V1.0

EK NETSEC LABS

PROJET – CLUSTER PROXMOX (LAB)

ELIAS KARISS © EK NETSEC LABS - TOUS DROITS RESERVES

Table des matières

introduction 3
Résumé exécutif 3
Présentation du projet 3
Objectifs 3
Matériel & prérequis 3
Schéma d'architecture 3
Plan d'adressage 4
Mise en place 4
Préparation de l'environnement 4
Config réseaux dans VMware (2 VMnet) 5
Configuration de VMnet8 (NAT) 6
Configuration de VMnet9 (Host-only) 7
Pourquoi deux réseaux (VMnet8 & VMnet9) 7
Créer la VM pve-1 et installer Proxmox 9.0-1 8
Installer Proxmox 9
Réseau pendant l'installation 9
Premier accès Web 10
Ajouter le 2e bridge (cluster) après installation 11
Créer la VM pve-2 et installer Proxmox 9.0-1 13
Vérifications rapides 13
Créer et joindre le cluster 15
Joindre pve-2 au cluster 16
Étape QDevice (quorum à 2 nœuds) 17
Créer la VM "qnetd" (Debian 12) 17
Installer le service qnetd (sur la VM Debian) 19
Attacher le QDevice au cluster (depuis pve-1) 20
Test éclair 20
Mise en place du NFS Storage 21
Caractéristique de la VM 21
1) Partitionner sda en GPT + 1 partition pleine 23
2) Formater en ext4 (0% réservé) et étiqueter 23
3) Créer le point de montage et rendre le montage persistant (noatime) 23
4) Vérifier 23
5) Droits pour NFS 24
exposer-le via NFS 24
Ajouter le NFS (GUI) 25
Améliorations possibles 27
1) Cluster & disponibilité 27

Elias KARISS © EK netsec labs - Tous droits réservés

2) Stockage 27
3) Réseau 27
4) Sauvegarde & PRA 27
5) Sécurité 27
6) Supervision & observabilité 27
7) Performance & bonnes pratiques 27

INTRODUCTION

Résumé exécutif

Ce document décrit la mise en place d'un cluster Proxmox VE pour un environnement de test/lab, basé sur deux nœuds (avec possibilité d'ajouter un troisième), un service de stockage NFS partagé pour la migration de machines virtuelles, et un QDevice (quorum) en configuration deux nœuds.

Présentation du projet

Objectif : Disposer d'un cluster Proxmox VE de laboratoire pour tester la gestion centralisée, la migration de VMs et l'extension d'un cluster (ajout d'un nœud). La haute disponibilité (HA) stricte n'est pas visée.

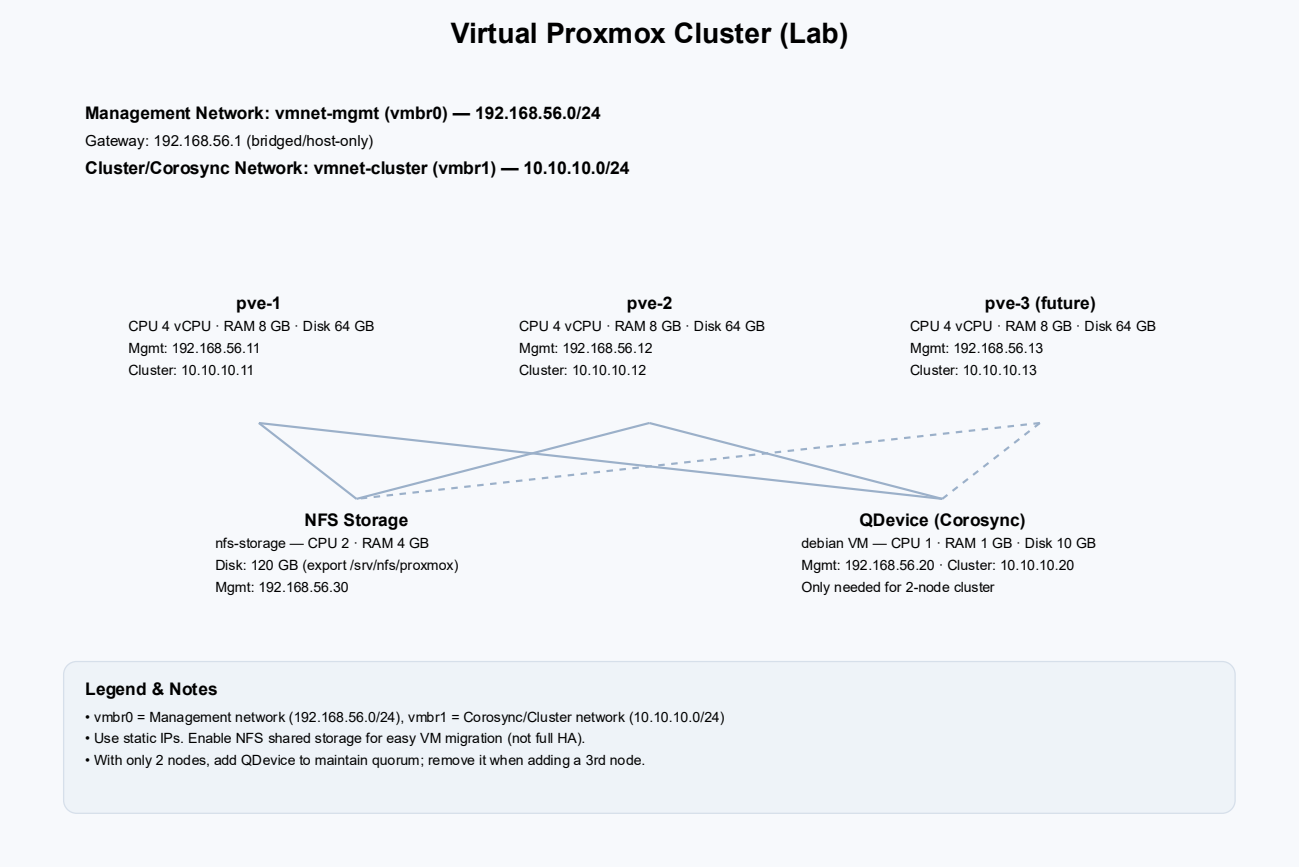
Objectifs

- Cluster de test/lab, gestion centralisée via Proxmox.
- NFS partagé pour simplifier la migration des VMs (pas de HA strict).
- Ajout ultérieur d'un 3e nœud et retrait du QDevice.

Matériel & prérequis

- Hôte physique : CPU Intel i7 12e gen, 32 Go RAM (ne pas allouer toutes les ressources aux VMs).
- Hyperviseur hôte : VMware (réseaux vmnet-mgmt, vmnet-cluster).
- ISOs : Proxmox VE 9.0-1, Debian 12 (NFS & QDevice).
- Accès réseau : IP fixes pour tous les nœuds, DNS interne optionnel.

Schéma d'architecture



Elias KARISS © EK netsec labs - Tous droits réservés

Plan d'adressage

Rôle	Interface	IP	Passerelle
pve-1	vmbr0 (mgmt)	192.168.56.11/24	192.168.56.1
pve-1	vmbr1 (cluster)	10.10.10.11/24	-
pve-2	vmbr0 (mgmt)	192.168.56.12/24	192.168.56.1
pve-2	vmbr1 (cluster)	10.10.10.12/24	-
pve-3 (futur)	vmbr0 (mgmt)	192.168.56.13/24	192.168.56.1
pve-3 (futur)	vmbr1 (cluster)	10.10.10.13/24	-
qdevice	vmbr0 (mgmt)	192.168.56.20/24	192.168.56.1
qdevice	vmbr1 (cluster)	10.10.10.20/24	-
nfs-storage	vmbr0 (mgmt)	192.168.56.30/24	192.168.56.1

Mise en place

1. Réseaux VMware

☐ Créer vmnet-mgmt (192.168.56.0/24, passerelle 192.168.56.1).
☐ Créer vmnet-cluster (10.10.10.0/24, sans passerelle).

2. Création des VMs

• pve-1 / pve-2 / pve-3 (futur) : 4 vCPU, 8 Go RAM, 64 Go disque, 2 NIC (vmbr0, vmbr1).
• nfs-storage : 2 vCPU, 4 Go RAM, 120 Go disque (Debian).
• qdevice : 1 vCPU, 1 Go RAM, 10 Go disque (Debian).

3. Installation Proxmox VE 9.0-1

☐ Installer Proxmox sur pve-1 et pve-2 (UEFI, disque 64 Go).
☐ Configurer vmbr0 (management) et vmbr1 (cluster) avec IP fixes.
☐ Accéder à l'interface web de pve-1 : https://<IP_pve-1>:8006

4. Création du cluster

5. QDevice (si 2 nœuds)

6. NFS (stockage partagé)

7. Tests & validation

8. Evolutions futures

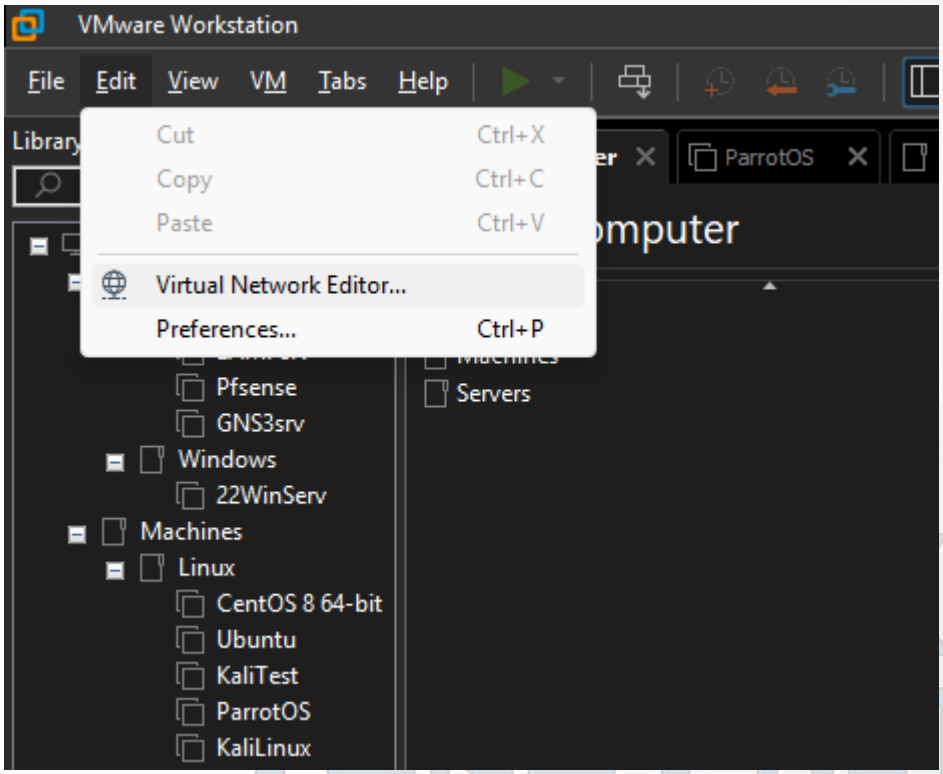
• Ajouter pve-3 et retirer le QDevice (quorum natif à 3 nœuds).
• Tester la réplication ZFS et/ou Ceph pour HA réelle.
• Intégrer des sauvegardes planifiées (Proxmox Backup Server).

PREPARATION DE L'ENVIRONNEMENT

Elias KARISS © EK netsec labs - Tous droits réservés

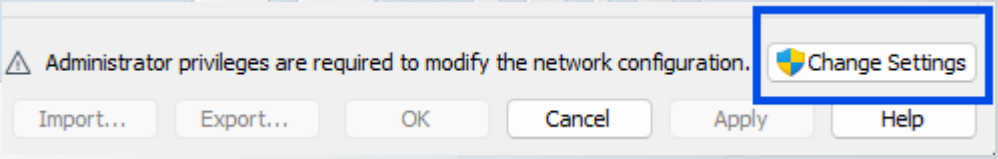
CONFIG RESEAUX DANS VMWARE (2 VMNET)

Ouvrir Virtual Network Editor (VMware Workstation).



