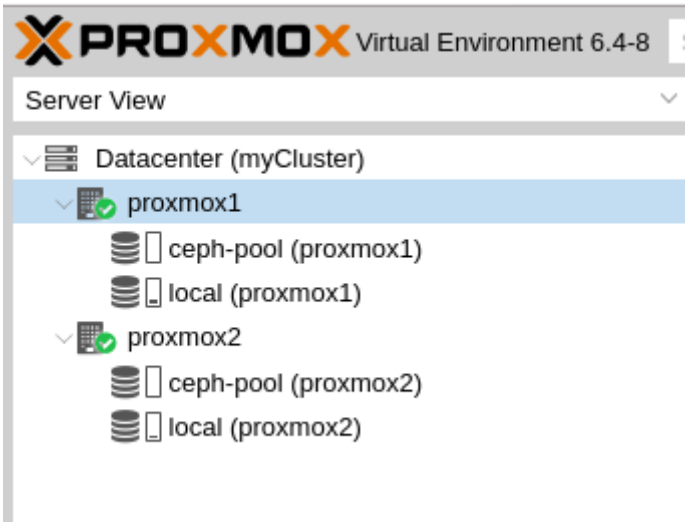
```
pveceph pool create ceph-pool --add_storages
```

- **Suppression du pool:**

```
pveceph pool destroy ceph-pool
```

- **PG autoscaler:** L'autoscaler PG permet au cluster de prendre en compte la quantité de données (attendues) stockées dans chaque pool et de choisir automatiquement les valeurs pg_num appropriées. Il est disponible depuis Ceph Nautilus.On va l'activer via la commande:

```
ceph mgr module enable pg_autoscaler
```



Pool de stockage mis en place

4.2. Création des machines virtuelles:

Sur le menu de l'interface de gestion d'un des nœuds du cluster Proxmox, on clique sur **Create VM**. Ensuite on suit les étapes en spécifiant les caractéristiques de notre VM et à choisir le fichier d'installation. On obtient ainsi la configuration suivante:

Menu après la création d'un VM:

On appuie ensuite sur le bouton **Start** pour démarrer la machine puis sur **Console** pour accéder à la console d'installation de notre VM.

On a aussi le menu **Shutdown** qui propose différentes options d'arrêts et de mise en pause de la machine. On peut aussi démarrer et arrêter les machines à partir de la console CLI d'un noeud avec la commande **qm**:

```
qm start/stop/shutdown… <VMID>
```

5. Semaine 5:

On s'est rendu compte que notre architecture avait des limites pour gérer des machines virtuelles. L'installation d'un nouveau système sur une VM nous a pris plus d'une heure. Après des tests réseau et d'écritures/lectures sur les disques on s'est

rendu compte que le souci ne venait pas de là mais plutôt du fait d'installer des machines virtuelles sur une machine virtuelle car le noeud principal de notre cluster a été installé sur une machine virtuelle dans Cordouan. Pour résoudre le problème, on va installer notre nœud directement sur Cordouan et auparavant on va devoir supprimer le cluster déjà mis en place afin de pouvoir rajouter les autres nœuds dans le prochain cluster qui sera mis en place.

5.1. Démonter notre cluster Proxmox:

Tout d'abord, on arrête les services corosync et pve-cluster sur le nœud :

```
systemctl stop pve-cluster
systemctl stop corosync
```

On redémarre le système de fichiers du cluster en mode local :

```
pmxcfs -l
```

On supprime ensuite les fichiers de configuration de corosync :

```
rm /etc/pve/corosync.conf
rm -r /etc/corosync/*
```

Nous pouvons maintenant redémarrer le système de fichiers en tant que service normal en quittant le mode local :

```
killall pmxcfs
systemctl start pve-cluster
```

Le nœud est maintenant séparé du cluster. Nous pouvons le supprimer à partir du nœud restant du cluster avec :

