Rapport Technique : Infrastructure de messagerie sécurisée et haute disponibilité

- Entreprise : **Evil Corp**

Membres :

- MEVENGUE Franck
- Lilia Halfaou
- Arthur Nguedia
- Karimou Aweni

1. Contexte et objectifs du projet :

1.1- Présentation de l'entreprise :

Evil Corp est une entreprise dont l'ensemble des processus internes repose sur l'utilisation du courrier électronique. Les communications internes, la prospection commerciale, les comptes rendus de réunion et même les processus décisionnels reposent sur cette infrastructure. En raison de l'importance critique de ces

communications, l'entreprise souhaite mettre en place une infrastructure de messagerie sécurisée et hautement disponible, accompagnée d'un système de sauvegarde fiable capable de préserver l'intégrité et la confidentialité des courriels.

La solution devra garantir :

1. La sécurisation des communications
2. La confidentialité des données
3. La haute disponibilité du stockage de sauvegarde
4. La protection contre la perte de données
5. La gestion automatisée des certificats et mises à jour

1.2- Objectifs techniques :

Le projet vise à concevoir et déployer une infrastructure capable de fournir plusieurs services critiques.

Les principaux objectifs sont les suivants :

Mise en place d'un serveur de messagerie sécurisé

Le système devra permettre :

- L'envoi de messages via le protocole SMTP
- La consultation des emails via le protocole IMAP
- La sécurisation de ces protocoles via SSL/TLS

Gestion des certificats

Une autorité de certification interne devra être mise en place afin de générer les certificats nécessaires pour :

- Le serveur de messagerie
- Les communications sécurisées
- Les infrastructures VPN

Mise en place d'une infrastructure de sauvegarde

Les emails devront être sauvegardés automatiquement vers un site distant.

Mise en œuvre d'un tunnel IPSec

Afin d’assurer la confidentialité des données, les sauvegardes devront être transmises via un tunnel chiffré.

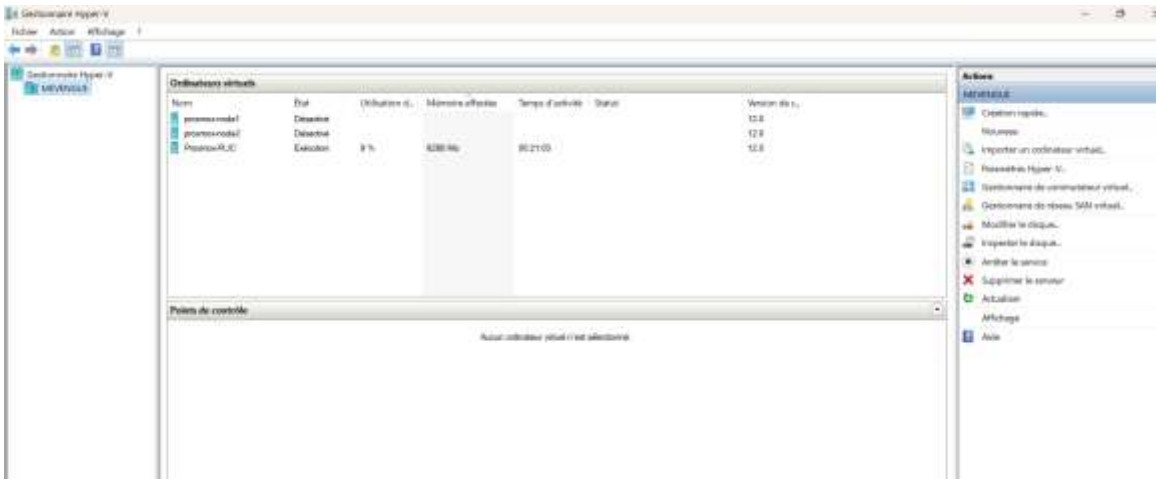
Mise en œuvre d’un stockage hautement disponible

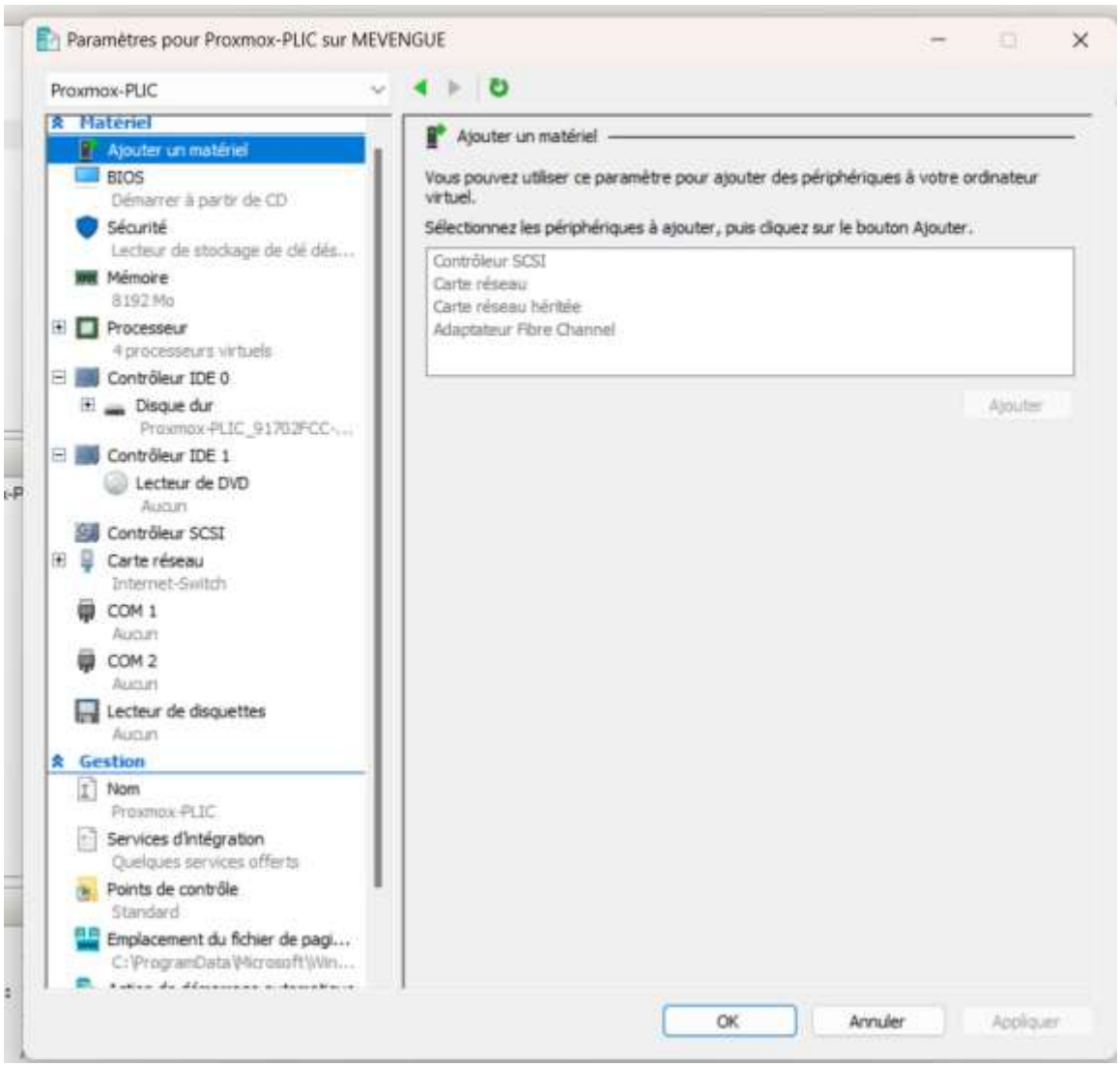
Le stockage de sauvegarde devra être assuré par deux serveurs synchronisés via DRBD afin de garantir la continuité du service.