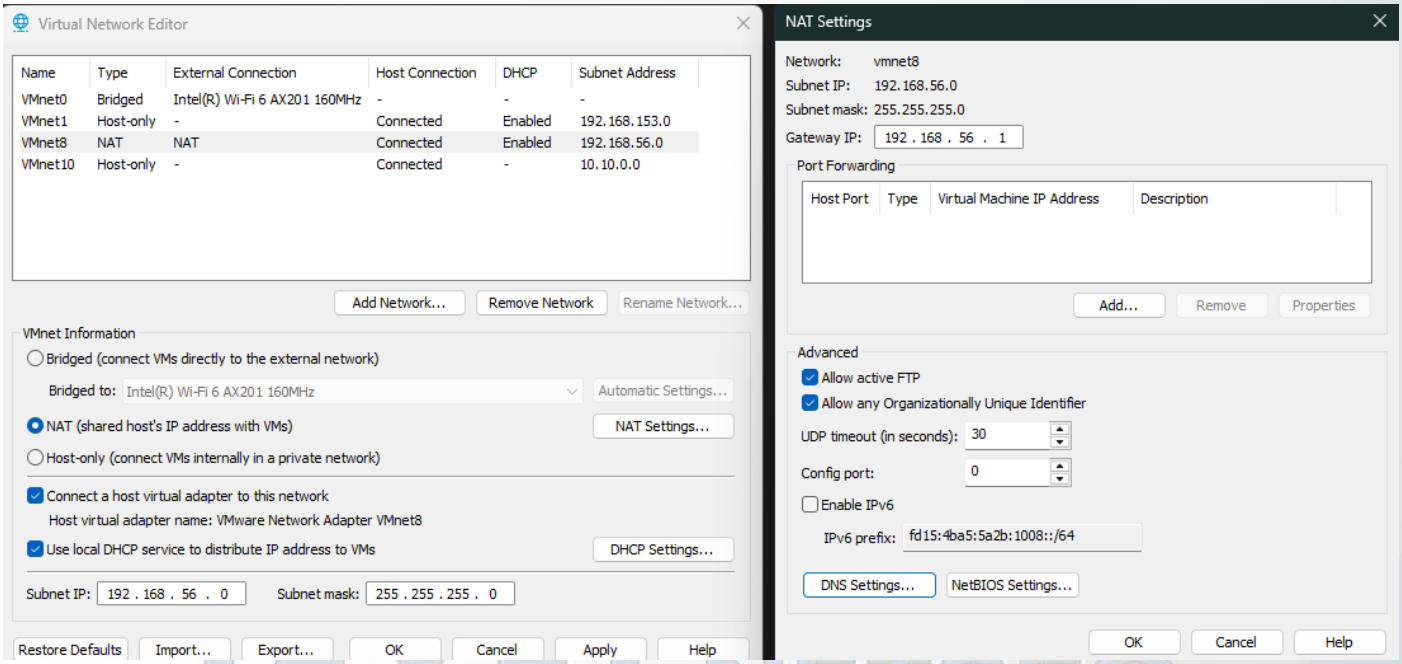
S'assurer d'avoir lancé au préalable VMware en administrateur

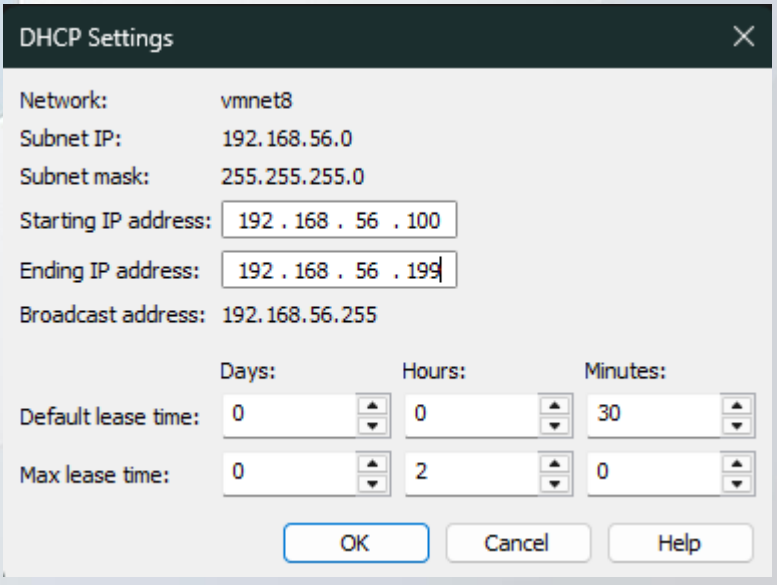
SI VMware n'est pas lancé en tant qu'administrateur, il est possible d'établir la modification des paramètres en activant le mode administrateur directement sur le panneau d'édition de paramètre Virtual Network Editor



Configuration de VMnet8 (NAT)

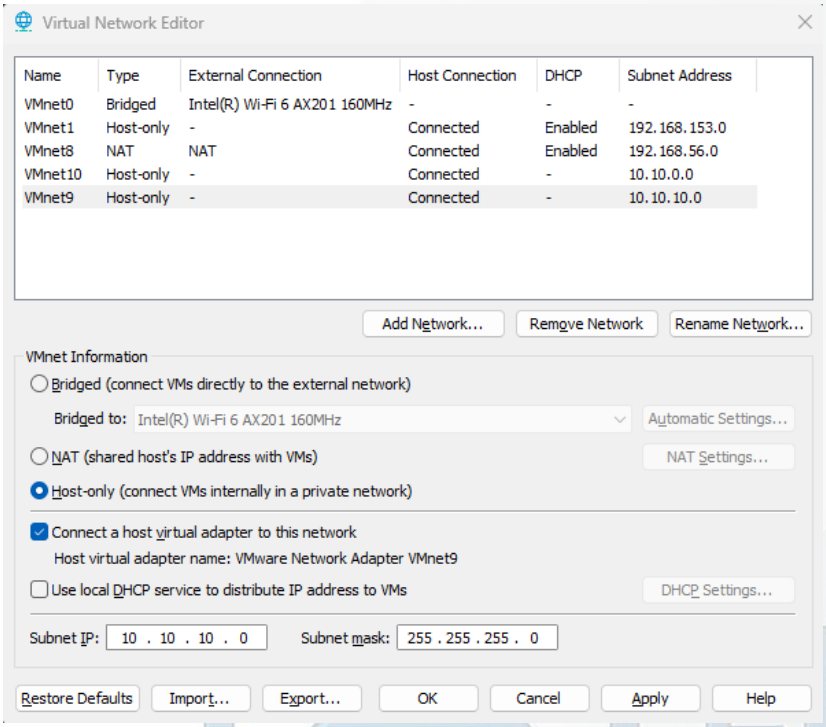
Type : NAT
Subnet : 192.168.56.0/24
Gateway (NAT) : 192.168.56.1





DHCP : Mettre une **plage** qui n'empiète pas sur nos IP fixes.
192.168.56.100–199

Configuration de VMnet9 (Host-only)



Type : Host-only
Subnet : 10.10.10.0/24
DHCP désactivé, pas de passerelle (laisser vide)
Apply/OK.

Astuce vérif : sur l'hôte, ipconfig (Windows) doit montrer un adaptateur host-only en 10.10.10.x.

```
Carte Ethernet VMware Network Adapter VMnet10 :

   Suffixe DNS propre à la connexion. . . :
   Adresse IPv6 de liaison locale. . . . .: fe80::ff2d:6cd6:b6ea:60ce%15
   Adresse IPv4. . . . . . . . . . . . . .: 10.10.0.1
   Masque de sous-réseau. . . . . . . . . : 255.255.255.0
   Passerelle par défaut. . . . . . . . . :
```

Pourquoi deux réseaux (VMnet8 & VMnet9)

VMnet8 — NAT / Management

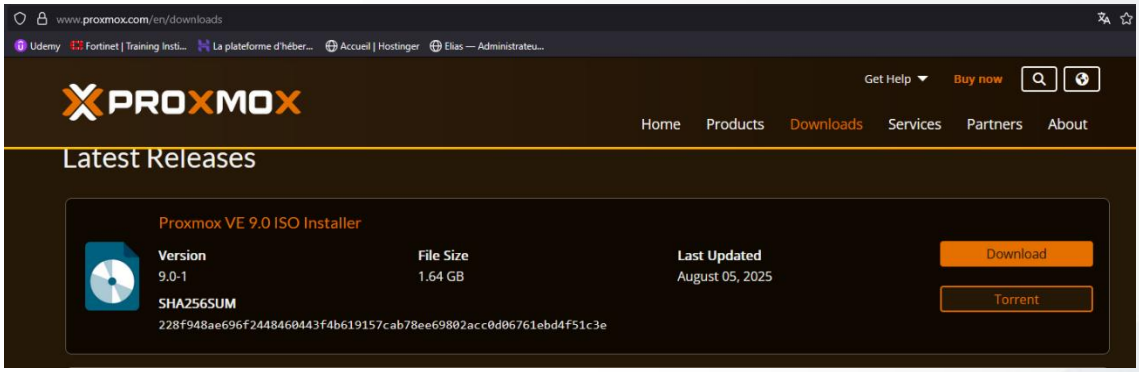
- But : accès admin à Proxmox (web :8006/SSH), Internet pour updates/NTP/DNS, accès au stockage NFS.
- NAT : les VMs sortent via l'hôte → pas besoin d'intégrer ton LAN, environnement reproductible.
- Plage : 192.168.56.0/24, Gateway 192.168.56.1 (route par défaut des nœuds).
- DHCP : optionnel (ex. 192.168.56.100–199) pour éviter tout conflit avec nos IP fixes.
- Adaptateur hôte : activé pour que la machine physique accède aux nœuds facilement.

VMnet9 — Host-only / Cluster (Corosync)

- But : réseau privé dédié au trafic cluster (heartbeats, quorum) → latence stable, isolation.
- Host-only : trafic confiné à l'hôte, aucune sortie vers Internet/LAN.
- Pas de passerelle : empêche tout routage accidentel du trafic Corosync.
- DHCP désactivé : IP statiques et déterministes, évite les renouvèlements DHCP qui cassent le cluster.
- Plage : 10.10.10.0/24 pour distinguer visuellement du réseau management.

Règle d'or : chaque nœud Proxmox a 2 NIC
NIC1 → VMnet8 (mgmt) · NIC2 → VMnet9 (cluster).