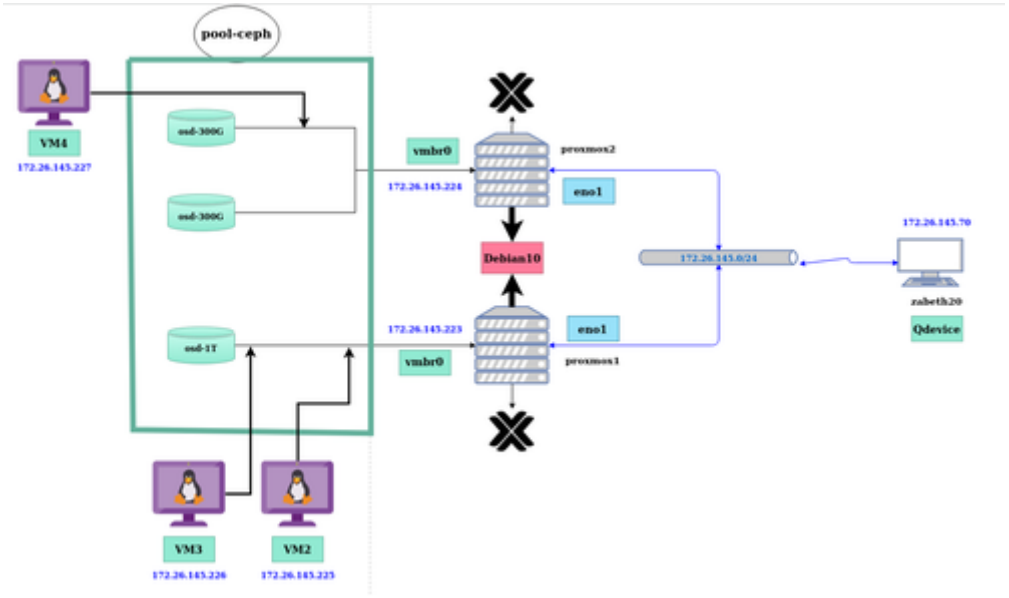
```
pvecm delnode proxmox2
```

Nous revenons maintenant au nœud séparé pour supprimer tous les fichiers restants de l'ancien cluster. Cela garantit que le nœud pourra être ajouté à nouveau à un autre cluster sans problème.

```
rm /var/lib/corosync/*
```

5.2. Mise en place de la nouvelle architecture:

On va mettre en place l'infrastructure décrite sur la figure suivante:



Nouvelle architecture à mettre en place

On va installer sur nos deux serveurs directement un système Debian qui utilise systemd afin de pouvoir mettre en place un cluster Proxmox optimal. Comme réalisé précédemment, la machine d'administration(Zabeth20) jouera le rôle du 3eme nœud en tant que Qdevice. Un disque de 1To a pu être ajouté sur le serveur Cordouan(Proxmox1) pour pouvoir avoir à disposition 3 OSDs qui est le minimum requis pour un cluster Ceph.

Le corosync-qdevice est un démon s'exécutant sur chaque nœud d'un cluster. Il fournit un nombre configuré de votes au sous-système de quorum en fonction de la décision d'un arbitre tiers. Son utilisation principale est de permettre à un cluster de supporter plus de défaillances de nœuds que ne le permettent les règles de quorum standard. Il est recommandé pour les clusters avec un nombre pair de nœuds et fortement recommandé pour les clusters à 2 nœuds comme dans notre cas.

On va ensuite paramétrer le réseau selon l'architecture, sur nos deux serveurs on mettra en place un pont virtuel **vmbr0** pour connecter nos machines virtuelles au réseau.

```
#Configuration réseau de Proxmox2 pour mettre en place un bridge pour les Vms
 auto eno1
 iface eno1 inet manual
 auto vmbr0
 iface vmbr0 inet static
      address 172.26.145.224/24
      gateway 172.26.145.254
      bridge-ports eno1
```