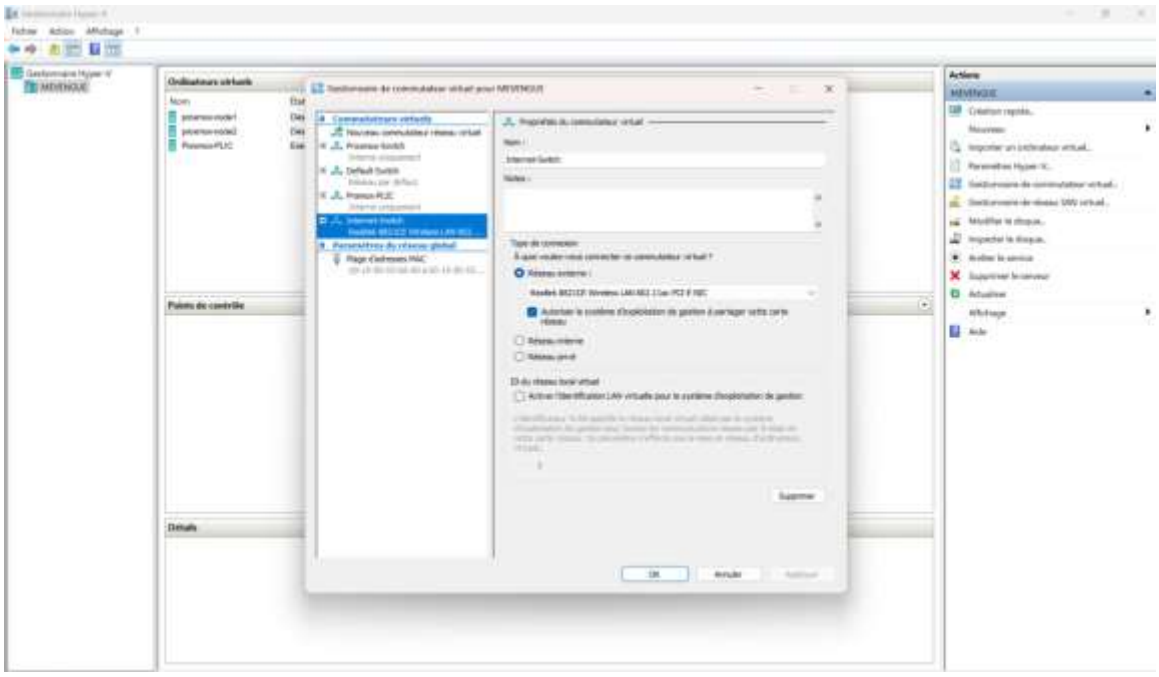
Automatisation via Puppet

Le déploiement du certificat racine ainsi que les mises à jour de sécurité devront être automatisés à l’aide de Puppet.