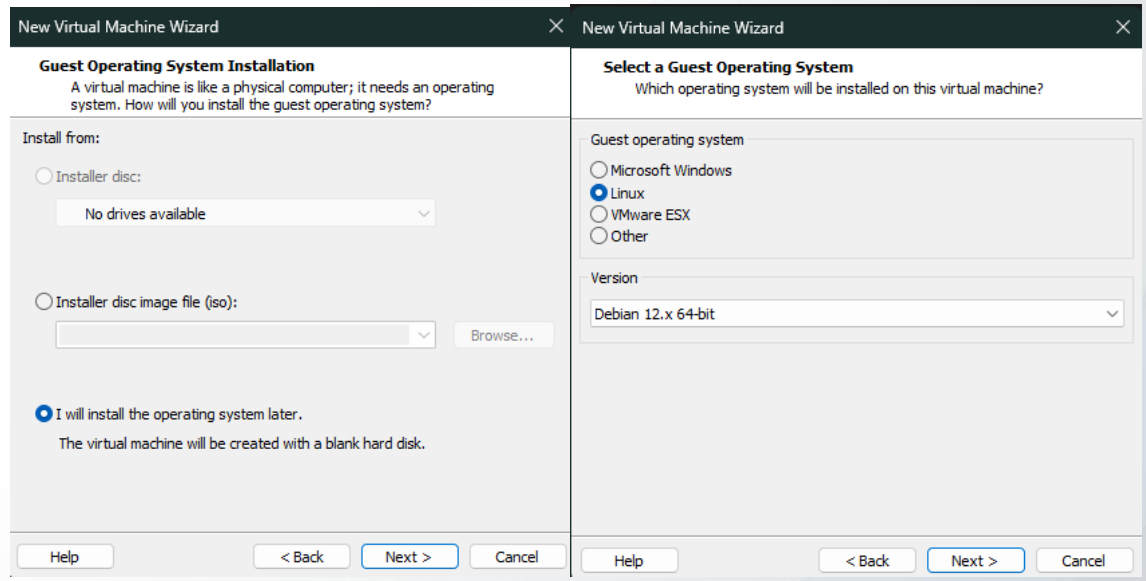
CREER LA VM PVE-1 ET INSTALLER PROXMOX 9.0-1

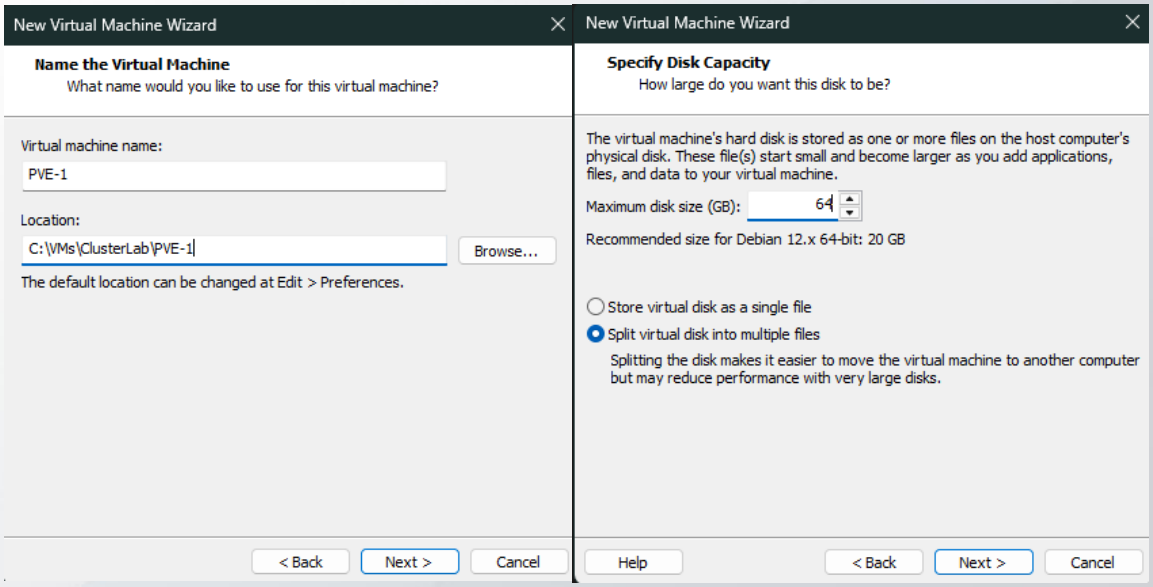


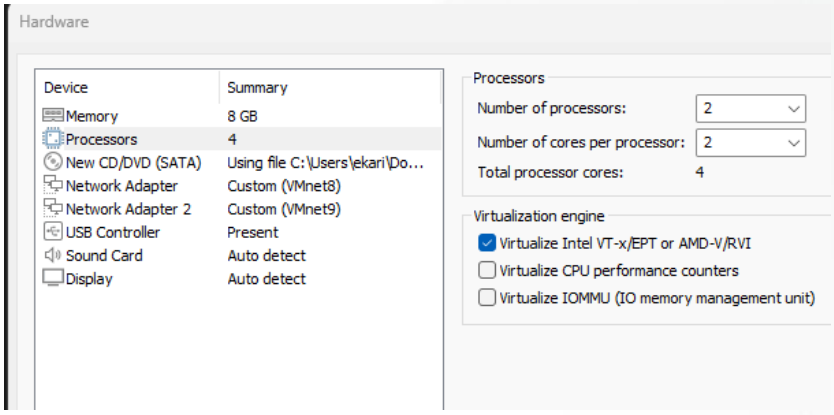
Depuis le lien officiel de proxmox, dans la page téléchargement, choisir la version 9.0-1

https://www.proxmox.com/en/downloads

Sur VMware, New Virtual Machine → Custom



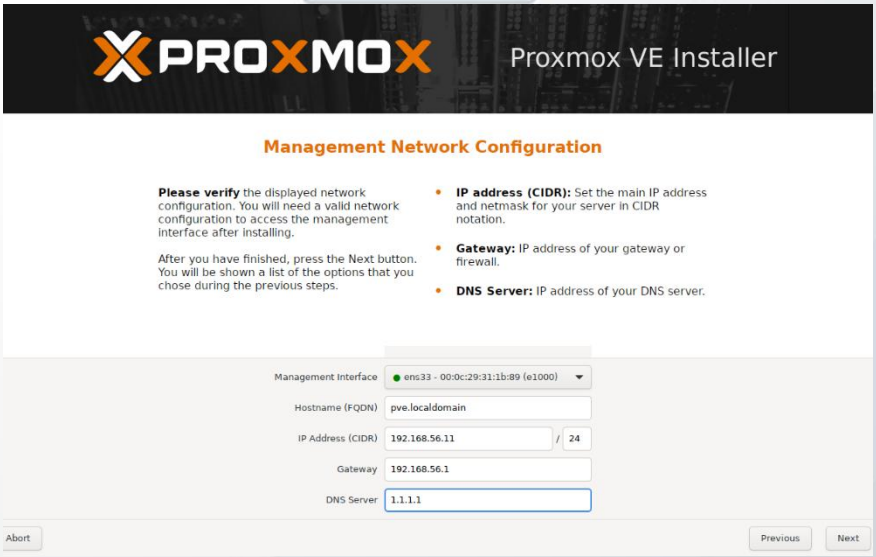




CPU : 4 vCPU
Cocher **Virtualize Intel VT-x/EPT (or AMD-V/RVI)**
RAM : 8192 Mo
Disque : 64 Go
2 cartes réseau :
Network Adapter 1 → Custom: VMnet8 (NAT) *mgmt*
Network Adapter 2 → Custom: VMnet9 (Host-only) *cluster*
Monte l'ISO Proxmox VE 9.0-1 dans le lecteur CD.

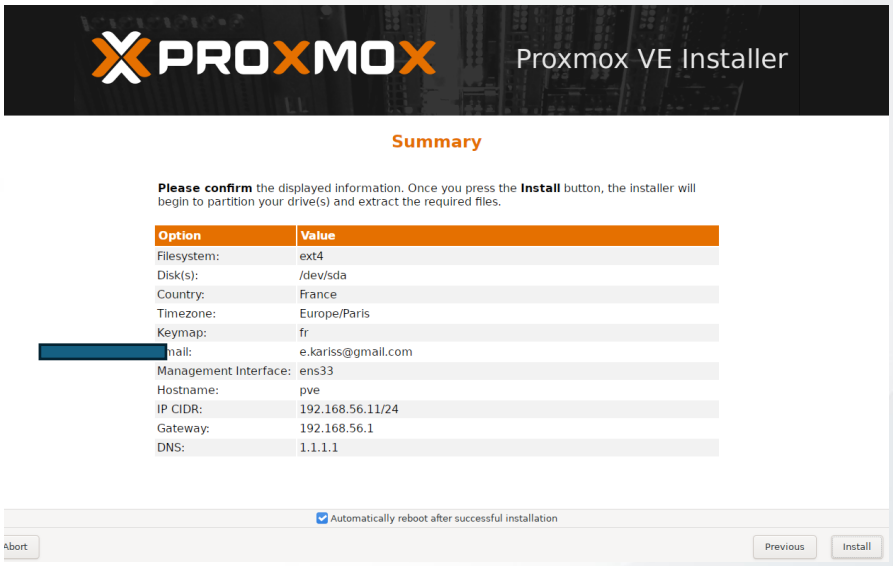
INSTALLER PROXMOX

Boot sur l'ISO → Install Proxmox VE



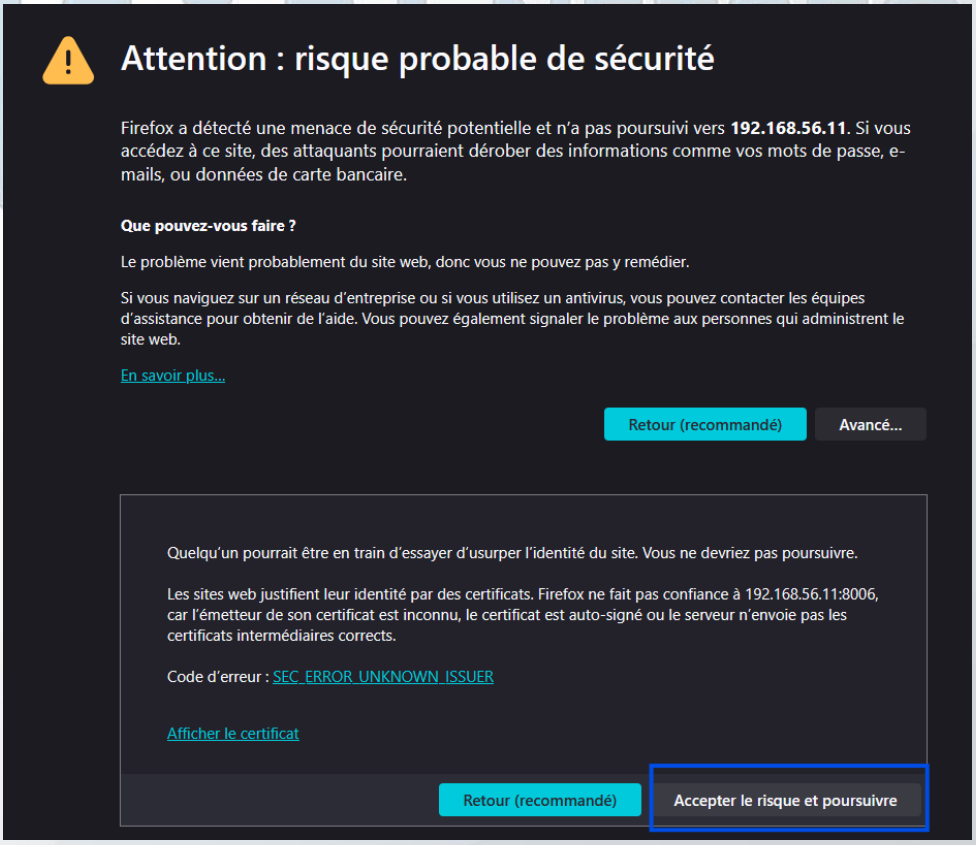
Réseau pendant l'installation

Choisir la carte liée à VMnet8 (mgmt)
IP : 192.168.56.11/24
Gateway : 192.168.56.1
DNS : 1.1.1.1
Terminer l'installation (mot de passe root + email).

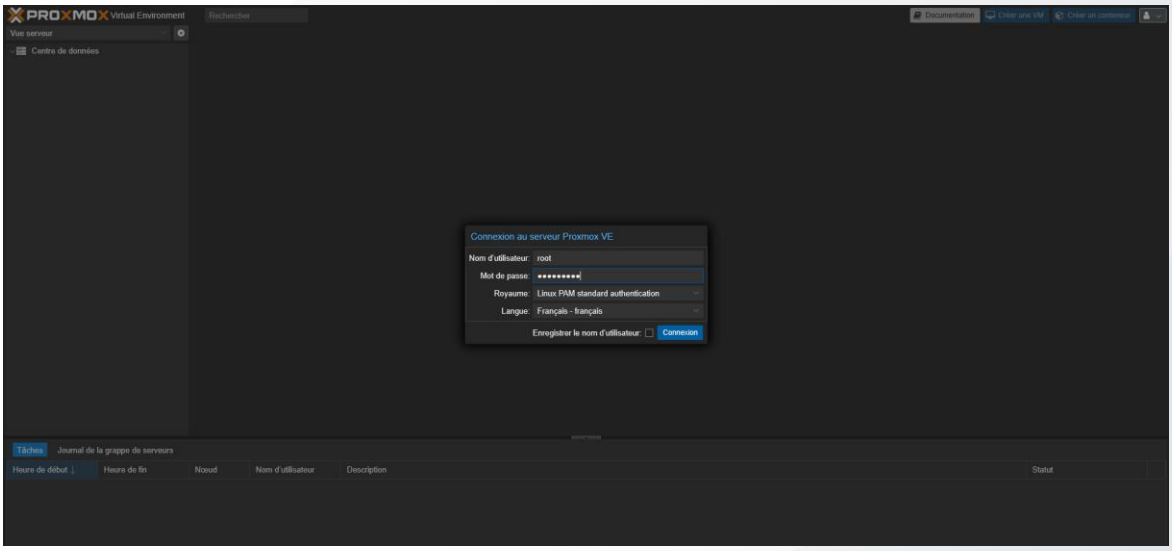


PREMIER ACCES WEB

Depuis le PC physique : ouvrir https://192.168.56.11:8006 (accepter le certificat).