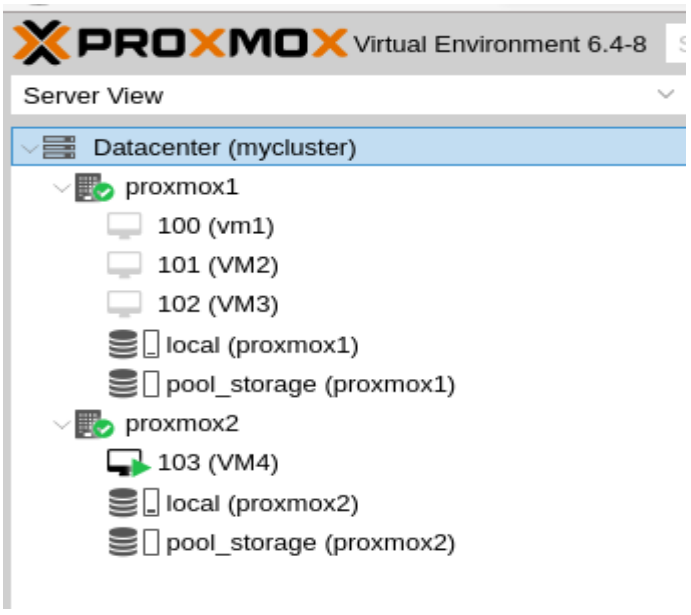
bridge-stp off
bridge-fd 0

On suit les étapes décrites précédemment afin d'installer les paquets requis, mettre en place le cluster Proxmox et enfin la mise en place du système de stockage distribué grâce à Ceph. Nous avons à disposition 3 OSDs, donc on peut mettre en place une règle de réplication pour éviter de perdre nos données en cas de dysfonctionnement d'un disque.

Après avoir fini la mise en place de notre pool de stockage Ceph avec une règle de réplication de 3, c'est à dire si on installe un VM sur l'un des noeuds le stockage de cette VM sera répliquée 3 fois pour assurer une disponibilité de ses données en cas de panne d'un disque ou même d'un noeud; on commence l'installation de nos machines virtuelles.

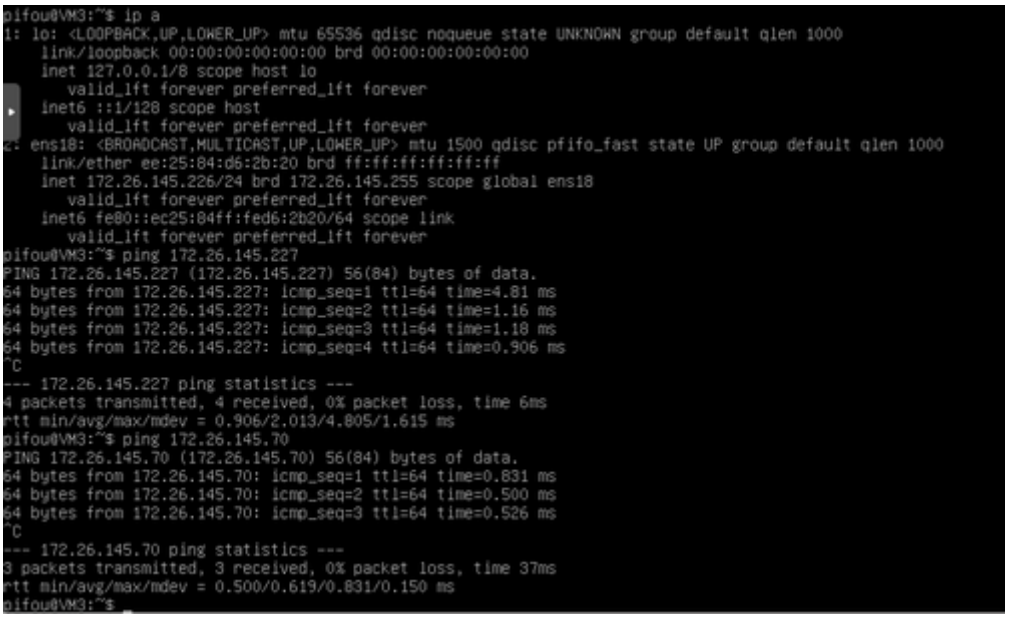
Lors de la création d'une machine virtuelle sur le nœud Proxmox2(Container) on a eu un message d'erreur comme quoi la machine ne supportait pas la virtualisation KVM ou qu'il était désactivé au niveau du BIOS. On a dû tester si le serveur pouvait faire de la virtualisation en installant le paquet cpu-checker et grâce à la commande **kvm-ok**. On a pu avoir la confirmation que le serveur était compatible à la virtualisation KVM mais qu'il fallait l'activer au niveau du BIOS ce qu'on a fait avant de pouvoir passer à l'installation de la machine virtuelle.

