Compte Rendu T-CLO-901

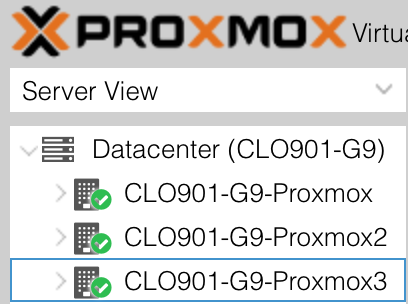
Groupe 9

Zakaria Chahbi – Julien Legay – Luc Soriano – Nathan Lagrange

# Installation Proxmox

Sur les trois serveurs Online.net, nous avons installé Proxmox VE 6.2

Nous les avons ensuite définis comme un cluster



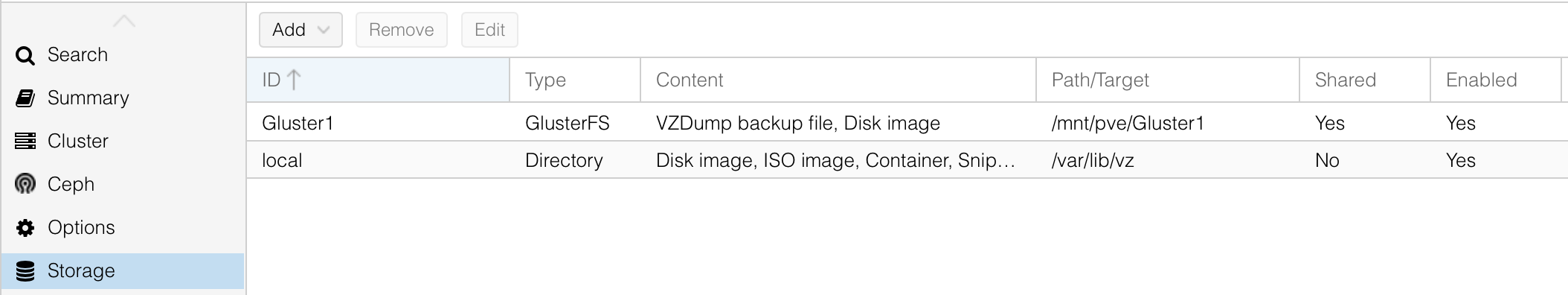
# Installation de GlusterFS

Nous avons installé un stockage partagé utilisant GlusterFS.

Chaque hyperviseur dispose d’une partition GlusterFS (formaté en ext4) de 100Go.

Cela permet donc d’avoir un stockage de 100Go, qui soit un mirroir sur chaque serveur.

Nous hébergeons sur ce stockage les VMs ainsi que les snapshots.



# Migration VM à chaud

Il est possible de migrer des VMs d’un hyperviseur a un autre à chaud, c’est-à-dire sans coupure sur la VM.