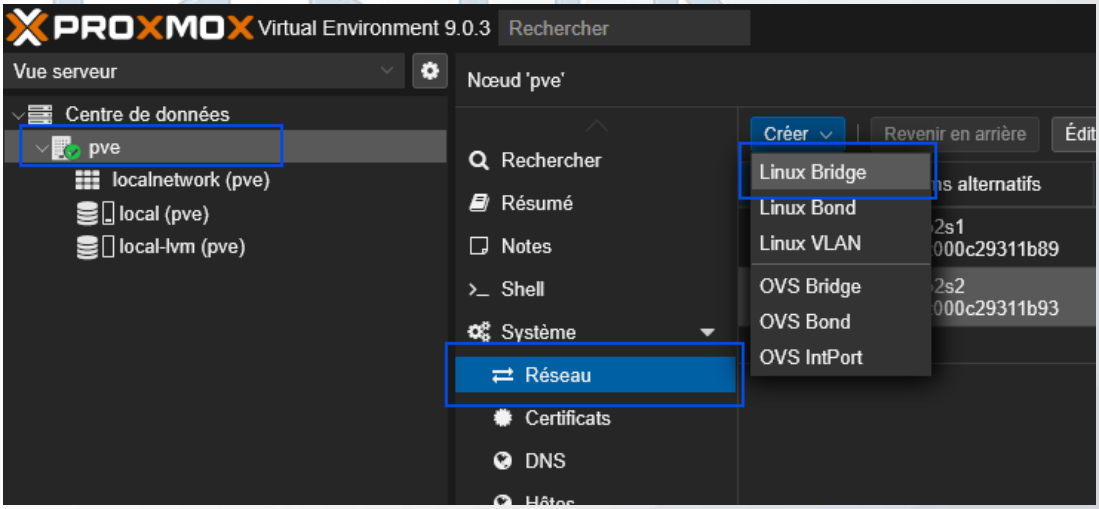


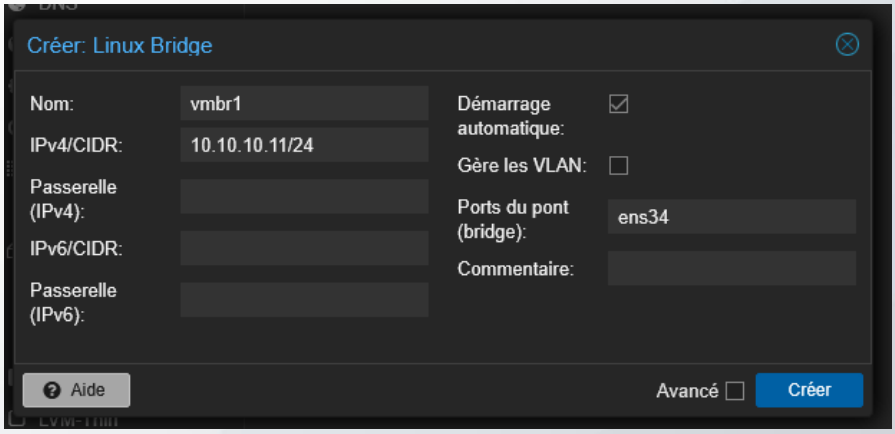
Connection en root.



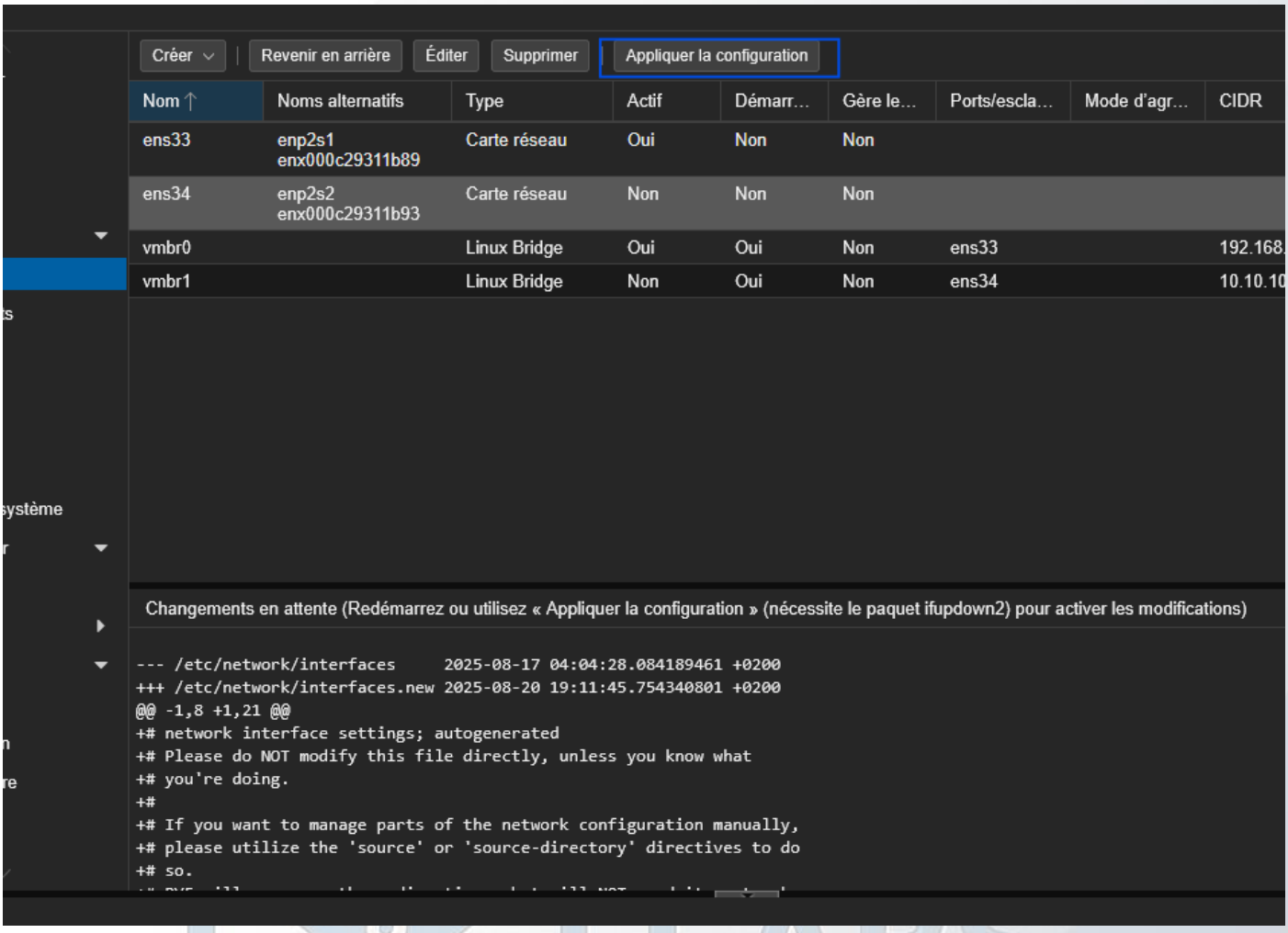
AJOUTER LE 2e BRIDGE (CLUSTER) APRES INSTALLATION

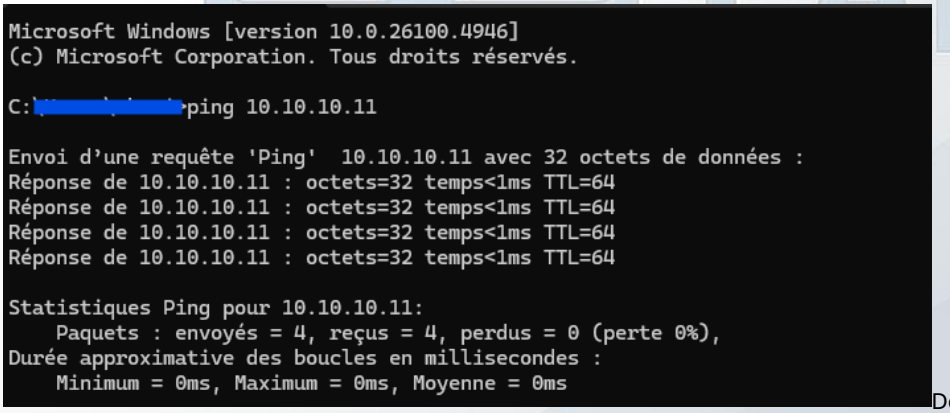
Datacenter → **pve-1** → **Système** > **Réseau** → **Créer** > **Linux Bridge**





Ports du pont : ens34
Nom : vmbr1
IPv4/CIDR : 10.10.10.11/24
Passerelle (IPv4) : laisser vide
Démarrage automatique : Oui
Créer puis **Apply Configuration**





Depuis l'hôte on ping pour tester la connexion

```
ping 10.10.10.11
```

CREER LA VM PVE-2 ET INSTALLER PROXMOX 9.0-1

Exécuter les mêmes actions que pour PVE-1 avec la config du PVE-2

Créer la VM dans VMware

New VM → Custom (ou Typical)
OS : **Debian 12 64-bit** (ou Other Linux 5.x 64-bit)
CPU 4 vCPU, **RAM 8 Go**, **Disque 64 Go** (SCSI/NVMe)
Network Adapter 1 → VMnet8 (NAT)
Network Adapter 2 → VMnet9 (Host-only)
Processors → coche **Virtualize Intel VT-x/EPT**
Monte l'ISO **Proxmox VE 9.0-1** puis boot.