On constate une nette amélioration de l'infrastructure, car la durée totale d'installation d'un système sur une machine virtuelle est de 20 minutes environ loin des 110 minutes sur l'architecture précédente. On peut ainsi pouvoir paramétrer nos Vms pour les mettre en utilisation.



Machines Virtuelles disponibles dans le cluster

Nos machines virtuelles utilisent le bridge mis en place sur le nœud ou ils sont installés pour leur accès au réseau. Ensuite dans leur fichier /etc/network/interfaces on configure leur interface ens18 selon l'architecture défini au dessus, ils ont des adresses appartenant au réseau externe et grâce au bridge mis en place ils pourront avoir accès à Internet, pouvoir communiquer entre elles et le reste du réseau. Il ne faut pas oublier de redémarrer l'interface avec ifdown et ifup pour que la machine prenne en compte les modifications du fichier de configuration. Ensuite pour tester les configurations on effectue des pings sur l'ensemble des machines du réseau pour voir si on obtient des réponses.



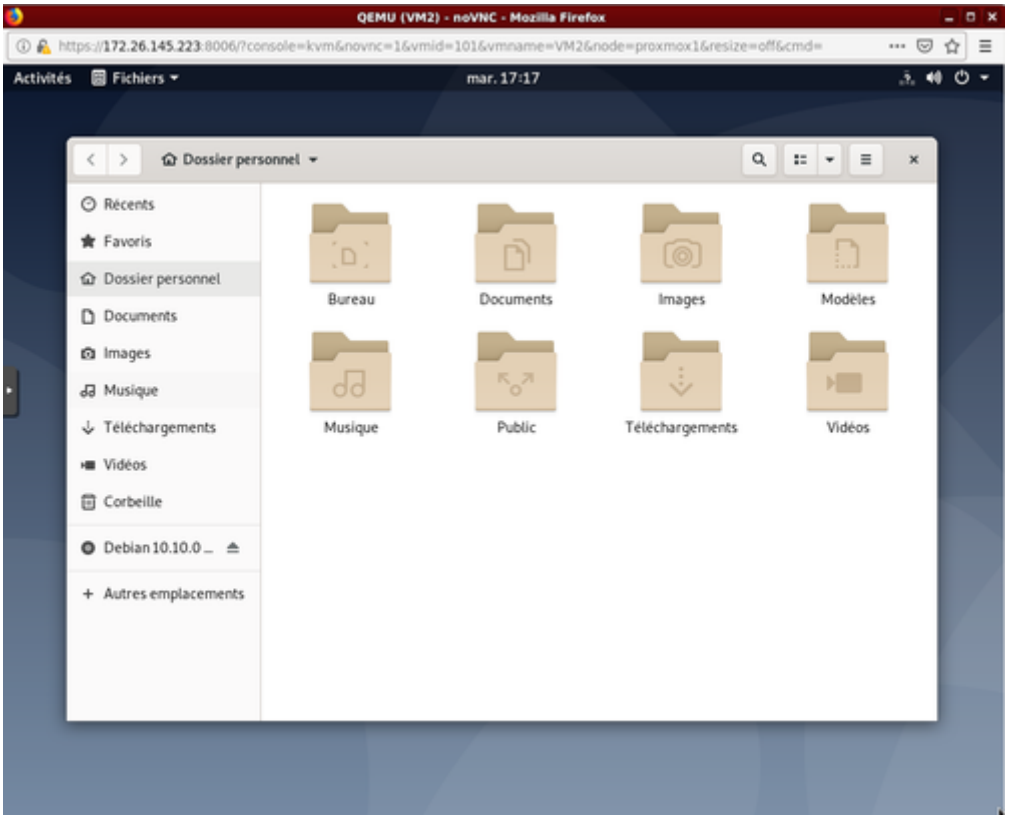
Test des configurations réseau

On a pu communiquer avec l'ensemble des machines du cluster, se connecter à Internet et activer l'accès à distance dans le réseau interne par ssh. Pour permettre l'accès aux VMs à partir de l'extérieur, on devra disposer d'une adresse IP "routée".