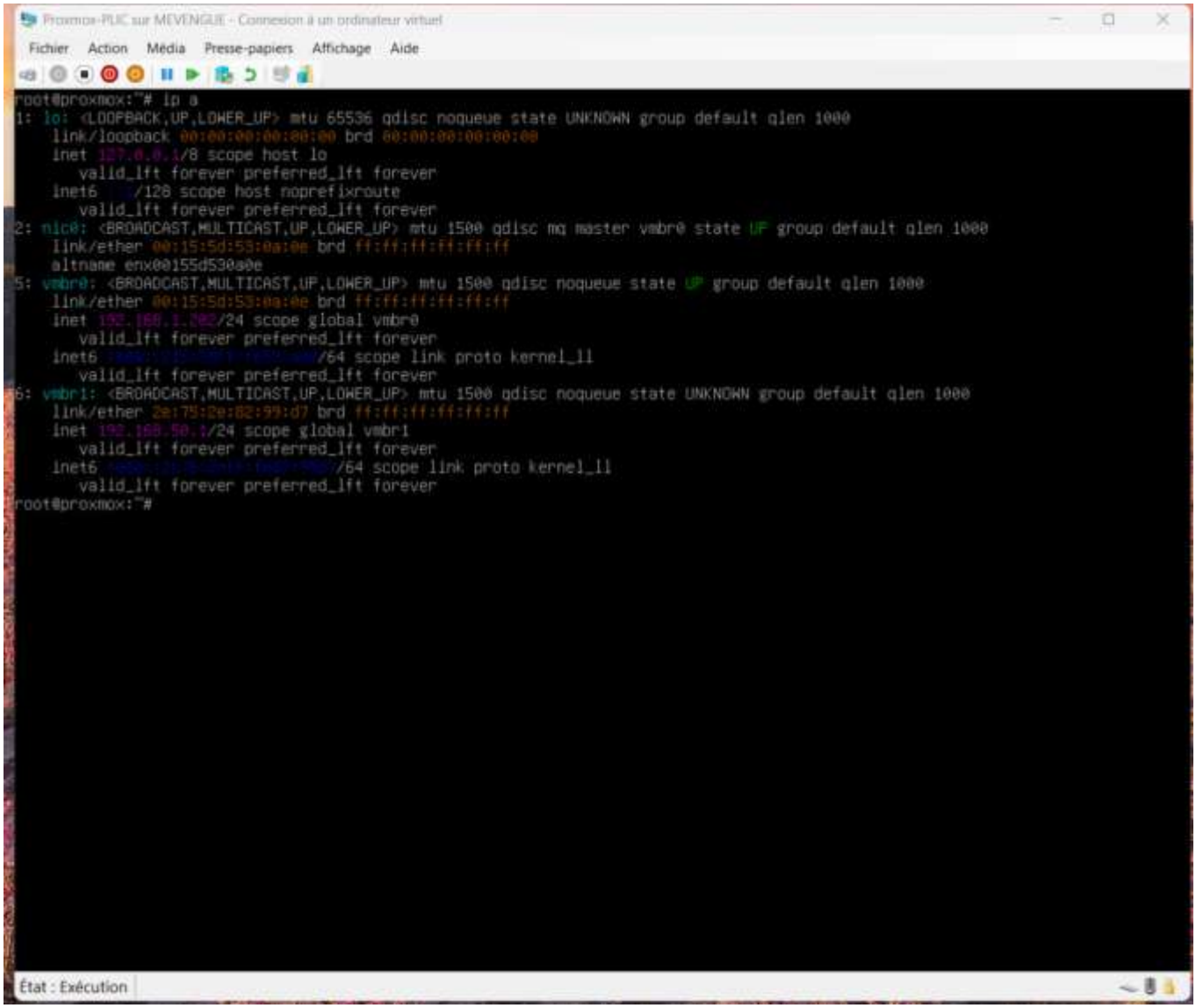
2. Hyper-V:

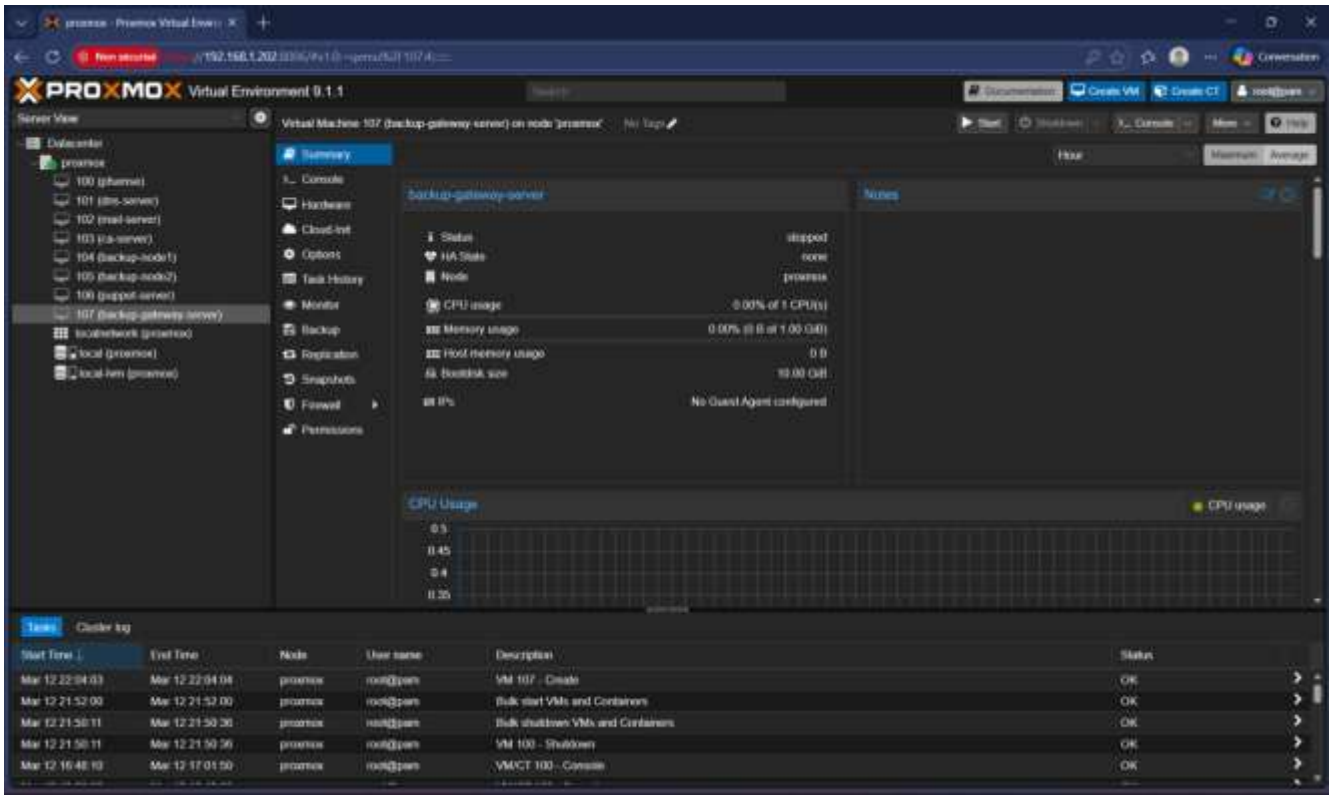



Paramètres pour Proxmox-PLIC sur MEVENGUE
Proxmox-PLIC
Matériel
Ajouter un matériel
BIOS
Démarrer à partir de CD
Sécurité
Lecteur de stockage de clé dés...
Mémoire
8192 Mo
Processeur
4 processeurs virtuels
Contrôleur IDE 0
Disque dur
Proxmox-PLIC_91702FCC-...
Contrôleur IDE 1
Lecteur de DVD
Aucun
Contrôleur SCSI
Carte réseau
Internet-Switch
COM 1
Aucun
COM 2
Aucun
Lecteur de disquettes
Aucun
Gestion
Nom
Proxmox-PLIC
Services d'Intégration
Quelques services offerts
Points de contrôle
Standard
Emplacement du fichier de pagi...
C:\ProgramData\Microsoft\Win...
Ajouter un matériel
Vous pouvez utiliser ce paramètre pour ajouter des périphériques à votre ordinateur virtuel.
Sélectionnez les périphériques à ajouter, puis cliquez sur le bouton Ajouter.
Contrôleur SCSI
Carte réseau
Carte réseau héritée
Adaptateur Fibre Channel
Ajouter
OK
Annuler
Appliquer



