Cluster Proxmox Hyperconvergé

Il existe une autre documentation beaucoup plus complexe et robuste ici. Je vous déconseille de commencer par celle-ci.

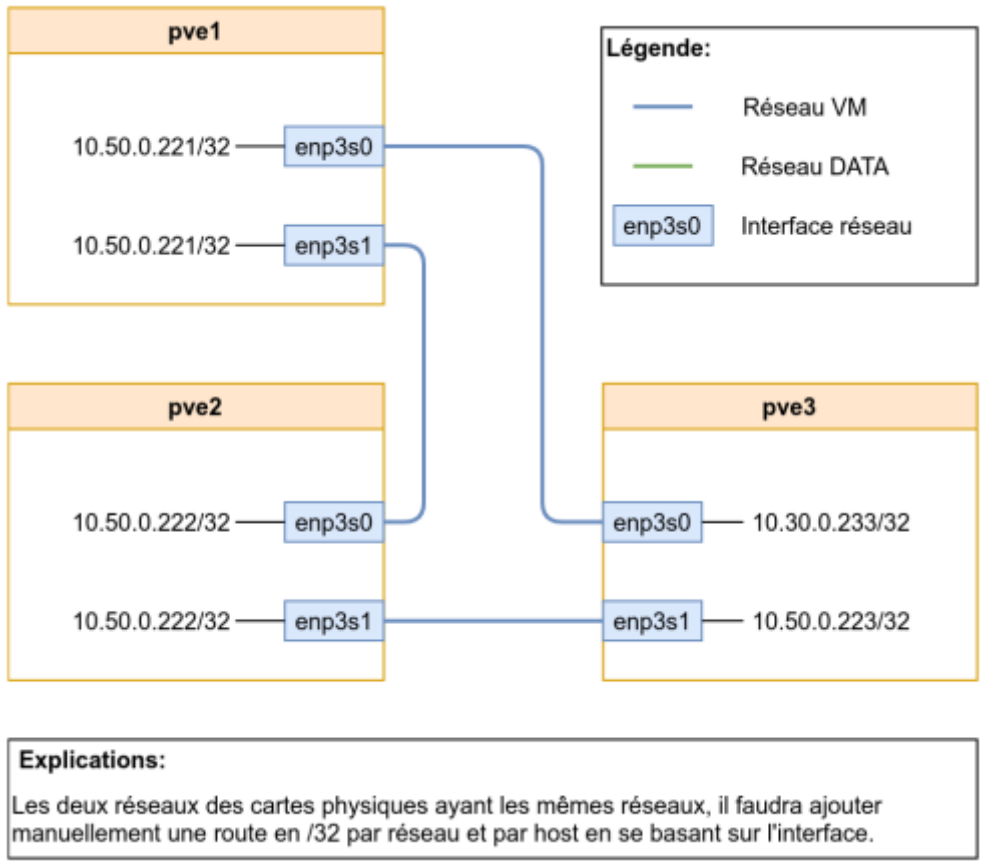
Cette documentation risque d'être obsolète a la sortie de **pve-network** qui intègre le module SDN. Elle sera réécrite quand le module sortira.

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 routées statiquement.
- Les IPs du réseau 10.50.0.0 servent pour les liaisons CEPH, pour les echanges BGP EVPN, pour les liaisons VxLAN, ainsi que pour les laisons CoroSync.

Procédure

Pour commencer, nous avons besoin d'installer **ifupdown2** [1] :

```
# apt install ifupdown2
```

et ensuite il nous faut installer FRR ;

```
# apt install frr
```

Il faut ensuite faire les configurations réseau[2] :

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 vmbr1
iface vmbr1 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 vmbr1
iface vmbr1 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 vmbr1
iface vmbr1 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 peux vérifier si les sessions BGP sont actives :

```
pve1# sh bgp summary

L2VPN EVPN Summary:
BGP router identifier 10.50.0.221, local AS number 65000 vrf-id 0
BGP table version 0
RIB entries 5, using 920 bytes of memory
Peers 2, using 41 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down
State/PfxRcd
10.50.0.222     4      65000      79      85        0    0    0 00:01:52
1
10.50.0.223     4      65000      13      13        0    0    0 00:00:46
1

Total number of neighbors 2
```

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 peux vérifier si les sessions BGP sont actives :

```
pve2# sh bgp summary

L2VPN EVPN Summary:
BGP router identifier 10.50.0.222, local AS number 65000 vrf-id 0
BGP table version 0
RIB entries 5, using 920 bytes of memory
Peers 2, using 41 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down
State/PfxRcd
10.50.0.221     4      65000      79      81        0    0    0 00:01:52
1
10.50.0.223     4      65000      13      18        0    0    0 00:00:44
1

Total number of neighbors 2
```