Installer Proxmox

Install Proxmox VE
Choisis la carte **VMnet8** pendant l'install :

- IP : **192.168.56.12/24**
- Gateway : **192.168.56.1**
- DNS : 1.1.1.1 (ou ton DNS)
- Hostname : **pve-2**

Finis l'install, reboot, puis ouvre :
https://192.168.56.12:8006

Ajouter le bridge cluster (vmbr1)

Node pve-2 > System > Network > Create > Linux Bridge

- **Name**: vmbr1
- **Bridge ports**: (ex.) ens34
- **IPv4/CIDR**: **10.10.10.12/24**
- **Gateway**: (laisser vide)
- **Autostart**: Yes → **Apply Configuration**

Vérifications rapides

Tester la connexion avec la passerelle.

```
Ping 192.168.56.1.
```

Résultat :

```
root@pve-2:~# ping -c3 192.168.56.1
PING 192.168.56.1 (192.168.56.1) 56(84) bytes of data.

--- 192.168.56.1 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2082ms
```

Explication de la perte :

Sur Windows, le **pare-feu bloque l'ICMP** (ping) sur l'interface **VMnet8** par défaut.
Du coup, depuis les PVE, le ping vers **192.168.56.1** peut **perdre 100 %** des paquets **même si le NAT marche**.
Il faut donc modifier la règle du pare-feu windows en activant les profil ICMPv4-In

PowerShell

```
netsh advfirewall firewall add rule name="ICMPv4 Echo In VMnet8" dir=in action=allow protocol=icmpv4:8,any localip=192.168.56.1
```

```
PS C:\WINDOWS\system32> netsh advfirewall firewall add rule name="ICMPv4 Echo In VMnet8" dir=in action=allow protocol=icmpv4:8,any localip=192.168.56.1
>>
Ok.
```

```
root@pve:~# ping -c3 192.168.56.1
PING 192.168.56.1 (192.168.56.1) 56(84) bytes of data.
64 bytes from 192.168.56.1: icmp_seq=1 ttl=128 time=0.254 ms
64 bytes from 192.168.56.1: icmp_seq=2 ttl=128 time=0.431 ms
64 bytes from 192.168.56.1: icmp_seq=3 ttl=128 time=0.402 ms

--- 192.168.56.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2066ms
rtt min/avg/max/mdev = 0.254/0.362/0.431/0.077 ms
```

Elias KARISS © EK netsec labs - Tous droits réservés

On ping le PVE-1 via la console PVE-2

```
ping 10.10.10.11
```

```
root@pve-2:~# ping -c3 10.10.10.11
PING 10.10.10.11 (10.10.10.11) 56(84) bytes of data.
64 bytes from 10.10.10.11: icmp_seq=1 ttl=64 time=0.871 ms
64 bytes from 10.10.10.11: icmp_seq=2 ttl=64 time=0.433 ms
64 bytes from 10.10.10.11: icmp_seq=3 ttl=64 time=0.436 ms

--- 10.10.10.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2060ms
rtt min/avg/max/mdev = 0.433/0.580/0.871/0.205 ms
```

On verifie l'écoute pveproxy

```
Ss -tlnp | grep 8006
```

```
root@pve-2:~# ss -tlnp | grep 8006
LISTEN 0      4096               *:8006            *:*    users:(("pveproxy worker",pi
d=2331,fd=6),("pveproxy worker",pid=2330,fd=6),("pveproxy worker",pid=2329,fd=6),("pve
proxy",pid=2328,fd=6))
root@pve-2:~#
```

Depuis l'hôte windows :

```
Test-NetConnection 192.168.56.12 -Port 8006
```

```
PS C:\Users\ekari> Test-NetConnection 192.168.56.12 -Port 8006
ComputerName     : 192.168.56.12
RemoteAddress    : 192.168.56.12
RemotePort       : 8006
InterfaceAlias   : VMware Network Adapter VMnet8
SourceAddress    : 192.168.56.1
TcpTestSucceeded : True
```

Et on effectue les test ping depuis PVE-1

```
ping -c3 10.10.10.12

ping -c3 192.168.56.12
```

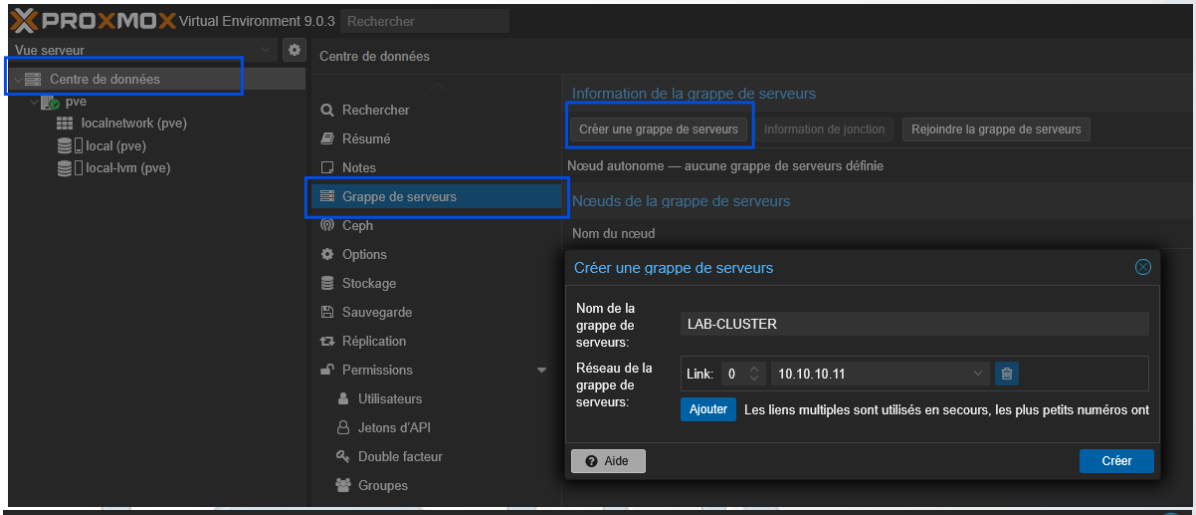
```
root@pve:~# ping -c3 10.10.10.12
PING 10.10.10.12 (10.10.10.12) 56(84) bytes of data.
64 bytes from 10.10.10.12: icmp_seq=1 ttl=64 time=0.224 ms
64 bytes from 10.10.10.12: icmp_seq=2 ttl=64 time=0.318 ms
64 bytes from 10.10.10.12: icmp_seq=3 ttl=64 time=0.246 ms

--- 10.10.10.12 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2051ms
rtt min/avg/max/mdev = 0.224/0.262/0.318/0.040 ms
root@pve:~#
root@pve:~# ping -c3 192.168.56.12
PING 192.168.56.12 (192.168.56.12) 56(84) bytes of data.
64 bytes from 192.168.56.12: icmp_seq=1 ttl=64 time=0.637 ms
64 bytes from 192.168.56.12: icmp_seq=2 ttl=64 time=1.56 ms
64 bytes from 192.168.56.12: icmp_seq=3 ttl=64 time=0.382 ms

--- 192.168.56.12 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2032ms
rtt min/avg/max/mdev = 0.382/0.860/1.561/0.506 ms
root@pve:~#
```

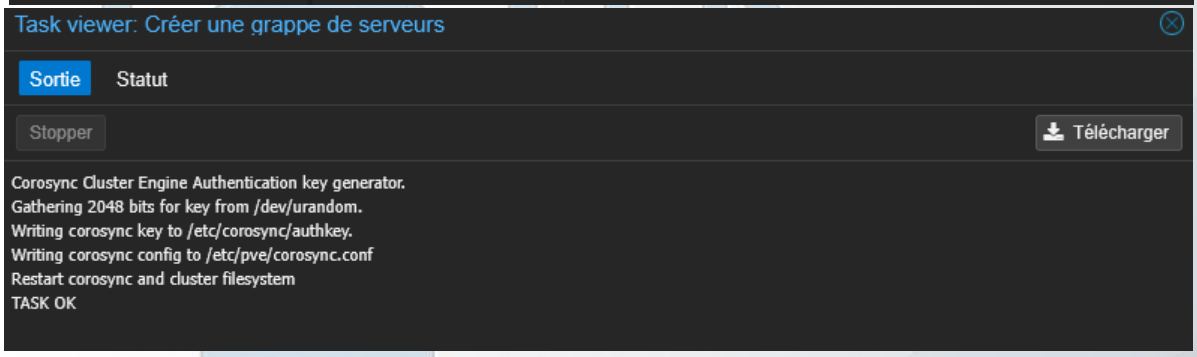
Elias KARISS © EK netsec labs - Tous droits réservés

CREER ET JOINDRE LE CLUSTER

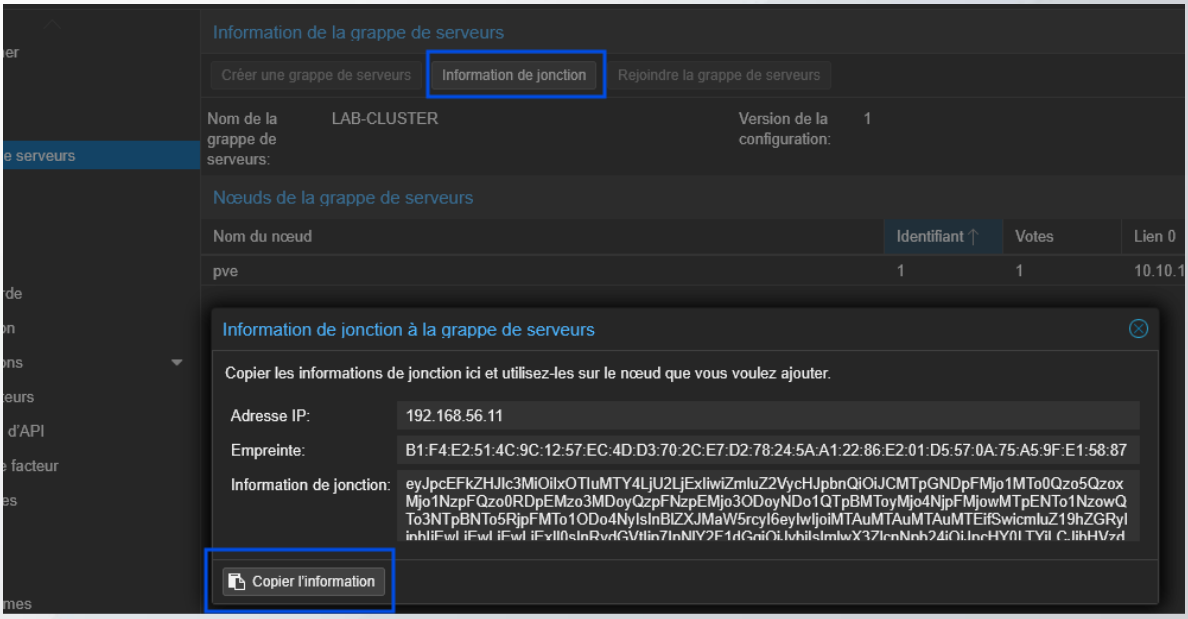
Création du cluster sur PVE-1

1. **Centre de donnée → Grappe de serveurs → Create Cluster**
2. **Cluster Name** : LAB-CLUSTER
3. **Network : 10.10.10.11 (vmbr1)** ← très important (corosync sur le réseau cluster)
4. **Creer** puis attends que ce soit “OK”.





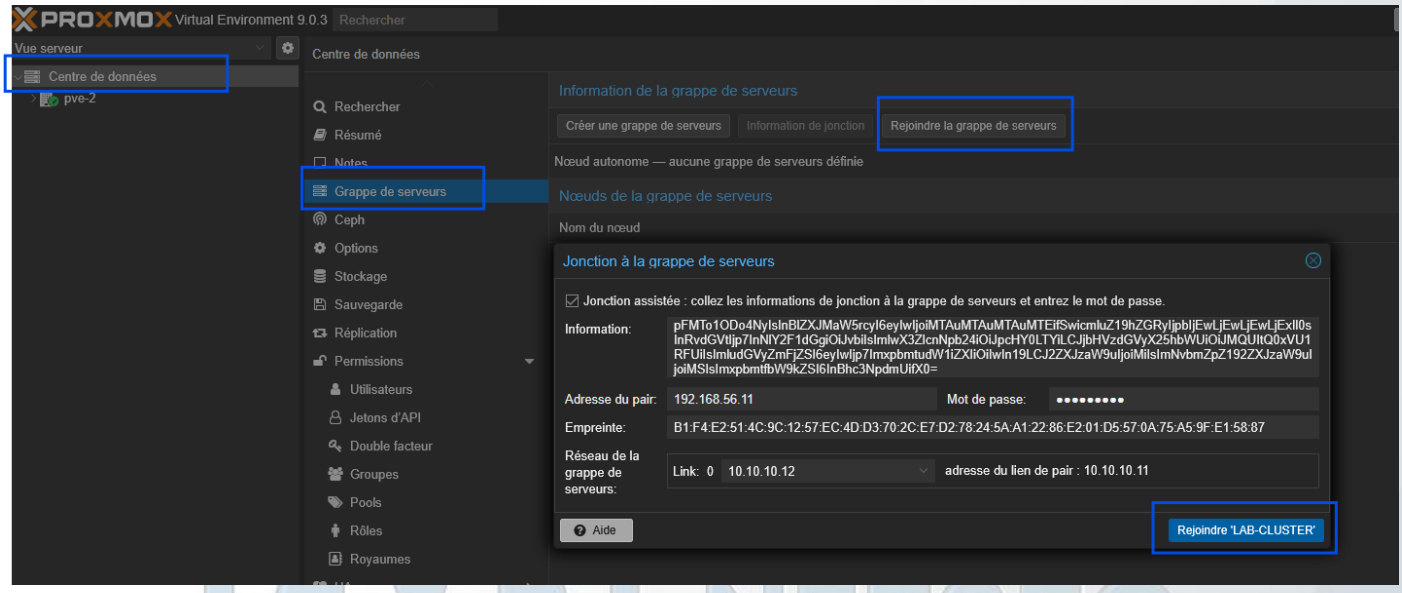
5. Clique **Information de jonction → Copier l’information** (à coller sur pve-2).