- **Installation d'une interface graphique sur une machine virtuelle:**

Pour installer une interface graphique sur une machine virtuelle, on a 3 options de paquets: **Gnome**, **KDE** et **LXDE**. On choisit d'installer la version minimale de gnome sans aucune application installée avant de rebooter la machine pour qu'elle puisse lancer l'interface graphique:

```
apt-get install gnome-core
reboot
```



Interface graphique installée sur une machine virtuelle

- **Migration de machine virtuelle:**

En virtualisation, la migration de machines virtuelles consiste à déplacer l'état d'une machine virtuelle, d'un hôte physique à un autre. Il existe plusieurs raisons pour lesquelles il est nécessaire de migrer des systèmes d'exploitation, la plus importante étant l'équilibrage de la charge de travail sur les serveurs physiques. Il est également nécessaire de migrer des machines virtuelles lorsqu'un hôte physique est défectueux ou nécessite une maintenance.

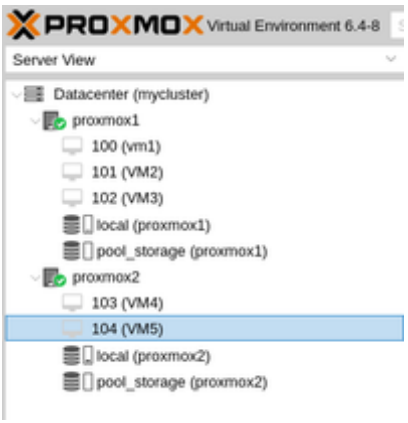
On va essayer de migrer une machine virtuelle (VM5) de notre cluster Proxmox d'un nœud à un autre.

Migration de VM5 de Proxmox1 à Proxmox2

Il suffit d'aller dans le menu de gestion de notre VM et cliquer sur l'onglet **Migrate** on obtient le menu suivant ou on choisit le noeud de destination de notre machine virtuelle:

Menu de paramétrage de la migration

On valide ensuite et on attend que la migration se termine. Si tout se passe bien, notre machine virtuelle se retrouvera dans le nœud de destination.



VM5 dans le noeud Proxmox2

5.3. Installation de machines virtuelles avec Xen pour comparer les performances:

Xen est un logiciel de (para)virtualisation de type hyperviseur. Il permet donc de faire tourner plusieurs systèmes d'exploitation (OS) sur une même ressource matérielle (PC, Serveur,…). Pour créer une machine virtuelle avec les mêmes caractéristiques que celles de notre cluster Proxmox, il suffit de lancer la commande suivante:

```
xen-create-image --hostname=proxmox --lvm=virtual --dist=buster --boot
--memory=4096Mb --fs=ext4 --vcpus=4 --bridge=ima5SC
--nameserver="193.48.57.48” --swap=1Gb --size=30G
```

Le temps d'installation totale du système sur la machine virtuelle Xen est de 5 minutes. On constate une grande différence de temps entre nos deux installations qui peut s'expliquer d'une part par le type de stockage distribué Ceph de notre machine virtuelle du cluster Proxmox. Mais cela ne suffirait pas pour justifier cette différence. On croit plutôt qu’elle viendrait du fait d'avoir choisi de mettre le trafic réseau et le trafic des données de stockage sur le même réseau car lors de l'installation des paquets de Debian sur la machine virtuelle, le second noeud Proxmox2 de notre cluster devient injoignable dans le réseau momentanément ce