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 peux vérifier si les sessions BGP sont actives :

```
pve3# sh bgp summary

L2VPN EVPN Summary:
BGP router identifier 10.50.0.223, local AS number 65000 vrf-id 0
BGP table version 0
RIB entries 5, using 800 bytes of memory
Peers 2, using 41 KiB of memory
Peer groups 1, using 64 bytes of memory

Neighbor        V         AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down
State/PfxRcd
10.50.0.221     4      65000      13      13        0    0    0 00:00:46
1
10.50.0.222     4      65000      14      14        0    0    0 00:00:44
1

Total number of neighbors 2
```

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 [3)]:

```
# pvecm add 10.50.0.221
```

```
# pvecm add 10.50.0.221
Please enter superuser (root) password for '10.50.0.221': ************
Establishing API connection with host '10.50.0.221'
The authenticity of host '10.50.0.221' can't be established.
X509 SHA256 key fingerprint is
38:5F:E9:43:F7:58:18:40:70:93:57:0F:CE:DD:CB:F1:CB:88:DD:CF:85:5A:52:5C:FA:5
6:45:C3:38:B0:E6:D1.
Are you sure you want to continue connecting (yes/no)? yes
Login succeeded.
Request addition of this node
Join request OK, finishing setup locally
```

```
stopping pve-cluster service
backup old database to '/var/lib/pve-
cluster/backup/config-1582313226.sql.gz'
waiting for quorum...OK
(re)generate node files
generate new node certificate
merge authorized SSH keys and known hosts
generated new node certificate, restart pveproxy and pvedaemon services
successfully added node 'pve2' to cluster.
```

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

```
# pveceph install
```

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

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

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

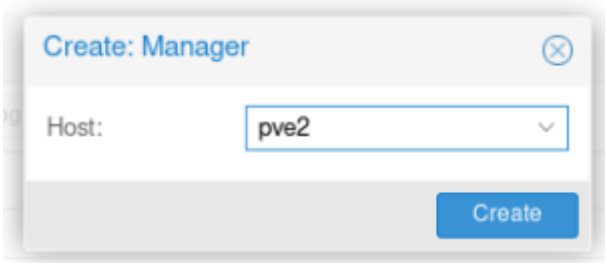
[mon.pve2]
     host = pve2
     mon_addr = 10.50.0.222
     public_addr = 10.50.0.222
     public_bind_addr = 10.50.0.222

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

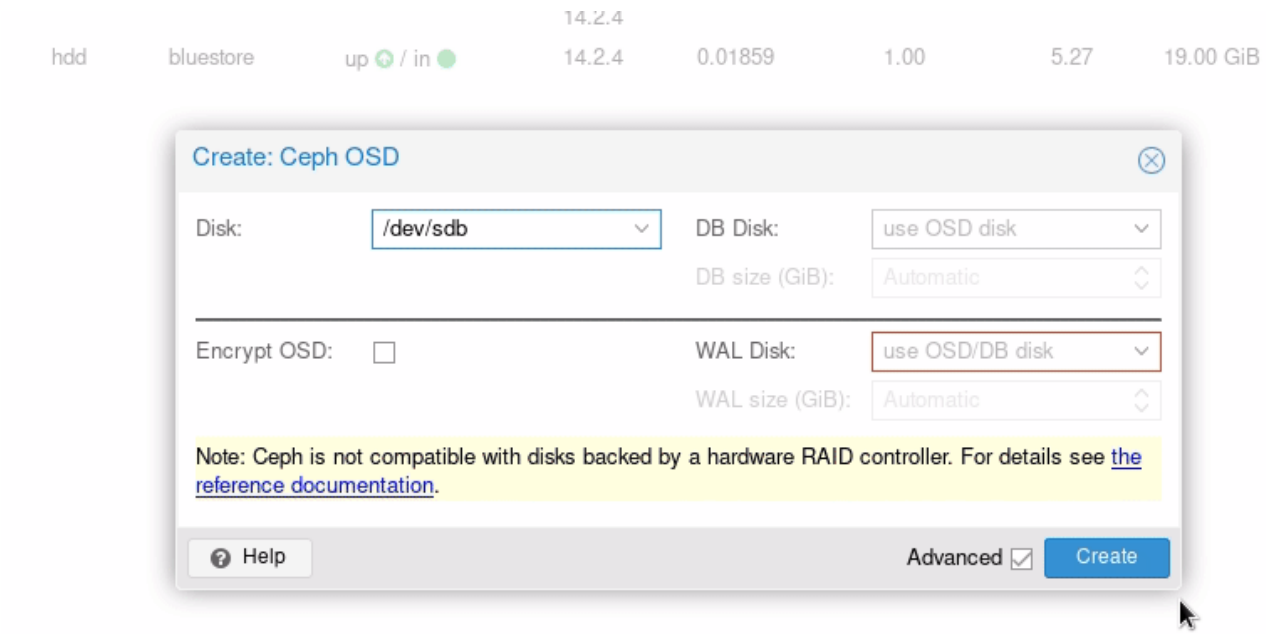
puis initialiser sur chaque noeud, le monitor ceph en y adaptant l'IP [4] :