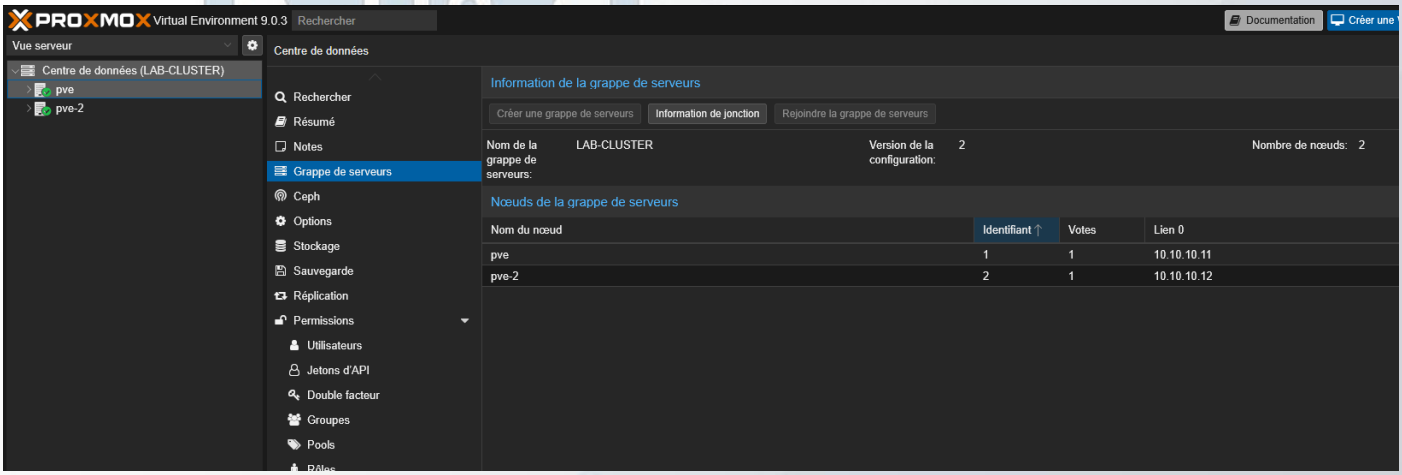
Joindre pve-2 au cluster

Méthode UI

1. Sur **pve-2** : **Centre de données → Grappe de serveur → Rejoindre la grappe de serveur**
2. **Information** : coller le bloc copié depuis pve-1
3. **Réseau de la grappe de serveurs** : **10.10.10.12 (vmbr1)**
4. Mot de passe root de **pve-1** → **Rejoindre 'LAB-CLUSTER'**.



Le Cluster est bien monté



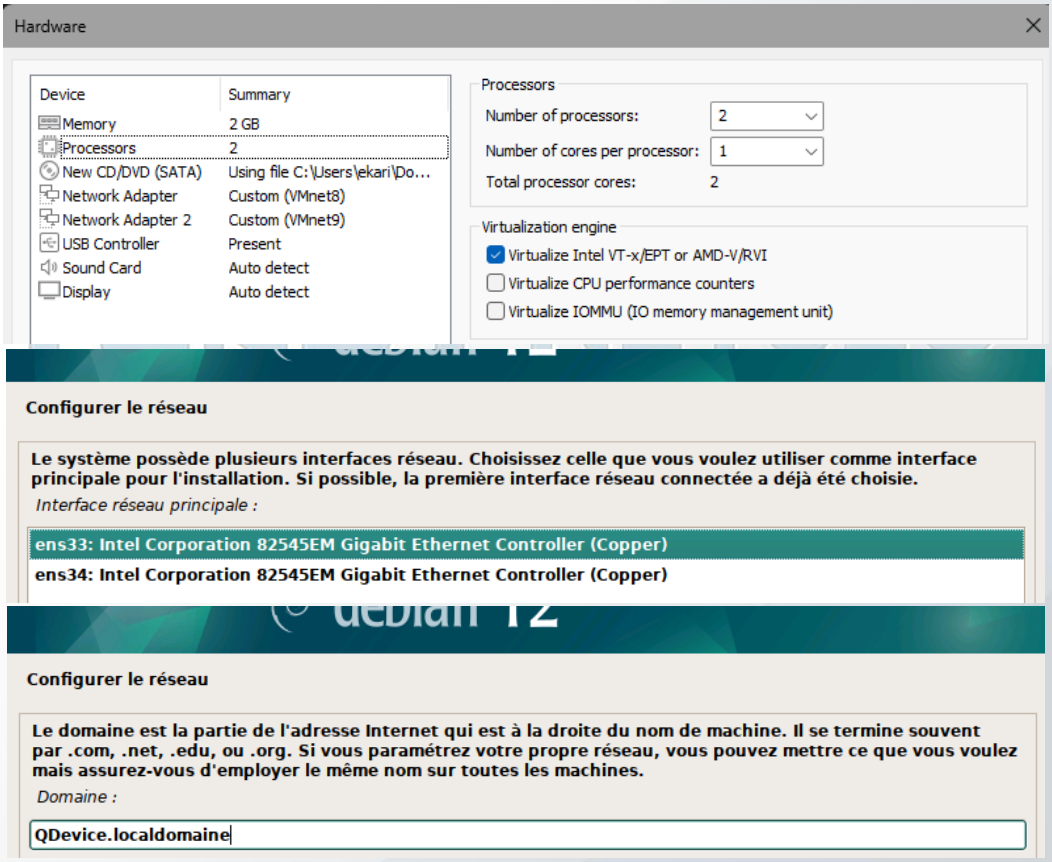
ÉTAPE QDEVICE (QUORUM A 2 NŒUDS)

le QDevice est un "arbitre réseau" pour un cluster à **2 nœuds**.

- **Sans QDevice** : un cluster de 2 nœuds a 2 voix. Le quorum exige **>50 %**, donc **2 voix**. Si un nœud tombe ou qu'il y a une coupure réseau, l'autre nœud n'a plus qu'**1/2** → **pas de quorum** → le cluster se met en sécurité (pour éviter le split-brain).
- **Avec QDevice (qnetd)** : on ajoute **une 3e voix** externe. Total = 3 voix → le côté qui voit l'**arbitre** garde **2/3** → **quorum maintenu**. En cas de partition, le QDevice **départage** et n'accorde sa voix qu'à **un seul côté**.
- **Ce que c'est** : un petit service qnetd (ex. Debian) joignable par chaque nœud (TCP **5403**), pas de données, **juste un vote**.
- **Où l'attacher** : sur le **réseau cluster** (chez toi : 10.10.10.20 sur **vmbr1**).
- **Quand l'enlever** : dès que tu ajoutes un **3e nœud Proxmox**, tu peux retirer le QDevice (pvecm qdevice destroy), car 3 nœuds suffisent au quorum.

Créer la VM "qnetd" (Debian 12)

- 1 vCPU, 1 Go RAM, 10 Go disque



Choix de l'installation sans l'environnement de bureau pour économiser de la RAM et du CPU

```
Debian GNU/Linux 13 QDevice tty1

QDevice login: userqdevice
Password:
Linux QDevice 6.12.38+deb13-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.12.38-1 (2025-07-16) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
userqdevice@QDevice:~$ _
```

- **NIC1 → VMnet8 (NAT)** → IP **192.168.56.20/24**
- **NIC2 → VMnet9 (Host-only)** → IP **10.10.10.20/24**
- Pas de passerelle sur l'interface **10.10.10.x**

```
userqdevice@QDevice:~$ su
Mot de passe :
root@QDevice:/home/userqdevice# nano /etc/network/interfaces_
```

```
  GNU nano 8.4                                          /etc/network/interfaces
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug ens33
iface ens33 inet dhcp
# This is an autoconfigured IPv6 interface
iface ens33 inet6 auto
```

Changer les paramètres en : rendant l'IP statique, entrant l'adresse IP et le bon masque réseau, inclure la passerelle par défaut et le dns nameserver

Pour ens34 on va garder la passerelle vide.

```
  GNU nano 8.4                                                  interfaces *
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
allow-hotplug ens33
iface ens33 inet static
        address 192.168.56.20/24
        gateway 192.168.56.1
        dns-nameservers 1.1.1.1

# This is an autoconfigured IPv6 interface
iface ens33 inet6 auto

# The secondary network interface
allow-hotplug ens34
iface ens34 inet static
        address 10.10.10.20/24
```

Vérification de l'attribution de la validation des paramètres

```
root@QDevice:/etc/network# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:0c:92:54 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    altname enx000c290c9254
    inet 192.168.56.103/24 brd 192.168.56.255 scope global dynamic noprefixroute ens33
       valid_lft 1192sec preferred_lft 967sec
    inet6 fd15:4ba5:5a2b:1008:20c:29ff:fe0c:9254/64 scope global dynamic mngtmpaddr proto kernel_ra
       valid_lft 86384sec preferred_lft 14384sec
    inet6 fd15:4ba5:5a2b:1008:30bf:c74:5d2:8b45/64 scope global dynamic mngtmpaddr noprefixroute
       valid_lft 86384sec preferred_lft 14384sec
    inet6 fe80::bb0d:146f:d1ae:ba83/64 scope link
       valid_lft forever preferred_lft forever
3: ens34: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default qlen 1000
    link/ether 00:0c:29:0c:92:5e brd ff:ff:ff:ff:ff:ff
    altname enp2s2
    altname enx000c290c925e
```

Un restart du service networking est donc nécessaire

```
systemctl restart networking.service
```

```
root@QDevice:/etc/network# systemctl restart networking.service
root@QDevice:/etc/network# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:0c:92:54 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    altname enx000c290c9254
    inet 192.168.56.20/24 brd 192.168.56.255 scope global ens33
       valid_lft forever preferred_lft forever
    inet6 fd15:4ba5:5a2b:1008:20c:29ff:fe0c:9254/64 scope global tentative dynamic mngtmpaddr proto kernel_ra
       valid_lft 86400sec preferred_lft 14400sec
3: ens34: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:0c:92:5e brd ff:ff:ff:ff:ff:ff
    altname enp2s2
    altname enx000c290c925e
    inet 10.10.10.20/24 brd 10.10.10.255 scope global ens34
       valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:fe0c:925e/64 scope link proto kernel_ll
       valid_lft forever preferred_lft forever
```

Installer le service qnetd (sur la VM Debian)

```
apt update

apt install -y corosync-qnetd
```

```
root@QDevice:/home/userqdevice# apt update
Atteint :1 http://security.debian.org/debian-security bookworm-security InRelease
Atteint :2 http://deb.debian.org/debian bookworm InRelease
Atteint :3 http://deb.debian.org/debian bookworm-updates InRelease
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Tous les paquets sont à jour.
root@QDevice:/home/userqdevice# apt install -y corosync-qnetd_
```

```
systemctl enable --now corosync-qnetd
```

```
root@QDevice:/home/userqdevice# systemctl enable --now corosync-qnetd
Synchronizing state of corosync-qnetd.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable corosync-qnetd
root@QDevice:/home/userqdevice#
```

```
ss -tlnp | grep 5403   # doit écouter sur 0.0.0.0:5403
```

Attacher le QDevice au cluster (depuis pve-1)

```
pvecm qdevice setup 10.10.10.20
```

