

# Cluster Proxmox Hyperconvergé

! Il existe une autre documentation beaucoup plus complexe et robuste [ici](#). Je vous déconseille de commencer par celle-ci.

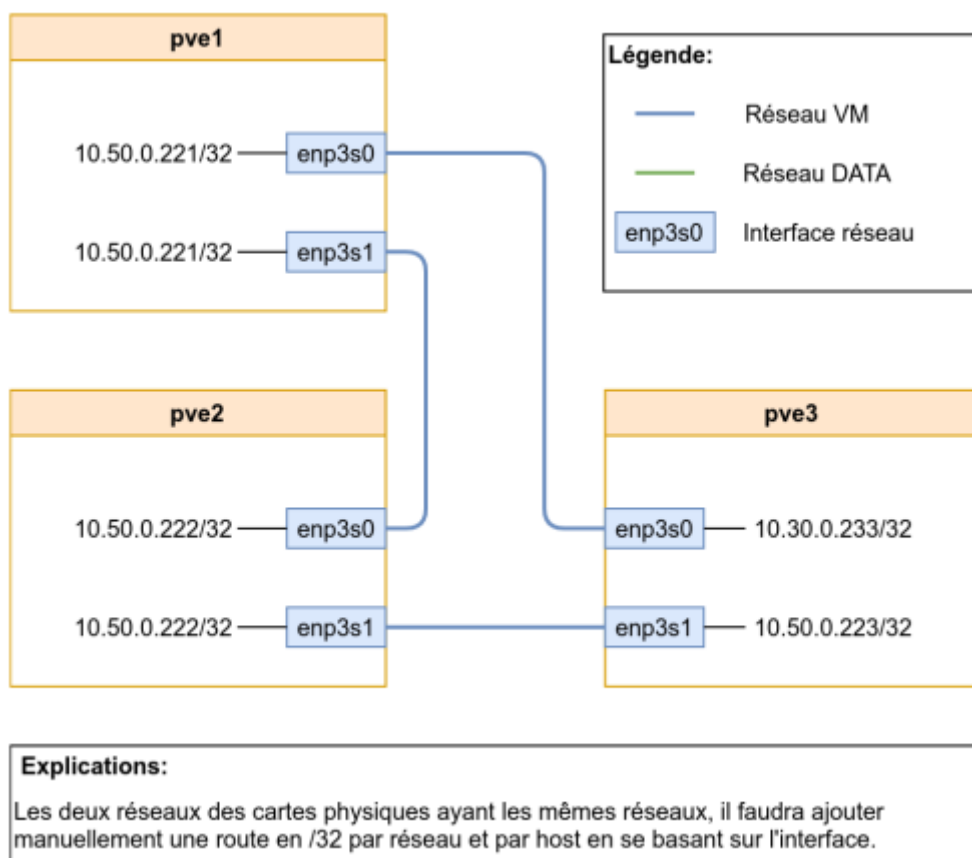
! Cette documentation risque d'être obsolète a la sortie de **pve-network** qui intègre le module SDN. Elle sera réécrite quand le module sortira.

## Informations préliminaires

La solution repose sur du VxLAN (assisté par du MP-BGP) et du CEPH.

Il faut bien comprendre que l'on va réaliser n'est pas natif a Proxmox et va demander de contourner certaines contraintes de l'interface en les configurant en CLI.

L'objectif va être de réaliser cette infrastructure :



avec les spécificités suivante:

- Les IPs dans le réseau 10.50.0.0 sont routés statiquement.
- Les IPs du réseau 10.50.0.0 servent pour les liaisons CEPH, pour les échanges BGP EVPN, pour les liaisons VxLAN, ainsi que pour les liaisons CoroSync.

## Procédure

Pour commencer, nous avons besoin d'installer **ifupdown2** <sup>1)</sup> :

```
# apt install ifupdown2
```

et ensuite il nous faut installer FRR ;

```
# apt install frr
```

Il faut ensuite faire les configurations réseau<sup>2)</sup> :

## pve1

[/etc/network/interfaces](#)

```
auto lo
iface lo inet loopback

auto enp3s0
iface enp3s0 inet static
    address 10.50.0.221/32
    mtu 9000

auto enp3s1
iface enp3s1 inet static
    address 10.50.0.221/32
    mtu 9000

auto vmbri
iface vmbri inet manual
    bridge-ports envxlan1
    bridge-stp off
    bridge-fd 0

auto envxlan1
iface envxlan1
    vxlan-id 1
    vxlan-learning no
```

## pve2

[/etc/network/interfaces](#)

```
auto lo
```

```
iface lo inet loopback

auto enp3s0
iface enp3s0 inet static
    address 10.50.0.222/32
    mtu 9000

auto enp3s1
iface enp3s1 inet static
    address 10.50.0.222/32
    mtu 9000

auto vmbri
iface vmbri inet manual
    bridge-ports envxlan1
    bridge-stp off
    bridge-fd 0

auto envxlan1
iface envxlan1
    vxlan-id 1
    vxlan-learning no
```