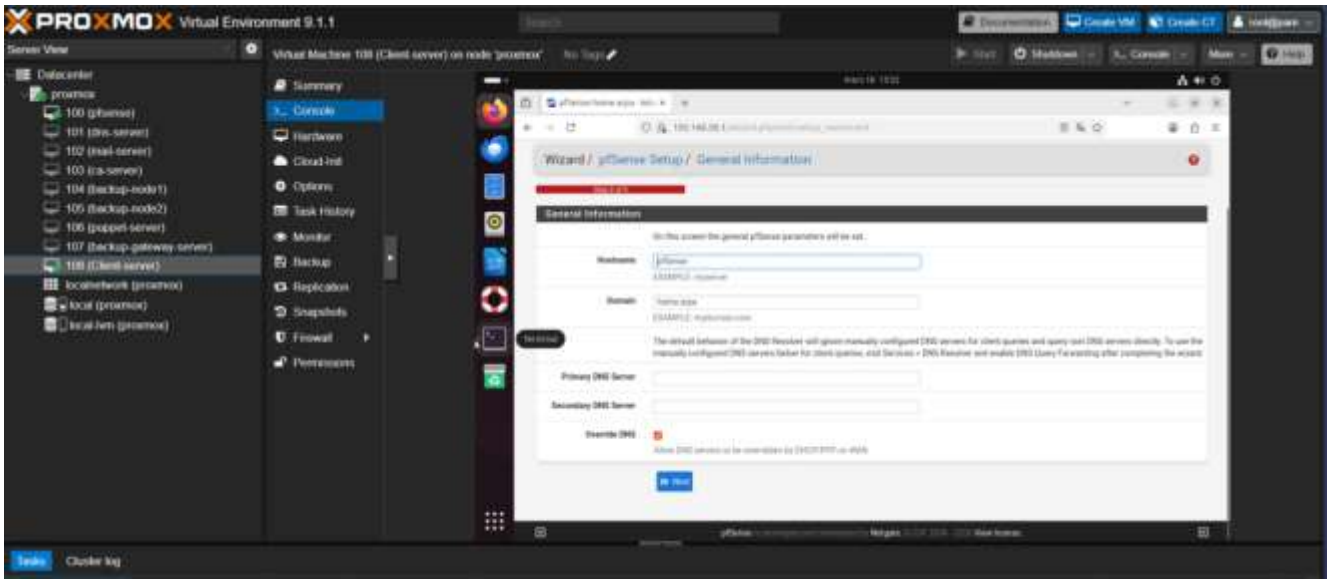
3. Proxmox :

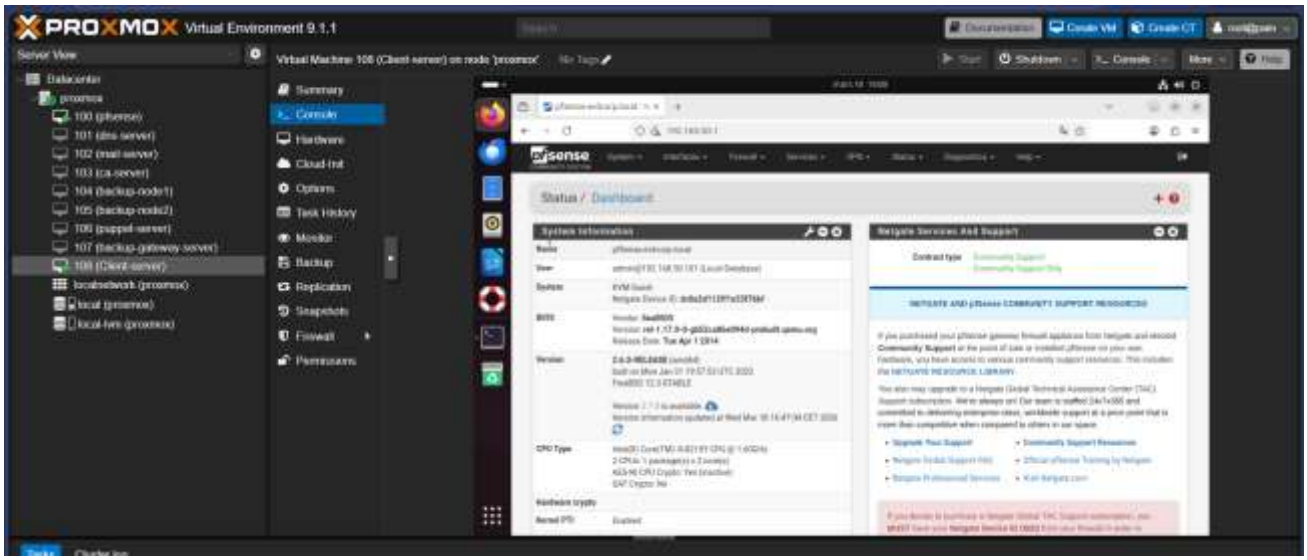

Proxmox-PLIC sur MEVENGUE - Connexion à un ordinateur virtuel
Fichier Action Média Presse-papiers Affichage Aide
root@proxmox:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
inet6 ::1/128 scope host noprefixroute
valid_lft forever preferred_lft forever
2: nic0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq master vmbr0 state UP group default qlen 1000
link/ether 00:15:5d:55:0a:0e brd ff:ff:ff:ff:ff:ff
altname enx00155d550a0e
5: vmbr0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default qlen 1000
link/ether 00:15:5d:53:0a:0e brd ff:ff:ff:ff:ff:ff
inet 192.168.1.202/24 scope global vmbr0
valid_lft forever preferred_lft forever
inet6 /64 scope link proto kernel_ll
valid_lft forever preferred_lft forever
6: vmbr1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group default qlen 1000
link/ether 2a:75:2e:82:99:d7 brd ff:ff:ff:ff:ff:ff
inet 192.168.50.1/24 scope global vmbr1
valid_lft forever preferred_lft forever
inet6 /64 scope link proto kernel_ll
valid_lft forever preferred_lft forever
root@proxmox:~#
État : Exécution