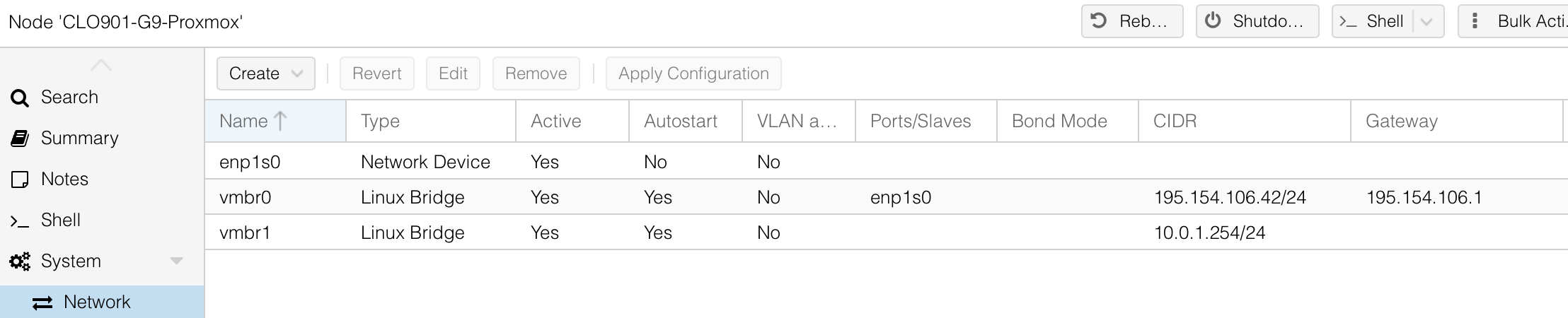


Par exemple ici, on peut voir que la coupure n’a duré que 66 ms, ce qui est très correct.

Afin que chaque VM puisse avoir accès à internet après une migrations, nous avons dû ruser en définissant le même réseau privé pour chaque hyperviseur (à savoir 10.0.1.0/24).

# Accès à internet des VMs

Pour pouvoir avoir donner l’accès à internet aux VMs, nous avons créé une carte réseau (vmbr1) sur chaque hyperviseur ayant pour IP : 10.0.1.254/24.



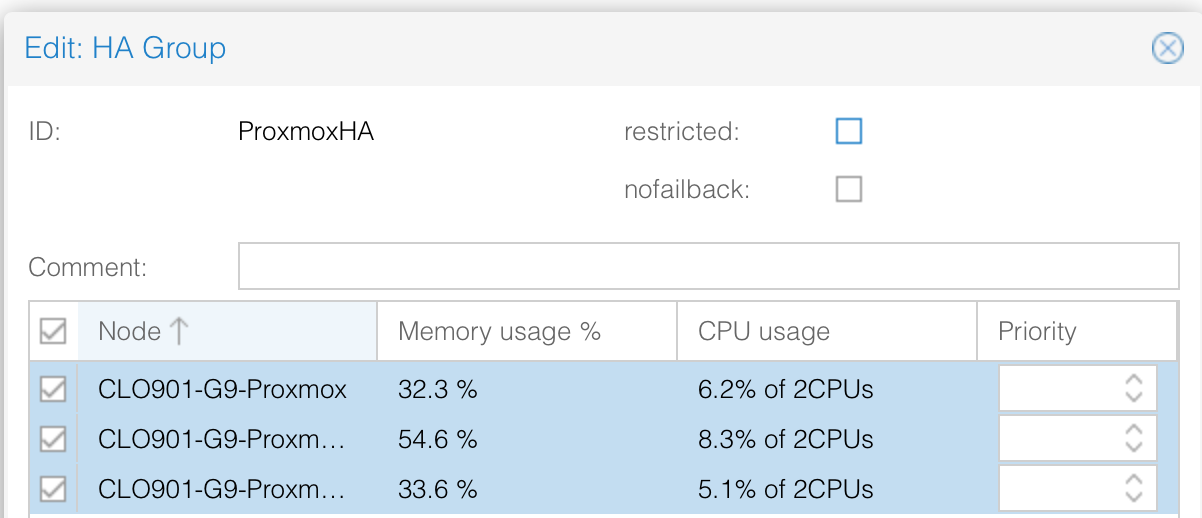
Nous avons ensuite utilisé des règles de NAT via iptables pour faire transiter les paquets vers internet (modification de l’interface dans /etc/network/interfaces) :