Ici SSH bloque l'accès de l'installation du pvecm qdevice sur le PVE1

```
root@pve:~# pvecm qdevice setup 10.10.10.20
/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/root/.ssh/id_rsa.pub"
/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
root@10.10.10.20's password:
root@10.10.10.20's password: Permission denied, please try again.
```

On va temporairement libérer l'accès sur la machine QDevice pour permettre le setup du pvecm qdevice depuis PVE1

```
sed -i 's/^#PermitRootLogin.*/PermitRootLogin yes/' /etc/ssh/sshd_config

sed -i 's/^#PassworAuthentication.*/ PassworAuthentication yes/' /etc/ssh/sshd_config
```

```
root@QDevice:/home/userqdevice# sed -i 's/^#PermitRootLogin.*/PermitRootLogin yes/' /etc/ssh/sshd_config
root@QDevice:/home/userqdevice# sed -i 's/^#PasswordAuthentication.*/PasswordAuthentication yes/' /etc/ssh/sshd_config
root@QDevice:/home/userqdevice# systemctl restart ssh
```

On test la connexion SSH depuis PVE1

```
ssh root@10.10.10.20
```

```
root@pve:~# ssh root@10.10.10.20
root@10.10.10.20's password:
Linux QDevice 6.1.0-38-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.147-1 (2025-08-02) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
```

Une fois corosync installé sur toute les machines et le setup exécuté on passe à un test pour vérifier que tout fonctionne

Test éclair

- On arrête corosync sur **pve-2** :

```
systemctl stop corosync
```

```
root@pve-2:/etc/network# systemctl stop corosync
root@pve-2:/etc/network# _
```

- Sur **pve-1** : pvecm status → le quorum doit rester **OK** grâce au QDevice.

```
Quorum information
------------------
Date:             Sat Aug 23 22:06:04 2025
Quorum provider:  corosync_votequorum
Nodes:            1
Node ID:          0x00000001
Ring ID:          1.68
Quorate:          Yes
```

- On relance :

```
systemctl start corosync sur pve-2.
```

MISE EN PLACE DU NFS STORAGE

Caractéristique de la VM

- **Nom** : nfs-storage
- **OS** : Debian 12
- **vCPU** : **4**
- **RAM** : **4 Go**
- **Disque** :
  - **Disque 1 (OS) 16 Go** + **Disque 2 (données NFS) 250 Go**
  - **FS** : ext4, point de montage **/srv/nfs/proxmox** (option : noatime)
- **Contrôleurs** : SCSI (LSI Logic SAS) ou NVMe ; **carte réseau VMXNET3**
- **Réseau** :
  - **NIC1 → VMnet8 (NAT / mgmt)**

IP fixe : 192.168.56.30/24, **GW 192.168.56.1**, **DNS 1.1.1.1**

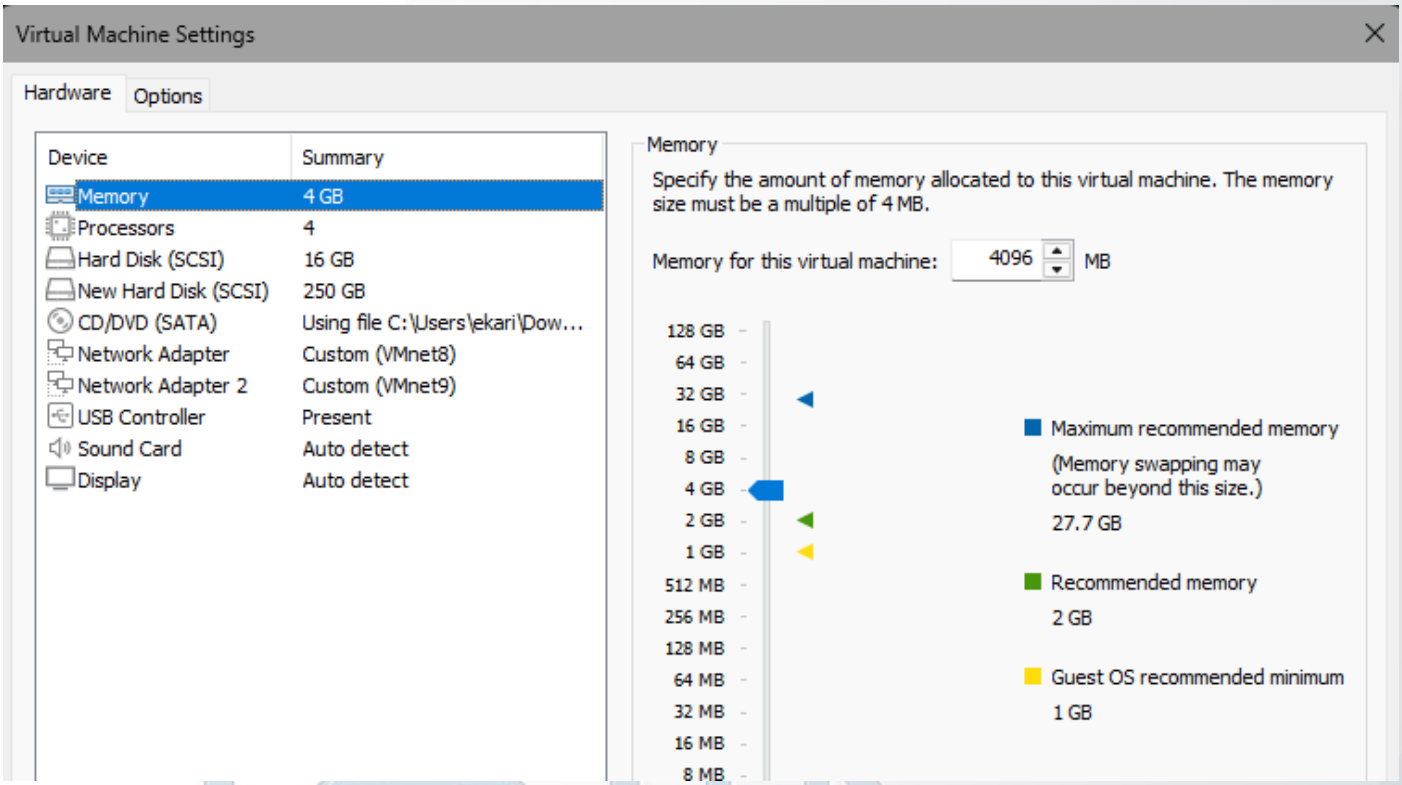
  - **NIC2 → VMnet9 (NAT / mgmt)**

IP fixe : 10.10.10.30/24

- **Paquets** : nfs-kernel-server (obligatoire), nfs-common, open-vm-tools, htop (facultatif)
- **Export NFS (v3/v4)** :

/srv/nfs/proxmox 192.168.56.0/24(rw,sync,no_subtree_check,no_root_squash)

- **Service** : nfs-server activé ; nombre de threads par défaut OK (8).



Partition des disques

Je lance la commande `lsblk` pour identifier mes disques

```
NAME   MAJ:MIN RM  SIZE RO TYPE MOUNTPOINTS
sda      8:0    0  250G  0 disk
sdb      8:16   0   16G  0 disk
├─sdb1   8:17   0   15G  0 part /
├─sdb2   8:18   0    1K  0 part
└─sdb5   8:21   0  975M  0 part [SWAP]
sr0     11:0    1 1024M  0 rom
root@NFS-STORAGE:/home/userstorage# _
```

D'après lsblk :

- **/dev/sdb (16 Go)** → c'est le **disque système** (sdb1 monté sur /, sdb5 = swap).
- **/dev/sda (250 Go)** → **disque libre** (parfait pour les données NFS).

Préparer et monter /srv/nfs/proxmox sir /dev/sda

J'installe le paquet parted

```
apt install -y parted
```

1) Partitionner sda en GPT + 1 partition pleine

```
parted -s /dev/sda mklabel gpt

parted -s /dev/sda mkpart primary ext4 0% 100%

partprobe
```

```
root@NFS-STORAGE:~# parted -s /dev/sda mklabel gpt
root@NFS-STORAGE:~# parted -s /dev/sda mkpart primary ext4 0% 100%
root@NFS-STORAGE:~# partprobe
root@NFS-STORAGE:~# _
```

2) Formater en ext4 (0% réservé) et étiqueter

mkfs.ext4 -L nfsdata -m 0 /dev/sda1

```
root@NFS-STORAGE:~# mkfs.ext4 -L nfsdata -m 0 /dev/sda1
mke2fs 1.47.0 (5-Feb-2023)
Creating filesystem with 65535488 4k blocks and 16384000 inodes
Filesystem UUID: 2c7b378c-f310-40f2-b714-0ca76ec9411d
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (262144 blocks): done
Writing superblocks and filesystem accounting information: done
```

3) Créer le point de montage et rendre le montage persistant (noatime)

```
mkdir -p /srv/nfs/proxmox

UUID=$(blkid -s UUID -o value /dev/sda1)

echo "UUID=$UUID  /srv/nfs/proxmox  ext4  defaults,noatime  0  2" >> /etc/fstab

mount -a
```

```
root@NFS-STORAGE:~# mkdir -p /srv/nfs/proxmox
root@NFS-STORAGE:~# UUID=$(blkid -s UUID -o value /dev/sda1)
root@NFS-STORAGE:~# echo "UUID=$UUID /srv/nfs/proxmox ext4 defaults,noatime 0 2" >> /etc/fstab
root@NFS-STORAGE:~# mount -a
```

4) Vérifier

```
lsblk -f

findmnt /srv/nfs/proxmox

df -h /srv/nfs/proxmox
```

Elias KARISS © EK netsec labs - Tous droits réservés

```
root@NFS-STORAGE:~# mkdir -p /srv/nfs/proxmox
root@NFS-STORAGE:~# UUID=$(blkid -s UUID -o value /dev/sda1)
root@NFS-STORAGE:~# echo "UUID=$UUID /srv/nfs/proxmox ext4 defaults,noatime 0 2" >> /etc/fstab
root@NFS-STORAGE:~# mount -a
montage : (astuce) votre fstab a été modifié mais systemd utilise encore
        l'ancienne version ; utilisez « systemctl daemon-reload » pour recharger.
root@NFS-STORAGE:~# lsblk -f
NAME   FSTYPE FSVER LABEL   UUID                                 FSAVAIL FSUSE% MOUNTPOINTS
sda
└─sda1 ext4   1.0   nfsdata 2c7b378c-f310-40f2-b714-0ca76ec9411d    245G     0% /srv/nfs/proxmox
sdb
├─sdb1 ext4   1.0           d130d1cb-9c73-45e7-9930-dcf575474aff   12,1G    12% /
├─sdb2
└─sdb5 swap   1             97c21b9e-b71f-4234-a6e3-41bfef07344e                [SWAP]
sr0
root@NFS-STORAGE:~# findmnt /srv/nfs/proxmox
TARGET          SOURCE    FSTYPE OPTIONS
/srv/nfs/proxmox /dev/sda1 ext4   rw,noatime
root@NFS-STORAGE:~# df -h /srv/nfs/proxmox/
Sys. de fichiers Taille Utilisé Dispo Uti% Monté sur
/dev/sda1          246G     28K  246G   1% /srv/nfs/proxmox
root@NFS-STORAGE:~#
```

5) Droits pour NFS

```
chown -R nobody:nogroup /srv/nfs/proxmox
```

```
root@NFS-STORAGE:~# chown -R nobody:nogroup /srv/nfs/proxmox/
root@NFS-STORAGE:~# _
```

EXPOSER-LE VIA NFS

```
echo "/srv/nfs/proxmox 192.168.56.0/24(rw,sync,no_subtree_check,no_root_squash)" >> /etc/exports

exportfs -ra

systemctl restart nfs-server
```

```
root@NFS-STORAGE:~# echo "/srv/nfs/proxmox 192.168.56.0/24(rw,sync,no_subtree_check,no_root,squash)"
/srv/nfs/proxmox 192.168.56.0/24(rw,sync,no_subtree_check,no_root,squash)
root@NFS-STORAGE:~# exportfs -ra
root@NFS-STORAGE:~# systemctl restart nfs-server
root@NFS-STORAGE:~#
```