## pve3

[/etc/network/interfaces](#)

```
auto lo
iface lo inet loopback

auto enp3s0
iface enp3s0 inet static
    address 10.50.0.223/32
    mtu 9000

auto enp3s1
iface enp3s1 inet static
    address 10.50.0.223/32
    mtu 9000

auto vmbri
iface vmbri inet manual
    bridge-ports envxlan1
    bridge-stp off
    bridge-fd 0

auto envxlan1
iface envxlan1
```

```
vxlan-id 1
vxlan-learning no
```

puis on recharge sur chaque nœud la configuration réseau :

```
# ifreload -a
```

Maintenant, il va falloir faire des manipulations sur chaque nœud avec certaines adaptations. Voici :

## pve1

Entrez en ligne de commande **frr** :

```
# vtysh
```

ensuite entrez en mode configuration :

```
pve1# conf t
```

On va définir les routes statiques afin d'épargner un peu la découverte ARP inutile :

```
pve1(config)# ip route 10.50.0.223/32 enp3s0
pve1(config)# ip route 10.50.0.222/32 enp3s1
```

ensuite on va dans la configuration **bgp**

```
pve1(config)# router bgp 65000
```

on applique les configurations usuel:

```
pve1(config-router)# bgp router-id 10.50.0.221
pve1(config-router)# no bgp default ipv4-unicast
pve1(config-router)# neighbor pg-evpn peer-group
pve1(config-router)# neighbor pg-evpn remote-as 65000
pve1(config-router)# neighbor pg-evpn timers 5 15
```

Puis la configuration EVPN :

```
pve1(config-router)# address-family l2vpn evpn
pve1(config-router-af)# neighbor pg-evpn activate
pve1(config-router-af)# advertise-all-vni
pve1(config-router-af)# advertise-default-gw
pve1(config-router-af)# exit
```

Ensuite on configure les neighbors :

```
pve1(config-router)# neighbor 10.50.0.222 peer-group pg-evpn
pve1(config-router)# neighbor 10.50.0.223 peer-group pg-evpn
```

Ensuite on clear les sessions BGP :

```
pve1(config-router-af)# end
pve1# clear bgp l2vpn evpn *
```

Après avoir attendu quelques secondes on peut vérifier si les sessions BGP sont actives :



Une fois validé, on enregistre et on quitte :

```
pve1# write memory
pve1# exit
```

## pve2

Entrez en ligne de commande **frr** :

```
# vtysh
```

ensuite entrez en mode configuration :

```
pve2# conf t
```

On va définir les routes statiques afin d'épargner un peu la découverte ARP inutile :

```
pve2(config)# ip route 10.50.0.221/32 enp3s0
pve2(config)# ip route 10.50.0.223/32 enp3s1
```

ensuite on va dans la configuration **bgp**

```
pve2(config)# router bgp 65000
```

on applique les configurations usuel:

```
pve2(config-router)# bgp router-id 10.50.0.222
pve2(config-router)# no bgp default ipv4-unicast
pve2(config-router)# neighbor pg-evpn peer-group
pve2(config-router)# neighbor pg-evpn remote-as 65000
pve2(config-router)# neighbor pg-evpn timers 5 15
```

Puis la configuration EVPN :

```
pve2(config-router)# address-family l2vpn evpn
pve2(config-router-af)# neighbor pg-evpn activate
pve2(config-router-af)# advertise-all-vni
pve2(config-router-af)# advertise-default-gw
pve2(config-router-af)# exit
```

Ensuite on configure les neighbors :

```
pve2(config-router)# neighbor 10.50.0.221 peer-group pg-evpn
pve2(config-router)# neighbor 10.50.0.223 peer-group pg-evpn
```

Ensuite on clear les sessions BGP :

```
pve2(config-router-af)# end
pve2# clear bgp l2vpn evpn *
```

Après avoir attendu quelques secondes on peut vérifier si les sessions BGP sont actives :



Une fois validé, on enregistre et on quitte :

```
pve2# write memory
pve2# exit
```

## pve3

Entrez en ligne de commande **frr** :

```
# vtysh
```

ensuite entrez en mode configuration :

```
pve3# conf t
```

On va définir les routes statiques afin d'épargner un peu la découverte ARP inutile :

```
pve3(config)# ip route 10.50.0.221/32 enp3s0
pve3(config)# ip route 10.50.0.222/32 enp3s1
```

ensuite on va dans la configuration **bgp**

```
pve3(config)# router bgp 65000
```

on applique les configurations usuel:

```
pve3(config-router)# bgp router-id 10.50.0.223
pve3(config-router)# no bgp default ipv4-unicast
pve3(config-router)# neighbor pg-evpn peer-group
pve3(config-router)# neighbor pg-evpn remote-as 65000
pve3(config-router)# neighbor pg-evpn timers 5 15
```