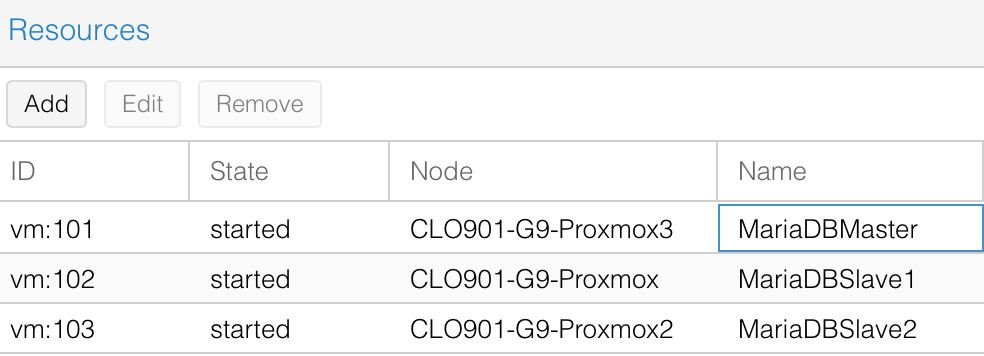
# High Availability des VMs

Nous avons configuré le HA sur Proxmox.

En effet, nous avons créé un groupe qui contient les trois hyperviseurs :



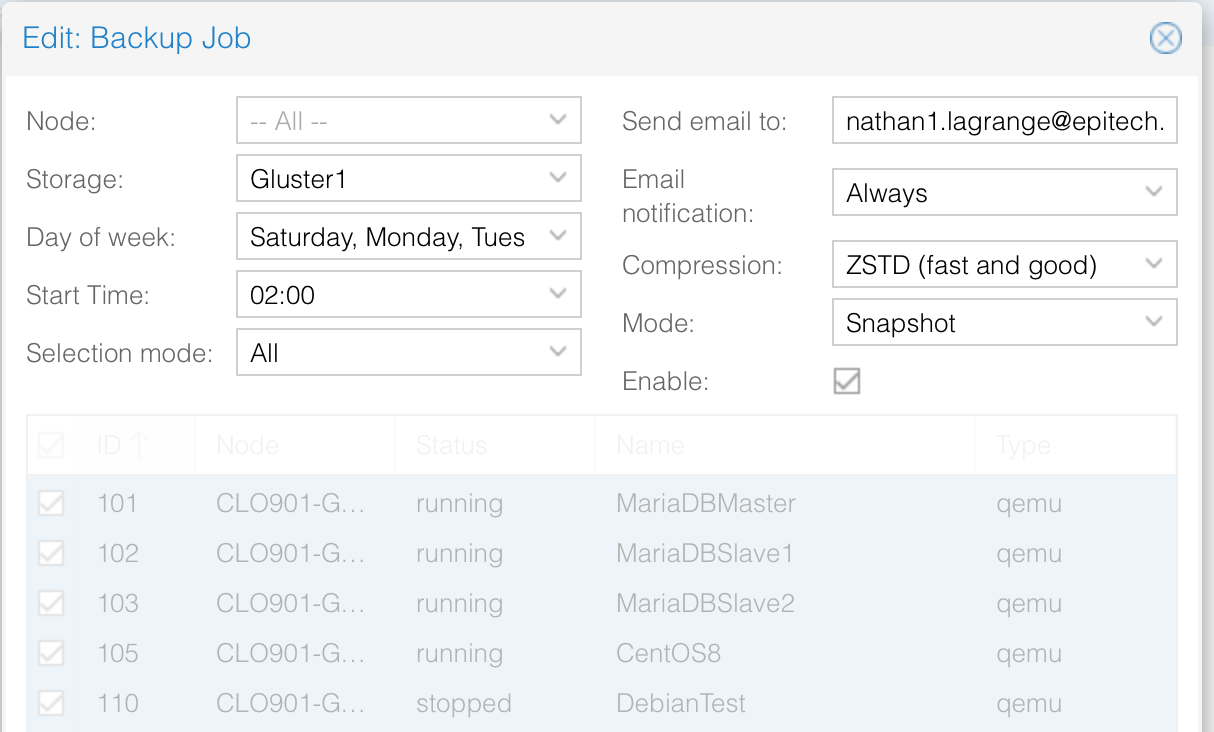
Puis nous avons ajouté les VMs que nous voulons sécuriser (à savoir les 3 VMs mariadb-server, client1 et client2) :