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

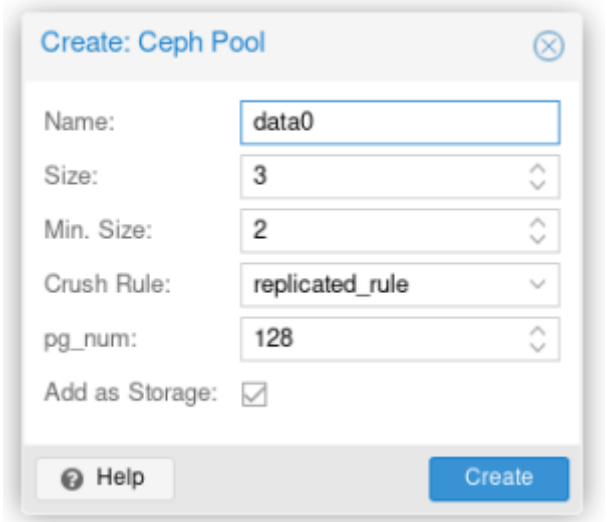
Une fois fait, on peut repasser sur l'interface pour ajouter les managers manquant :



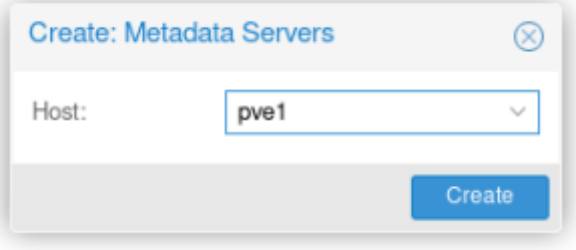
Puis les OSD sur chaque nœud :



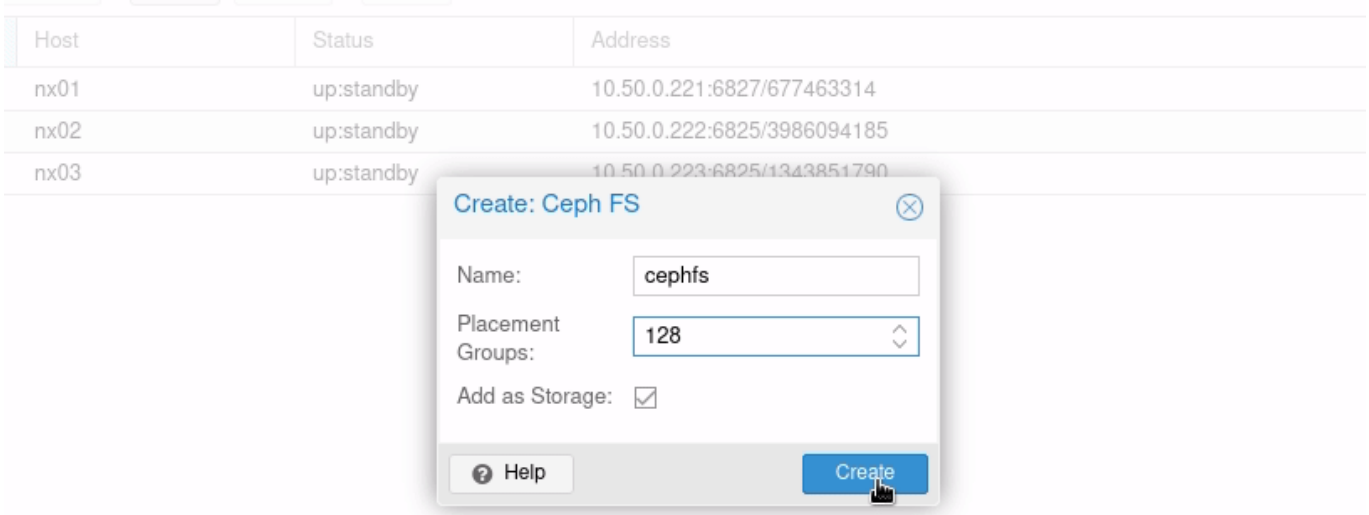
Et pour finir le pool de données :



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



Puis le pool CephFS :



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

Last update: **2020/02/21 20:10**