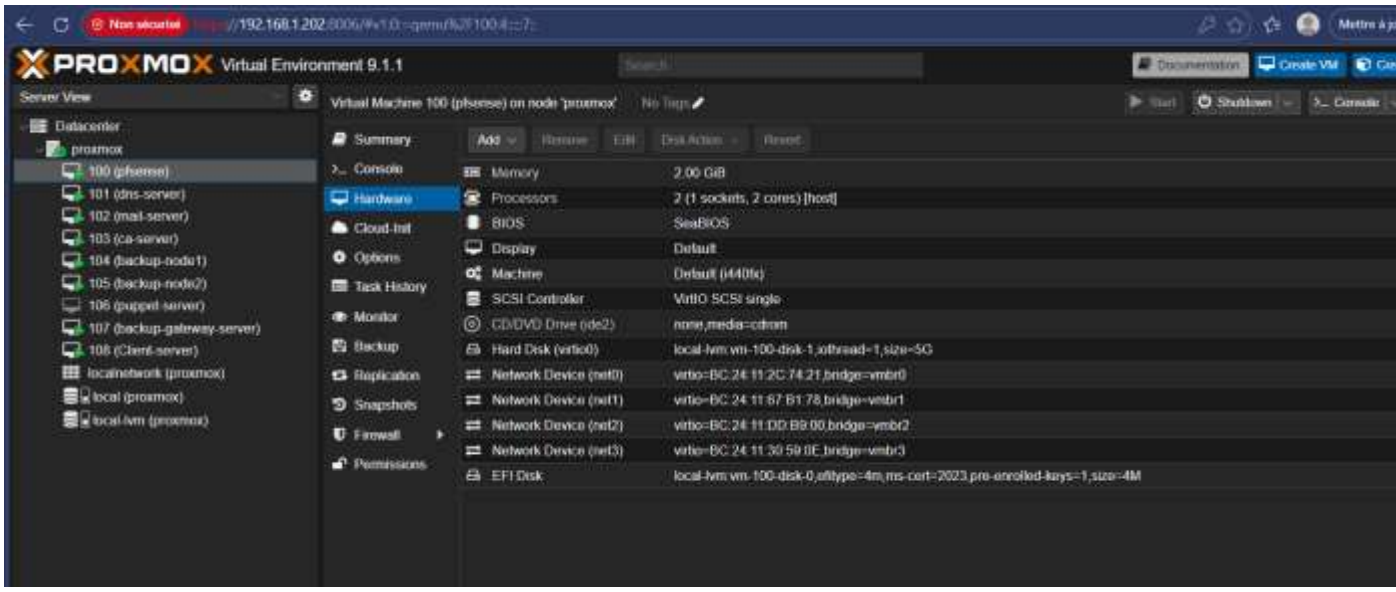


4. PFSENSE :

Pfsense fonctionnant sur la VM Client :


PROXMOX Virtual Environment 9.1.1
Server View
Datacenter
proxmox
100 (pfsense)
101 (dns-server)
102 (mail-server)
103 (ca-server)
104 (backup-node1)
105 (backup-node2)
106 (puppet-server)
107 (backup-gateway-server)
108 (Client-server)
localnetwork (proxmox)
local (proxmox)
local-lvm (proxmox)
Virtual Machine 108 (Client-server) on node 'proxmox'
Summary
Console
Hardware
Cloud-Init
Options
Task History
Monitor
Backup
Replication
Snapshots
Firewall
Permissions
Wizard / pfSense Setup / General Information
General Information
Tasks
Cluster log


PROXMOX Virtual Environment 9.1.1
Server View
Datacenter
proxmox
100 (pfsense)
101 (dns-server)
102 (mail-server)
103 (ca-server)
104 (backup-node1)
105 (backup-node2)
106 (puppet-server)
107 (backup-gateway-server)
108 (Client-server)
localnetwork (proxmox)
local (proxmox)
local-lvm (proxmox)
Virtual Machine 108 (Client-server) on node 'proxmox'
Summary
Console
Hardware
Cloud-Init
Options
Task History
Monitor
Backup
Replication
Snapshots
Firewall
Permissions
pfsense
Status / Dashboard
System Information
Netgate Services And Support



5. Architecture globale de l'infrastructure :

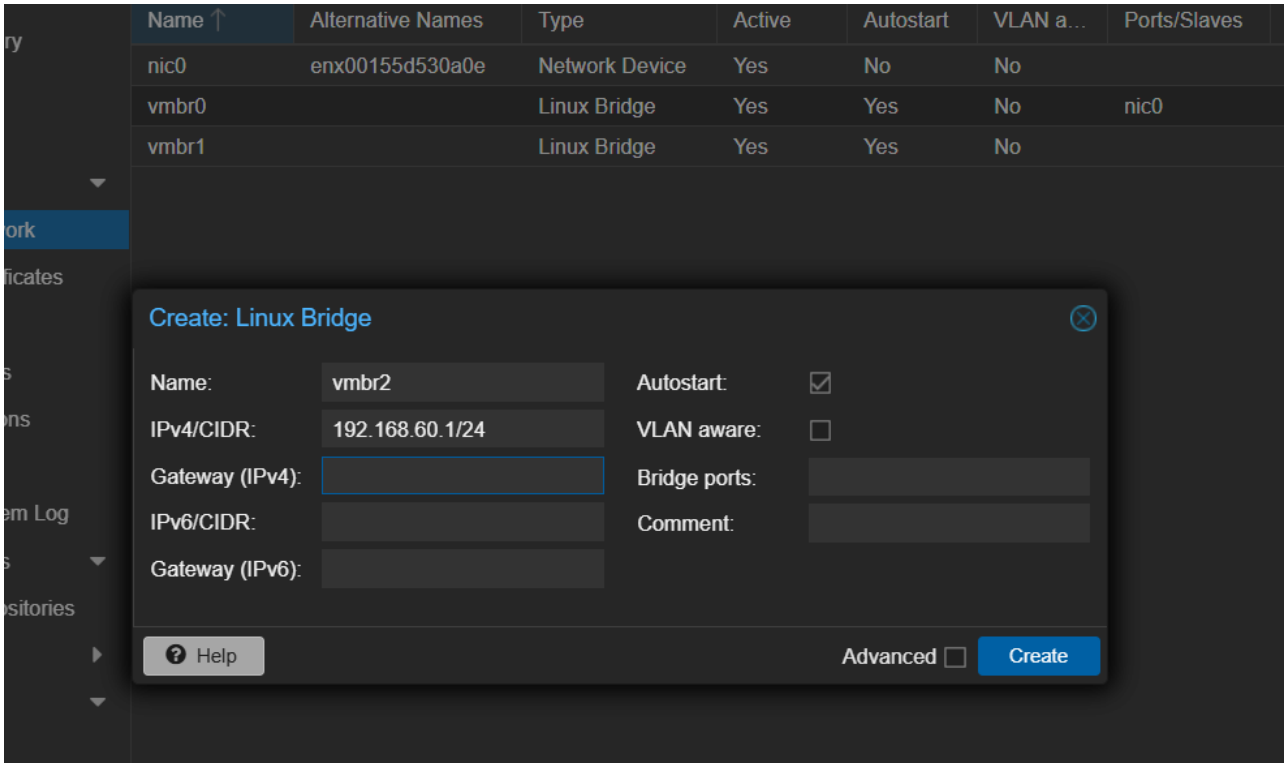
L'architecture repose sur une segmentation réseau claire afin d'assurer l'isolation des services et la sécurité de l'infrastructure.