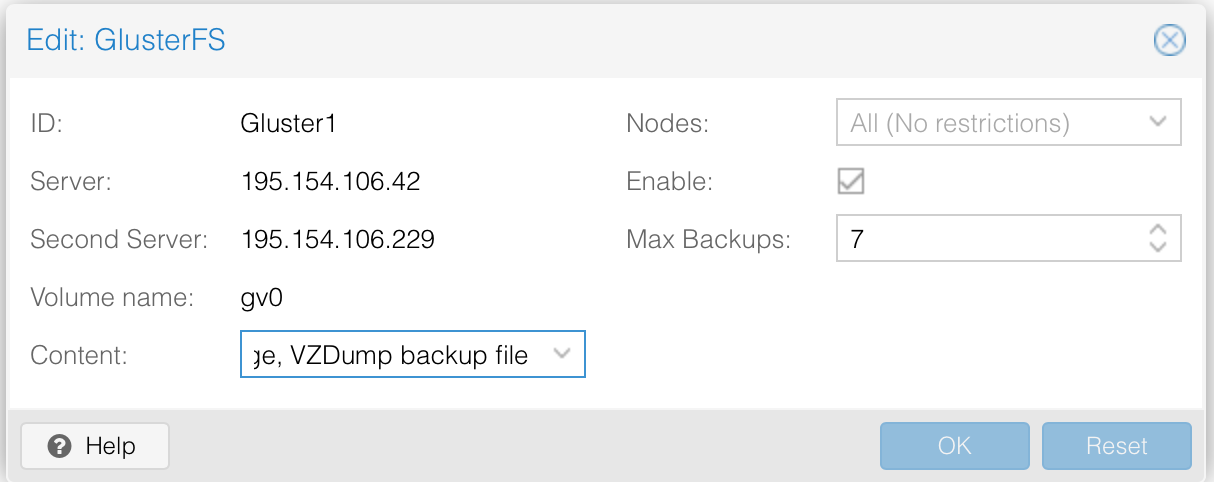
Donc lorsqu’un hyperviseur n’est plus accessible, la VM qu’il héberge va être migrée sur un autre hyperviseur et sera automatiquement rallumée.