Sanity check (10 sec)

Sur **pve-1** et **pve-2** :

ping -c2 192.168.56.30

```
root@pve:~# ping -c3 192.168.56.30
PING 192.168.56.30 (192.168.56.30) 56(84) bytes of data.
64 bytes from 192.168.56.30: icmp_seq=1 ttl=64 time=1.18 ms
64 bytes from 192.168.56.30: icmp_seq=2 ttl=64 time=0.303 ms
64 bytes from 192.168.56.30: icmp_seq=3 ttl=64 time=0.592 ms

--- 192.168.56.30 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2025ms
rtt min/avg/max/mdev = 0.303/0.693/1.184/0.366 ms
root@pve:~# _
```

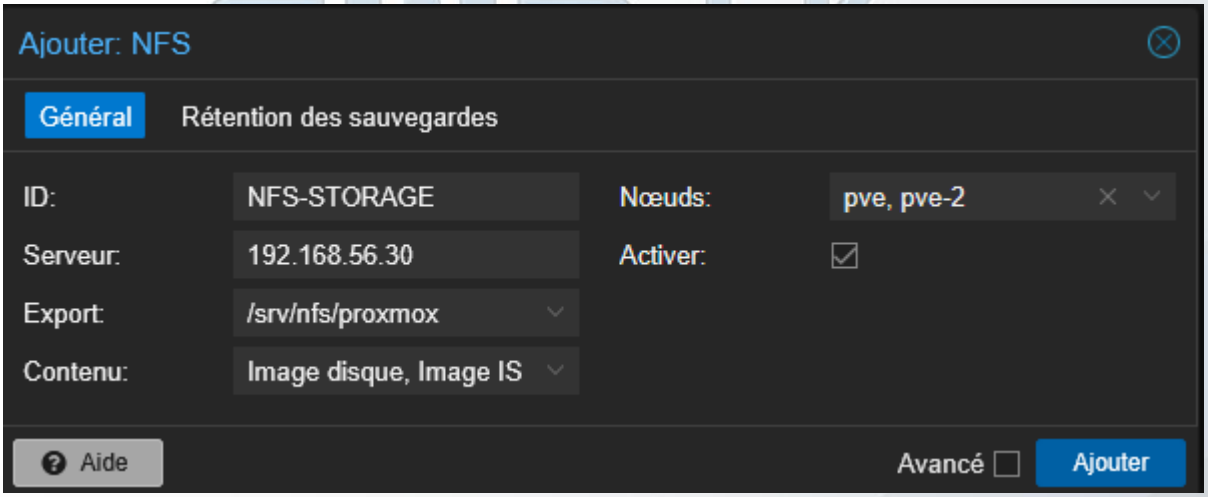
Elias KARISS © EK netsec labs - Tous droits réservés

```
root@pve-2:~# ping -c3 192.168.56.30
PING 192.168.56.30 (192.168.56.30) 56(84) bytes of data.
64 bytes from 192.168.56.30: icmp_seq=1 ttl=64 time=0.647 ms
64 bytes from 192.168.56.30: icmp_seq=2 ttl=64 time=0.401 ms
64 bytes from 192.168.56.30: icmp_seq=3 ttl=64 time=0.230 ms

--- 192.168.56.30 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2029ms
rtt min/avg/max/mdev = 0.230/0.426/0.647/0.171 ms
root@pve-2:~#
```

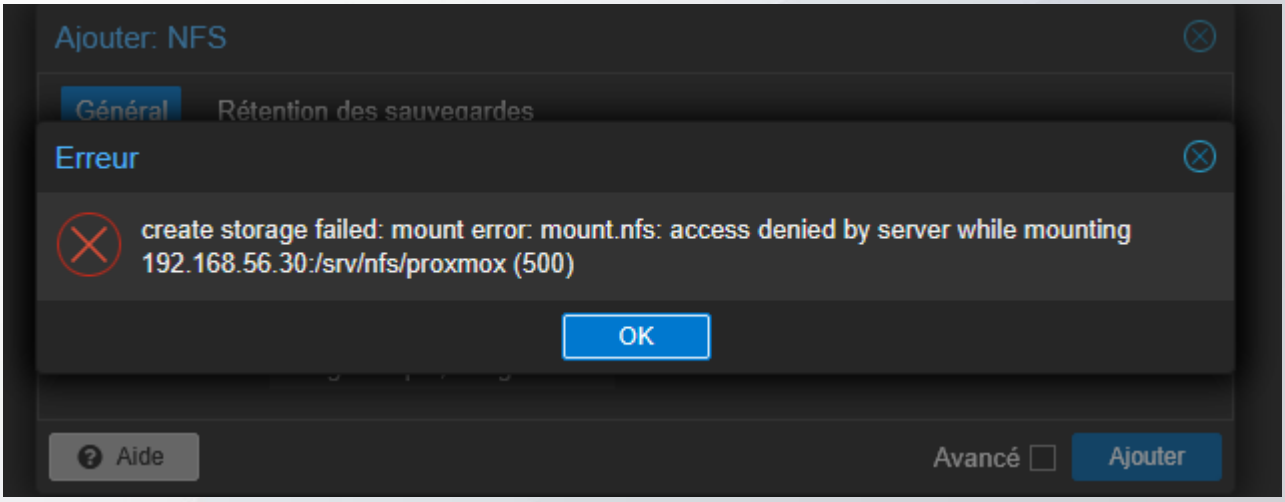
AJOUTER LE NFS (GUI)

Centre de données → Stockage → Ajouter → NFS

- **ID** : nfs-shared
- **Serveur** : 192.168.56.30
- **Export** : /srv/nfs/proxmox
- **Version NFS** : 3 (ou auto)
- **Contenu** : Disk image, ISO, VZDump backup file
- **Nœuds** : pve, pve-2
- **Activer** ✔ → **Ajouter**



Dans le cas où ce message apparait :



L'erreur **"access denied by server"** vient presque toujours des **droits d'export NFS** (le serveur ne permet pas à l'IP du nœud PVE de monter le partage).

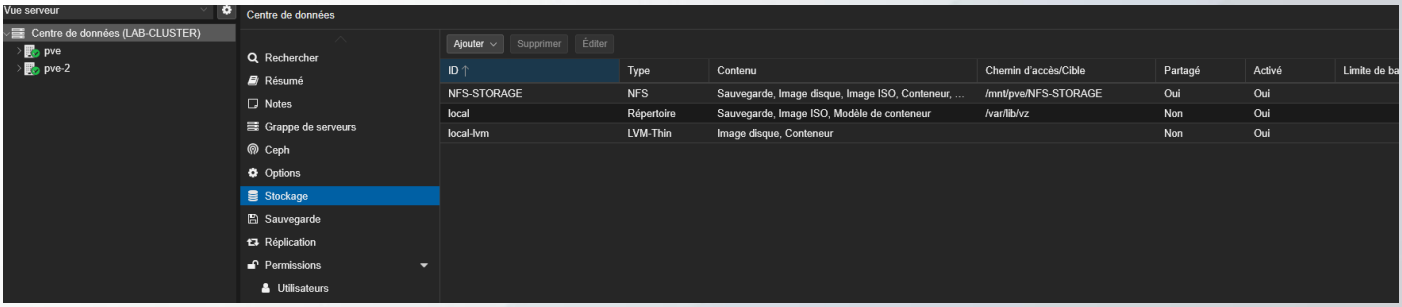
Dans /etc/exports, ajouter la ligne

```
/srv/nfs/proxmox 192.168.56.0/24(rwsync,no_subtree_check,no_root_squash)
```

```
 GNU nano 7.2                                                    exports *
# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/srv/nfs/proxmox 192.168.56.0/24(rw,sync,no_subtree_check,no_root_squash)_
```

```
Exportfs -ra

Exportfs -v
```

```
root@NFS-STORAGE:/etc# exportfs -ra
root@NFS-STORAGE:/etc# exportfs -v
/srv/nfs/proxmox
                192.168.56.0/24(sync,wdelay,hide,no_subtree_check,sec=sys,rw,secure,no_root_squash,no_all_squash)
root@NFS-STORAGE:/etc# systemctl status nfs-server --no-pager
● nfs-server.service - NFS server and services
     Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; preset: enabled)
     Active: active (exited) since Sun 2025-08-24 21:53:32 CEST; 30min ago
    Process: 2285 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCESS)
    Process: 2286 ExecStart=/usr/sbin/rpc.nfsd (code=exited, status=0/SUCCESS)
   Main PID: 2286 (code=exited, status=0/SUCCESS)
        CPU: 12ms

août 24 21:53:32 NFS-STORAGE systemd[1]: Starting nfs-server.service - NFS server and services...
août 24 21:53:32 NFS-STORAGE systemd[1]: Finished nfs-server.service - NFS server and services.
root@NFS-STORAGE:/etc# _
```



AMELIORATIONS POSSIBLES

1) Cluster & disponibilité

- **Ajouter un 3e nœud (pve-3)** : supprimer le besoin du QDevice et apporter une vraie tolérance de panne (quorum naturel à 3).
- **Activer la HA** (ha groups & ressources) : redémarrage automatique des VMs critiques sur l'autre nœud en cas de panne.
- **Deuxième anneau Corosync (ring1)** : lien réseau séparé pour la synchro/heartbeats → moins de risque de split-brain.

2) Stockage

- **Proxmox Backup Server (PBS)** : sauvegardes dédupliquées, compressées, vérifiables (pruning/retention clair), restauration rapide.
- **NFS mieux durci** : passer en **NFSv4**, restreindre aux IP des nœuds, revoir no_root_squash (OK en lab, déconseillé en prod).
- **ZFS (sur la VM NFS)** : miroir à 2 disques ou RAIDZ1 pour tolérer la perte d'un disque et profiter du checksumming.
- **Réseau stockage dédié** (si volumétrie) : isole le trafic NFS, améliore la latence des migrations/sauvegardes.

3) Réseau

- **Segmentation par VLAN** : mgmt, cluster, storage → meilleure isolation et lisibilité.
- **Agrégation (bond/LACP)** pour le mgmt/stockage : augmente la bande passante et supprime le SPOF au niveau NIC.
- **Jumbo frames (MTU 9000)** sur le réseau stockage/cluster (si l'ensemble du chemin le supporte) : baisse l'overhead sur gros flux.
- **Forcer la migration sur le réseau cluster** : évite de saturer le réseau mgmt pendant les live-migrations.

4) Sauvegarde & PRA

- **Plan de sauvegarde** (jobs VZDump/PBS) + **tests de restauration** réguliers : valide la chaîne de secours.
- **Rétention documentée** (ex. GFS : quotidien/hebdo/mensuel) : maîtrise de l'espace disque et des obligations de conservation.
- **Copie hors site** (rclone, NAS distant, cloud froid) : protection contre l'incident local (vol, sinistre).

5) Sécurité

- **Certificats ACME/Let's Encrypt** sur l'UI : élimine les alertes TLS et garantit l'identité des nœuds.
- **PVE Firewall** activé au niveau Datacenter + nœuds : n'autoriser que l'administratif (8006, SSH) et le trafic interne du cluster.
- **Durcir l'accès** : 2FA sur root@pam, clés SSH, désactiver l'authentification par mot de passe en SSH, comptes admins nominatifs.
- **Durcir NFS** : limiter l'export au sous-réseau mgmt, passer en root_squash en dehors du lab, options de montage explicites.

6) Supervision & observabilité

- **Metrics externes** (Prometheus/Grafana, InfluxDB/Telegraf) : visibilité sur CPU/RAM/IOPS/latence, alertes en cas d'anomalie.
- **Syslog centralisé** (Rsyslog/Graylog) : corrélation d'événements, recherche rapide lors d'incidents.

7) Performance & bonnes pratiques

- **Optimisations NFS** (si besoin) : vers=4.1, rsize/wsize adaptés, threads nfsd ajustés selon la charge.
- **NTP activé partout** : horloges alignées (impact direct sur quorum, logs, certificats).

Elias KARISS © EK netsec labs - Tous droits réservés