Les différentes zones réseau sont :

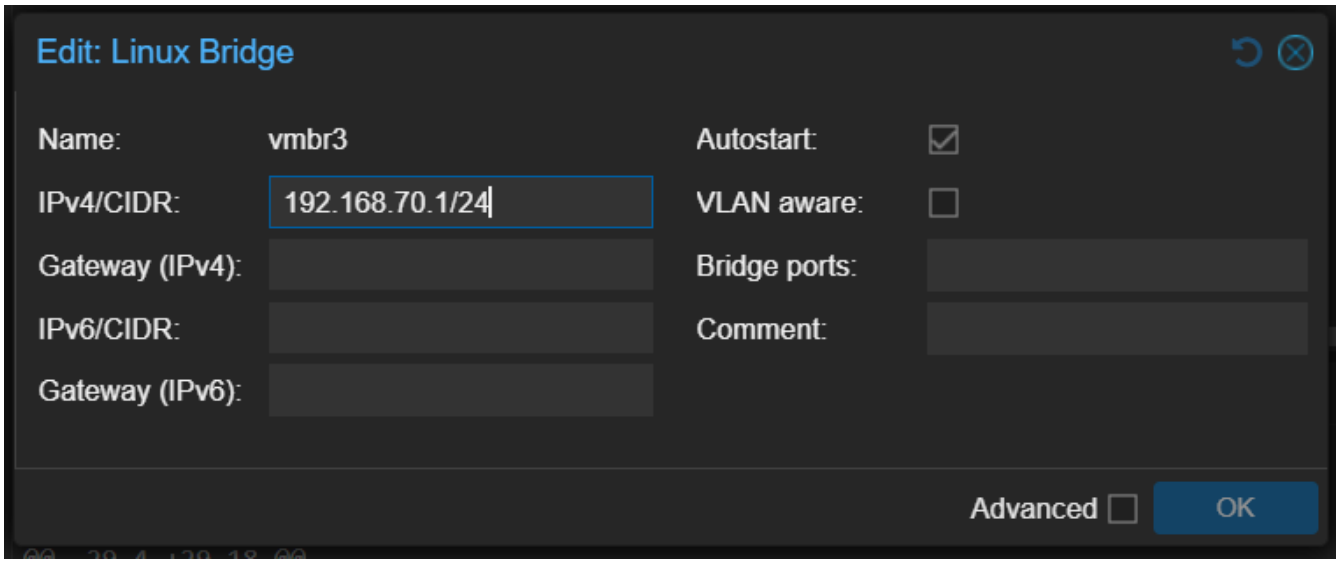
Réseau	Fonction
WAN	Connexion Internet simulée
LAN	Réseau interne de l'entreprise
Backup Gateway	Réseau dédié au tunnel IPSec
Backup Network	Réseau du site de sauvegarde

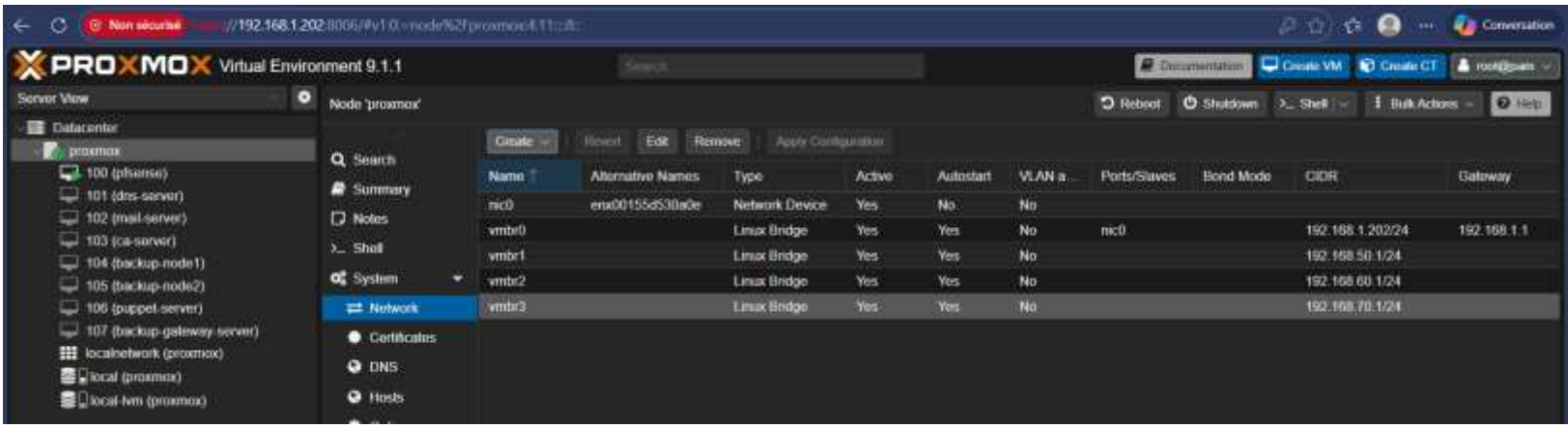
Cette segmentation permet de contrôler précisément les flux réseau et de garantir que seules les données de sauvegarde transitent dans le tunnel IPSec.