# Sauvegardes des VMs

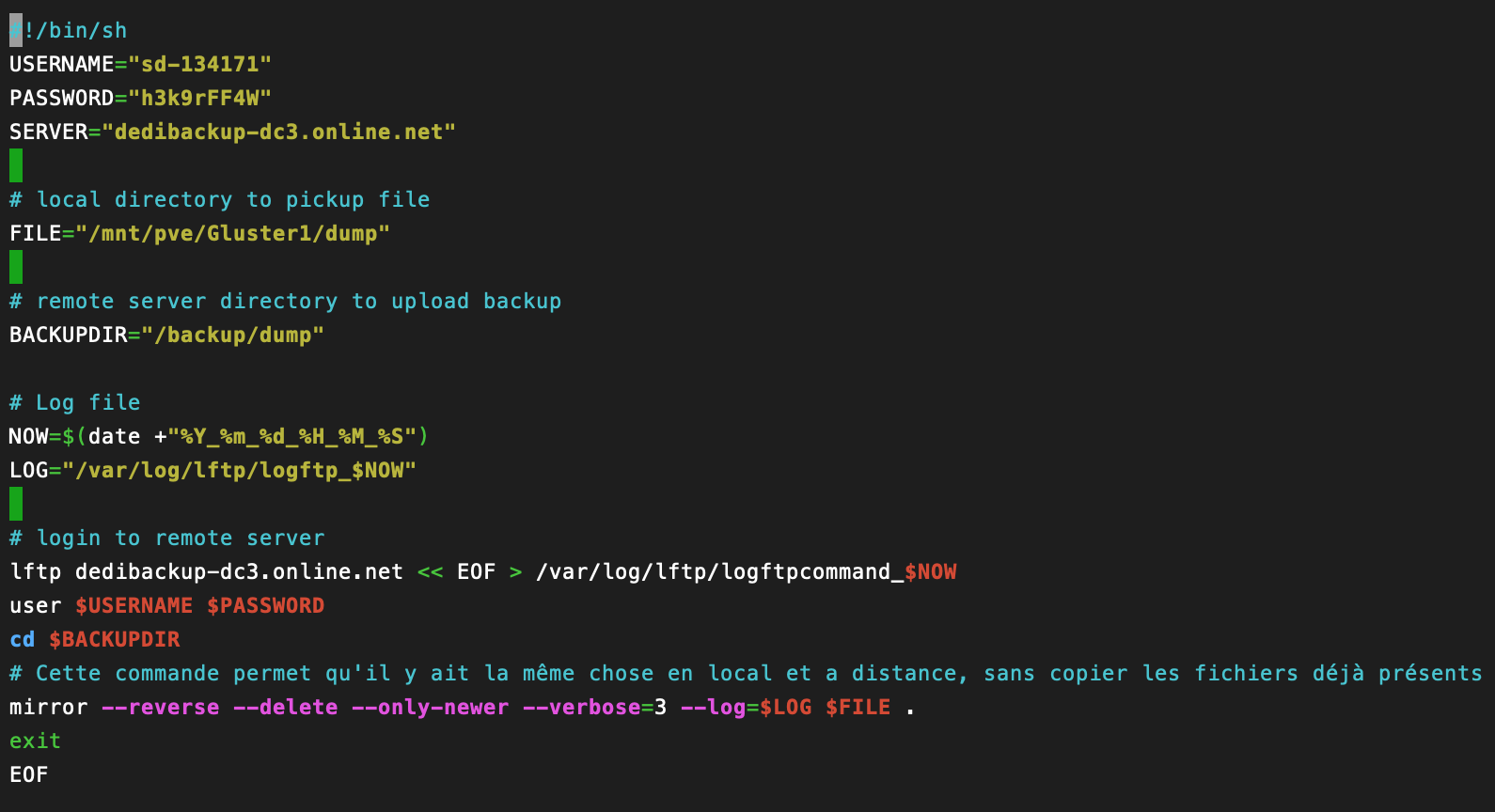
Nous avons mis un place une sauvegarde tous les jours de toutes les VMs à 2h du matin.



La rétention des fichiers est définie sur 7 jours :



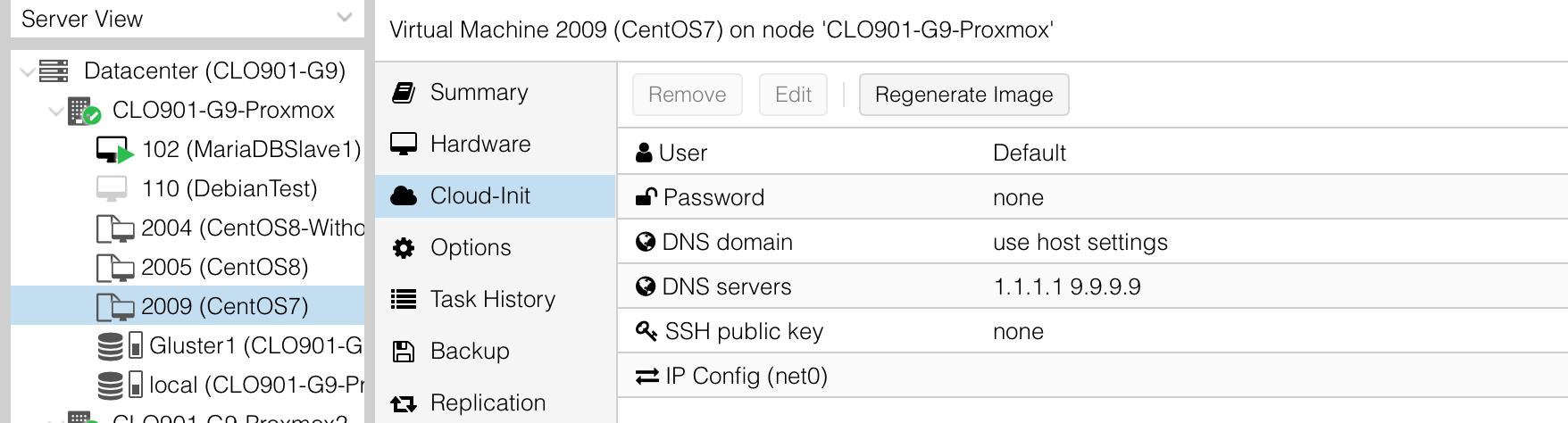
De plus, étant donné que chaque serveur est fourni avec un espace FTP de 100Go, ces fichiers de backups sont envoyés sur chaque FTP tous les jours à 3h du matin grâce à un script bash exécuté par cron :