Puis la configuration EVPN :

```
pve3(config-router)# address-family l2vpn evpn
pve3(config-router-af)# neighbor pg-evpn activate
pve3(config-router-af)# advertise-all-vni
pve3(config-router-af)# advertise-default-gw
pve3(config-router-af)# exit
```

Ensuite on configure les neighbors :

```
pve3(config-router)# neighbor 10.50.0.221 peer-group pg-evpn
pve3(config-router)# neighbor 10.50.0.222 peer-group pg-evpn
```

Ensuite on clear les sessions BGP :

```
pve3(config-router-af)# end
pve3# clear bgp l2vpn evpn *
```

Après avoir attendu quelques secondes on peut vérifier si les sessions BGP sont actives :



Une fois validé, on enregistre et on quitte :

```
pve3# write memory
pve3# exit
```

## Configuration du Cluster

On va créer le cluster **en CLI**. Pour ça, connectez vous sur le premier noeud, et tapez la commande :

```
# pvecm create pve-switchless
```

une fois fait, il faudra vous connecter sur les deux autres noeuds et taper la commande suivante en vous laissant guider <sup>3)</sup>:

```
# pvecm add 10.50.0.221
```



Il faut ensuite installer Ceph sur chaque nœud **en CLI** :

```
# pveceph install
```

Puis sur l'un des nœud, initialiser le cluster :

```
# pveceph init --network 10.50.0.0/24
```

Il va falloir modifier le fichier **/etc/pve/ceph.conf** pour :

- Retirer les deux lignes suivantes :

```
cluster_network = 10.50.0.0/24
public_network = 10.50.0.0/24
```

- Ajouter ceci a la fin du fichier :

```
[mon.pve1]
host = pve1
mon_addr = 10.50.0.221
public_addr = 10.50.0.221
public_bind_addr = 10.50.0.221

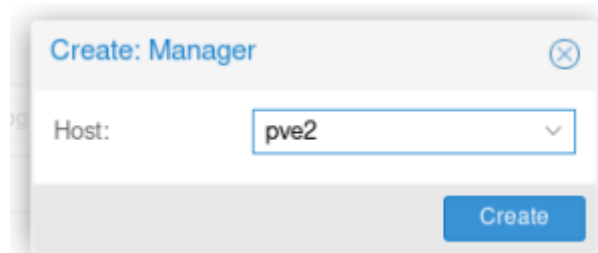
[mon.pve2]
host = pve2
mon_addr = 10.50.0.222
public_addr = 10.50.0.222
public_bind_addr = 10.50.0.222

[mon.pve3]
host = pve3
mon_addr = 10.50.0.223
public_addr = 10.50.0.223
public_bind_addr = 10.50.0.223
```

puis initialiser sur chaque nœud, le monitor ceph en y adaptant l'IP <sup>4)</sup> :

```
# pveceph createmon --mon-address 10.50.0.221
```

Une fois fait, on peut repasser sur l'interface pour ajouter les managers manquant :



Puis les OSD sur chaque nœud :

hdd      bluestore      up ● / in ●      14.2.4      0.01859      1.00      5.27      19.00 GiB

### Create: Ceph OSD

Disk:  DB Disk:

DB size (GiB):

---

Encrypt OSD:  WAL Disk:

WAL size (GiB):

Note: Ceph is not compatible with disks backed by a hardware RAID controller. For details see [the reference documentation](#).

Advanced

Et pour finir le pool de données :

### Create: Ceph Pool

Name:

Size:

Min. Size:

Crush Rule:

pg\_num:

Add as Storage:

On peut aussi créer un pool CephFS si nécessaire en créant les MDS :

### Create: Metadata Servers

Host:

Puis le pool CephFS :

Host	Status	Address
nx01	up:standby	10.50.0.221:6827/677463314
nx02	up:standby	10.50.0.222:6825/3986094185
nx03	up:standby	10.50.0.223:6825/1343851790

### Create: Ceph FS

Name:

Placement Groups:

Add as Storage:

[Help](#) [Create](#)

- 1) Attention, toutes les configurations réseau seront déchargées, donc vous allez perdre la main si vous êtes en SSH
- 2) Les interfaces vxlan doivent obligatoirement commencer par le préfix "en" et termine par un numéro
- 3) ,
- 4) A lancer nœud par nœud, pas en même temps

From: <https://wiki.virtit.fr/> - **VirtIT**

Permanent link: <https://wiki.virtit.fr/doku.php/kb:linux:proxmox:hyperconverged-proxmox?rev=1582308895>

Last update: **2020/02/21 18:14**