Vmbr2 - Réseau dédié au tunnel IPSec

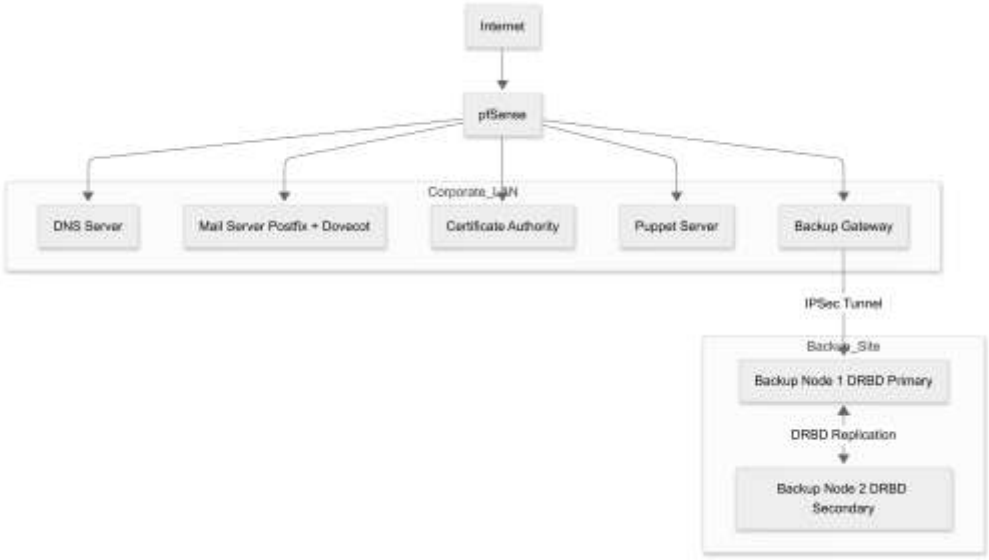


Vmbr3 - Réseau de stockage backup





6. Architecture réseau :



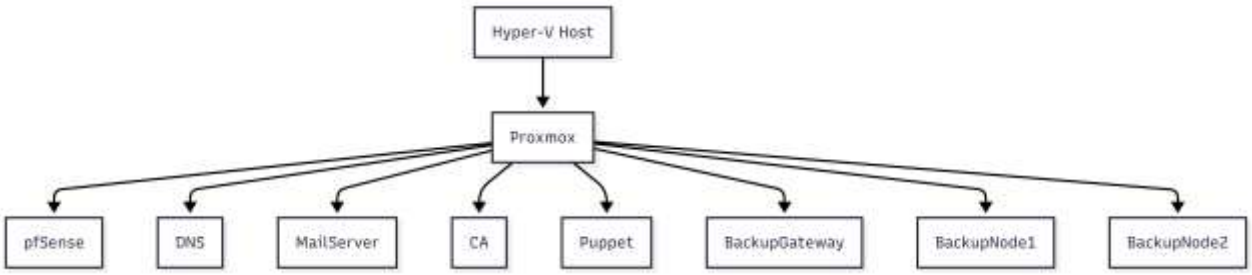
Ce diagramme représente la structure logique de l'infrastructure.

Les éléments principaux sont :

- Un firewall central (pfSense)
- Un réseau interne contenant les serveurs critiques
- Un tunnel IPSec sécurisé vers le site de sauvegarde
- Deux serveurs de stockage répliqués via DRBD

7. Architecture de virtualization :

L'infrastructure repose sur plusieurs niveaux de virtualisation. Cette architecture permet de simuler un environnement d'entreprise complet dans un laboratoire virtualisé.



8. Description des machines virtuelles :

L'infrastructure repose sur plusieurs machines virtuelles ayant chacune un rôle spécifique.

- **pfSense :**

Le firewall pfSense constitue la première ligne de défense du réseau. Il contrôle l'ensemble des flux entrants et sortants de l'infrastructure.

Fonctions :

- Routage
- Filtrage du trafic
- Gestion des VLAN
- Tunnel IPSec
- NAT

Configuration :

CPU : 2 vCPU
RAM : 2 GB
Disque : 20 GB

Interfaces réseau :