# Installation des VMs avec Cloud-init

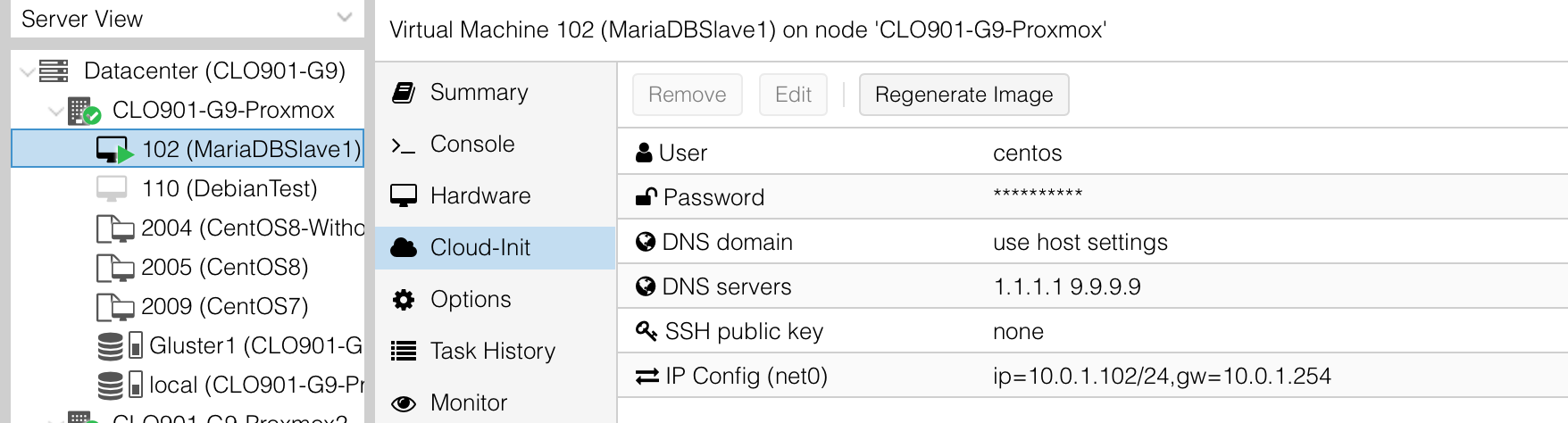
Nous avons utilisé Cloud-init pour déployer les 3 VMs Centos 7 à partir de l’image cloud (<https://cloud.centos.org/centos/7/images/>)

Pour cela, nous avons créé une VM template :



Lorsque l’on souhaite créer une nouvelle VM, il suffit de cloner celle-ci, de choisir un user, un password, et d’entrer sa configuration réseau.