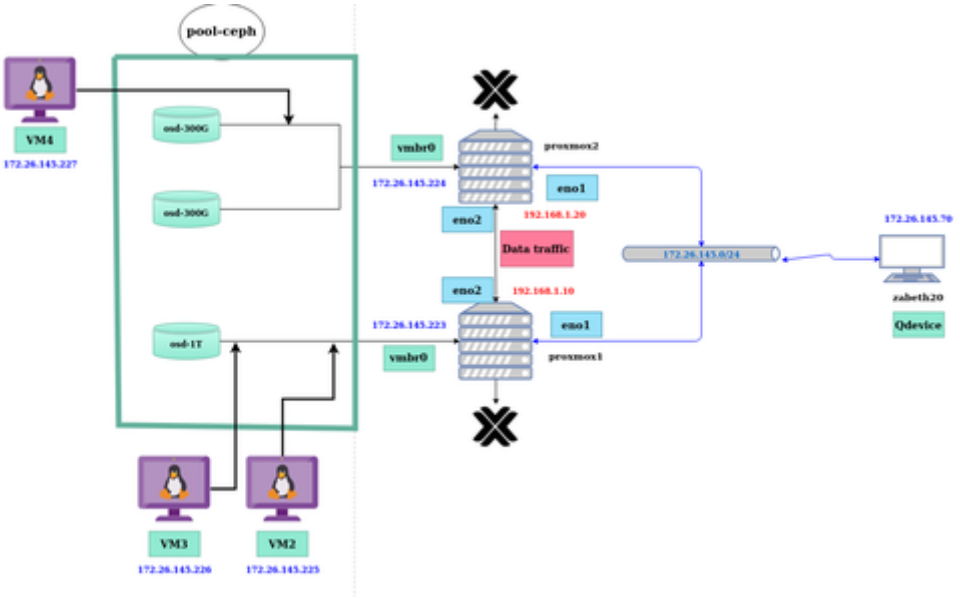
qui provoque la perte du corosync et le blocage des données le temps d'indisponibilité du second noeud.

On va essayer de mettre en place une nouvelle architecture en séparant le réseau de transit du corosync et celui du trafic des données de stockage.

6. Semaine 6:

6.1. Test de la nouvelle architecture:

On va mettre en place l'infrastructure décrite sur la figure suivante. La différence entre cette architecture et la dernière est la séparation du réseau de trafic des données de stockage et celui du trafic du Corosync.



Séparation du trafic Corosync et celui des données de stockage:

Pour mettre en place cette architecture, il a fallu modifier les paramètres Ceph du cluster dans /etc/pve/ceph.conf et remplacer l'ancien réseau par la nouvelle 192.168.1.0/24. Ce réseau a été paramétré sur les autres interfaces réseaux disponibles au niveau de nos deux serveurs à savoir l'interface **eno2** suivant l'architecture choisie.

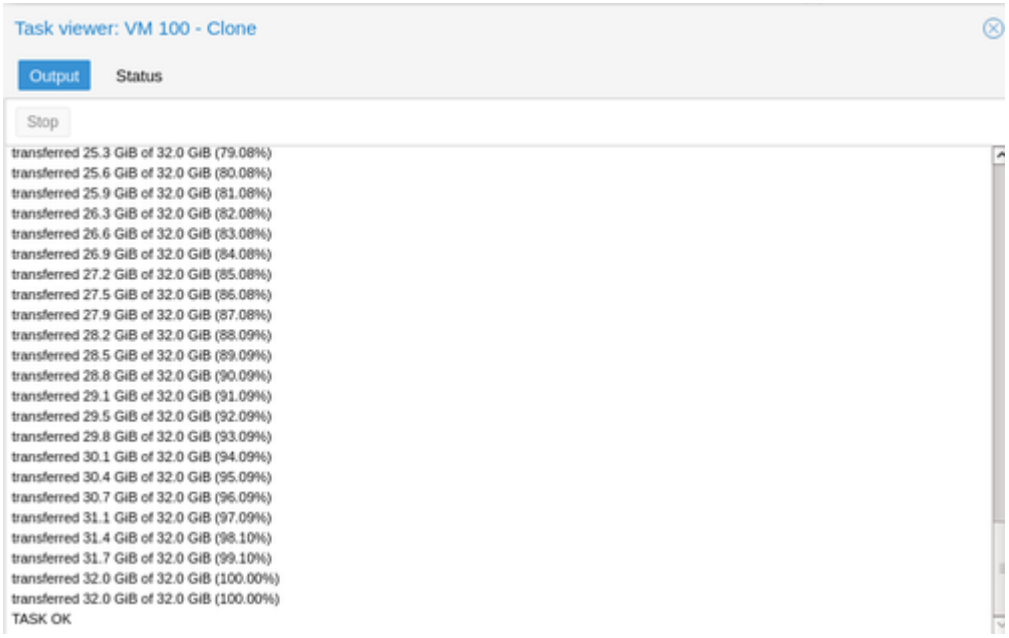
On va maintenant installer une machine virtuelle pour tester les performances de la nouvelle architecture. On constate que le temps d'installation total d'une nouvelle machine virtuelle est d'environ 12 minutes. Donc on a réussi à améliorer les performances de notre cluster et à trouver la principale cause de l'instabilité du cluster précédent. En effet, depuis la séparation des deux trafics du corosync et des données de stockage, le second nœud n'a jamais été injoignable ce qui garantit une infrastructure toujours disponible et une meilleure performance sur l'utilisation du stockage distribué.