Par exemple pour client1 :



Attention par contre, car il y a un problème avec les entrées de serveur DNS dans le fichier /etc/resolv.conf

En effet, dans l’image Cloud de CentOS 7, il y a un DNS déjà configuré qu’il faut supprimer (<https://bugzilla.redhat.com/show_bug.cgi?id=1545850>)

# Communication entre les VMs

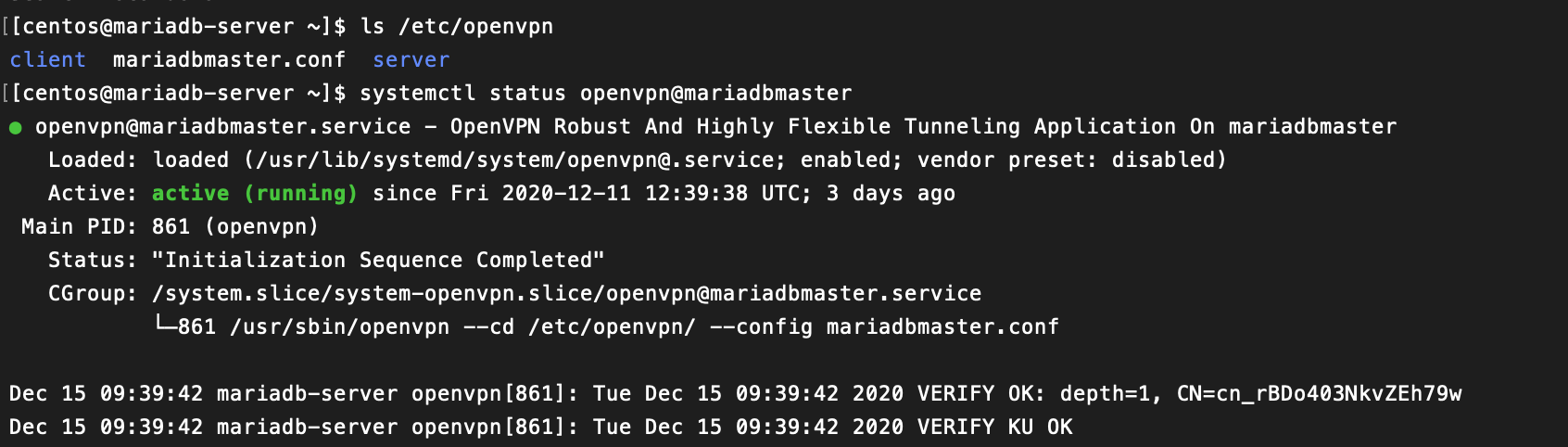
Pour pouvoir faire communiquer les VMs entre elles, nous avons dû installer un serveur OpenVPN (sur une VM Debian dans Azure).

<https://www.cyberciti.biz/faq/debian-10-set-up-openvpn-server-in-5-minutes/>

Nous avons généré 3 fichiers clients que nous avons placé sur les 3 VMs.

Nous avons ensuite installé sur les 3 VMs OpenVPN client avec une IP fixe pour chaque VM : 10.8.0.101 pour mariadb-server, 10.8.0.102 pour client1 et 10.8.0.103 pour client2.

C’est un service qui se lance au boot qui se charge de la connexion :



Les 3 VMs peuvent donc communiquer via un tunnel openVPN.

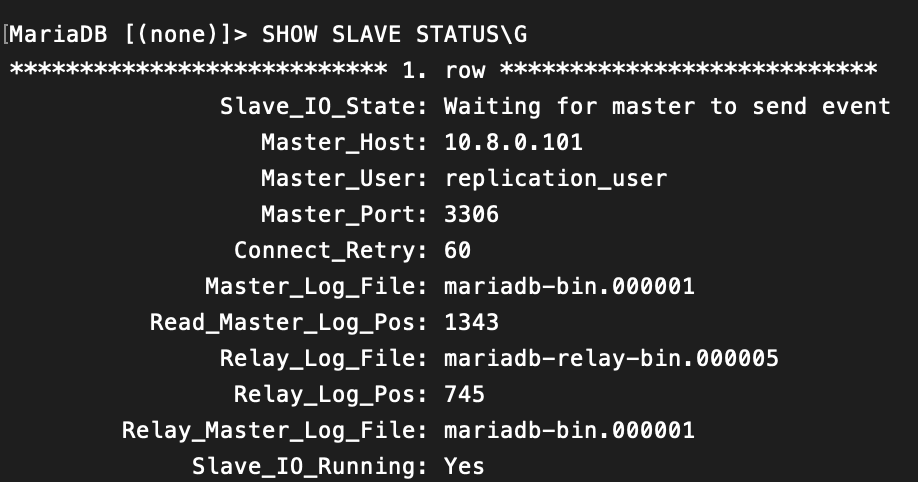
# Installation de MariaDB

Nous avons installé Mariadb sur les 3 serveurs.

<https://hevodata.com/learn/mariadb-replication-easy-steps/#master>

<https://www.journaldev.com/29314/set-up-mariadb-master-slave-replication-centos>

Nous avons ensuite relié mariadb-server à client1 et client2 qui sont définis comme slave :



Maintenant, lorsque nous faisons une modification de la DB sur mariadb-server, elle est automatiquement répliquée sur les deux autres MariaDB.

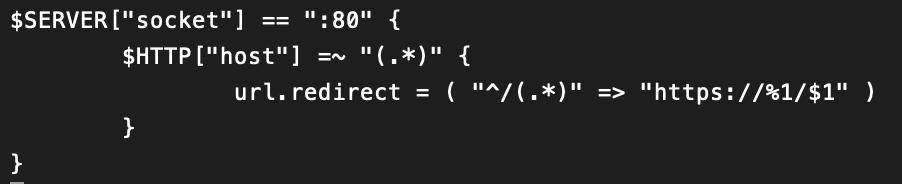
# En plus : modification du nom de domaine

Par défaut les URL des interfaces de Proxmox sont sd-xxxxx.dedibox.fr:8006 avec une erreur de certificat.

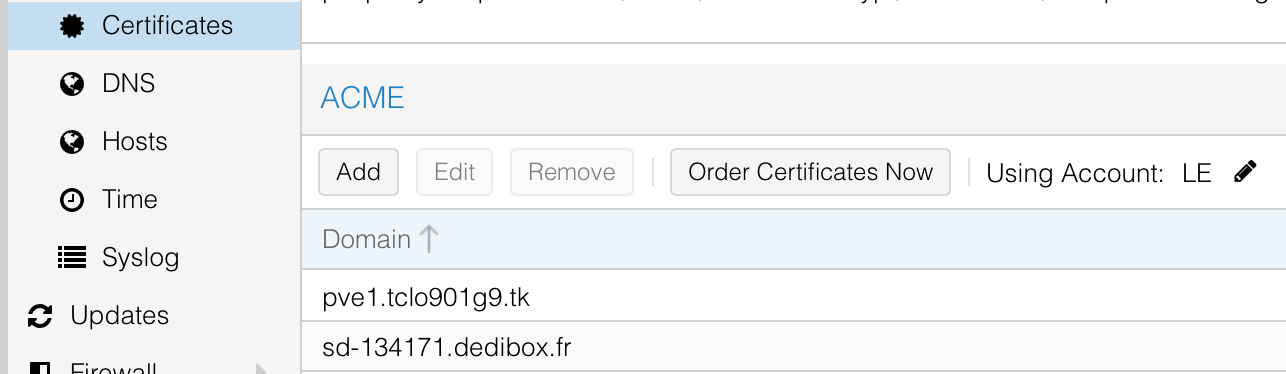
Nous avons donc pris un nom de domaine gratuit et fait des redirections (iptables + lighttpd) pour pouvoir rediriger <https://sd-xxxxx.dedibox.fr:8006> vers <https://pve1.tclo901g9.tk> (iptables)



De plus <http://pve1.tclo901g9.tk> redirige vers <https://pve1.tclo901g9.tk> (lighttpd) :



Grâce à Proxmox, il est possible d’installer un certificat Let’s Encrypt :



Pour l’hyperviseur 1 : <https://pve1.tclo901g9.tk>

Pour l’hyperviseur 2 : <https://pve2.tclo901g9.tk>

Pour l’hyperviseur 3 : <https://pve3.tclo901g9.tk>