Néanmoins, Xen propose la meilleure performance en termes de rapidité d'installation et cette différence s'explique particulièrement par le système de stockage distribué utilisé dans notre cluster Proxmox. Contrairement au Xen qui

stocke les données de la machine virtuelle sur l'espace de stockage local de la machine hôte, notre cluster Proxmox stocke quant à lui, l'ensemble des données des machines virtuelles créées dans le pool de stockage Ceph qui est partagé sur 2 noeuds séparés par un réseau et qui sera dupliqué pour garantir la disponibilité des données en cas de panne d'un noeud. En plus de cela, le temps d'écriture et de lecture de ces données dépend fortement du réseau par où passe le trafic des données de stockage.

6.2. Clonage de machines virtuelles:

Pour pouvoir déployer plusieurs machines virtuelles facilement et rapidement, on peut réaliser sur Proxmox le clonage de VM. En effet, il suffit d'installer une nouvelle machine virtuelle qui va servir de référence, ensuite cette machine sera convertie en template via l'interface de gestion du Proxmox et le menu de gestion des VMs avec l'onglet **Convert to template**. Cette machine sera automatiquement transformée en template, ensuite on pourra effectuer des clonages de VM à partir du même menu de gestion avec l'onglet **Clone**. On constate que la durée totale pour un clonage de VM ne dépasse pas une minute ce qui est très loin du temps d'installation d'une nouvelle machine virtuelle.Donc cette solution de clonage de machine virtuelle garantit une grande rapidité pour mettre en place un pool de VMs.



Clonage d'une machine virtuelle

Conclusion:

La virtualisation est aujourd'hui, sans aucun doute, une technique incontournable pour les systèmes d'information. Dans le cadre de notre projet, on a pu mettre en place deux types de virtualisation: la **virtualisation de serveur** qui permet d'exécuter plusieurs systèmes d'exploitation sur un seul serveur physique sous forme de machines virtuelles grâce à **Proxmox**; et la **virtualisation de stockage** qui consiste à assembler la capacité de stockage de multiples appareils de stockage en réseau sous forme d'un seul appareil de stockage (virtuel) grâce à Ceph.
La plupart des grandes entreprises dans le monde ont adopté cette technologie pour leur système d'information car elle offre plusieurs avantages techniques mais aussi financiers.
Ce projet nous a permis de comprendre cette notion de virtualisation, de saisir le potentiel et les avantages qu'elle offre mais aussi de développer les compétences pour la mettre en place. Parmi ces compétences, on peut citer l'administration d'un réseau, la mise en place d'une infrastructure de stockage distribuée sur un cluster mais aussi améliorer mes connaissances sur le système Linux.
Même si le projet était essentiellement porté sur la virtualisation, on a eu à manipuler le hardware et apporter une partie plutôt physique au projet à travers la modification des ventilateurs du serveur et aussi l'installation du serveur dans l'armoire à serveurs ce qui n'était pas de tout repos.

Webographie:

- https://pve.proxmox.com/wiki/
- https://docs.ceph.com/en/latest/start/intro/
- https://forum.proxmox.com/
- https://wiki.debian.org/
- https://www.devuan.org/
- https://linit.io/mettre-en-place-un-cluster-de-stockage-ceph/
- https://connect.ed-diamond.com/GNU-Linux-Magazine/GLMF-179/Presentation-et-installation-du-systeme-de-stockage-reparti-Ceph
- https://groupes.renater.fr/wiki/ceph/_media/public/dupont_in2p3-complete.pdf
- https://httpd.apache.org/docs/2.4/fr/howto/reverse_proxy.html
- https://wiki.debian.org/fr/BridgeNetworkConnections
- https://memo-linux.com/comment-creer-une-machine-virtuelle-avec-kvm/