WAN
LAN
Backup Network

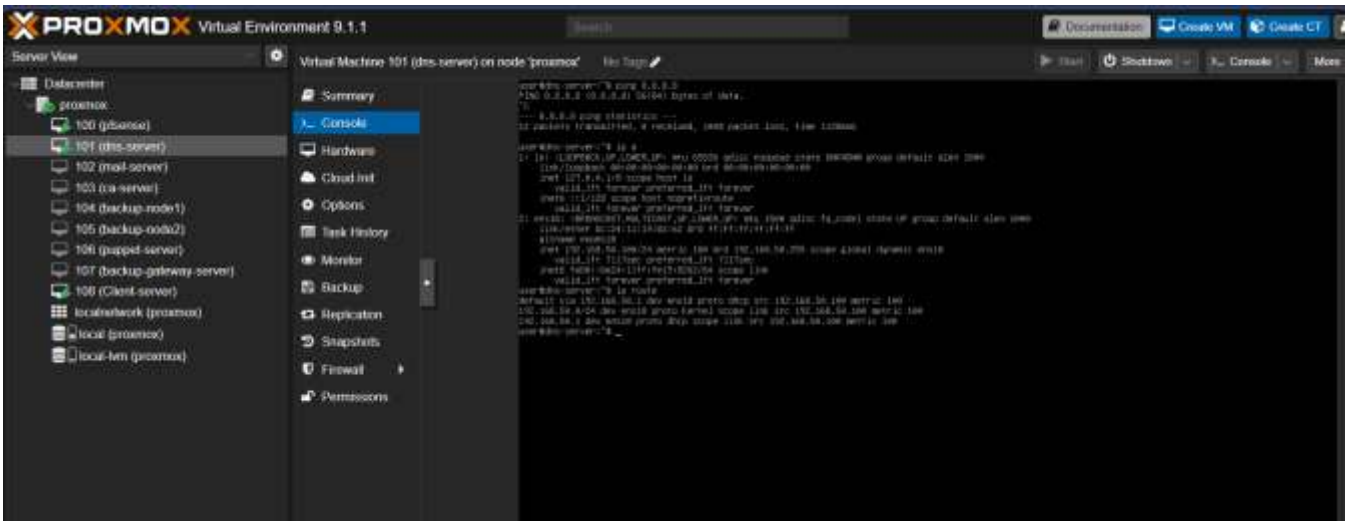
- **Serveur DNS :**

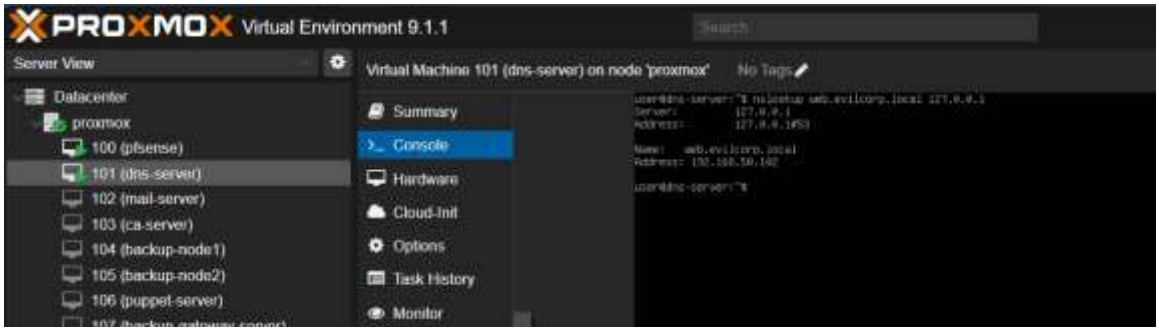
Le serveur DNS permet la résolution des noms internes de l'entreprise. Il est indispensable pour permettre aux différents services de communiquer via des noms de domaine plutôt que des adresses IP.

Logiciel utilisé : Bind9

Configuration :

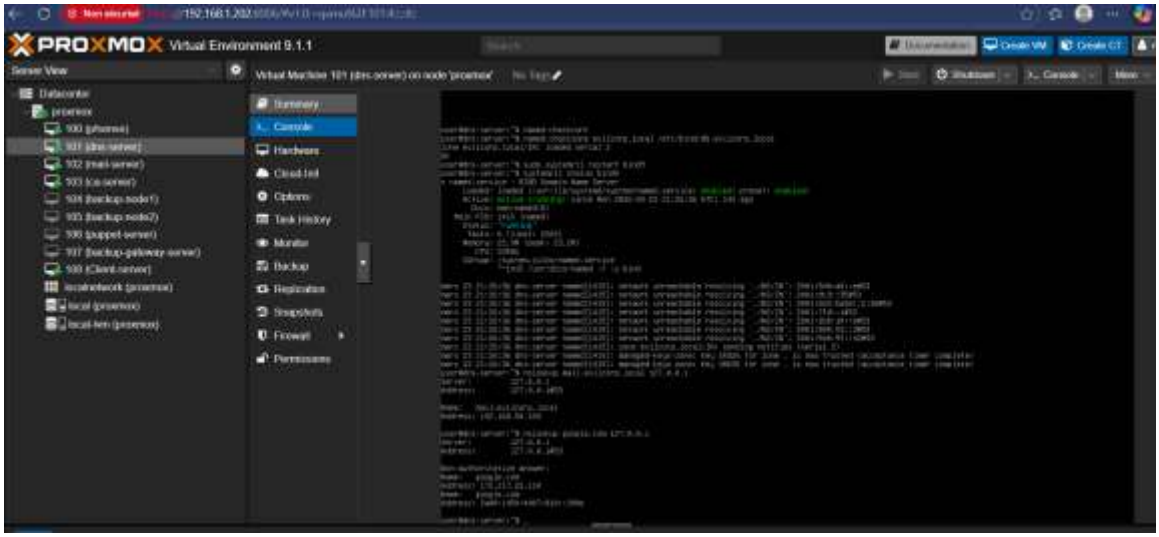
CPU : 1 vCPU
RAM : 1 GB
Disque : 10 GB





Rappel :

```
ca-server 192.168.50.102/24
dns-server 192.168.50.100/24
mail-server 192.168.50.103/24
client-server 192.168.50.107/24
```

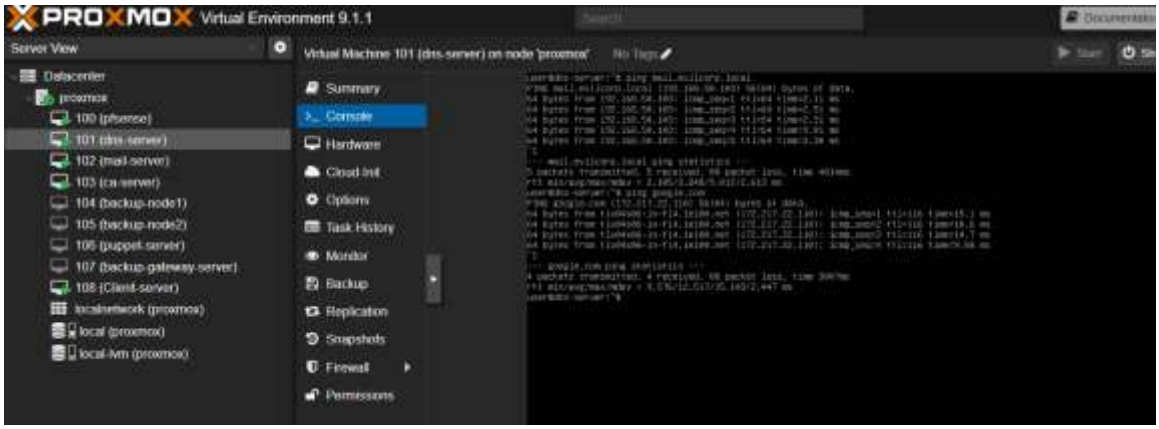


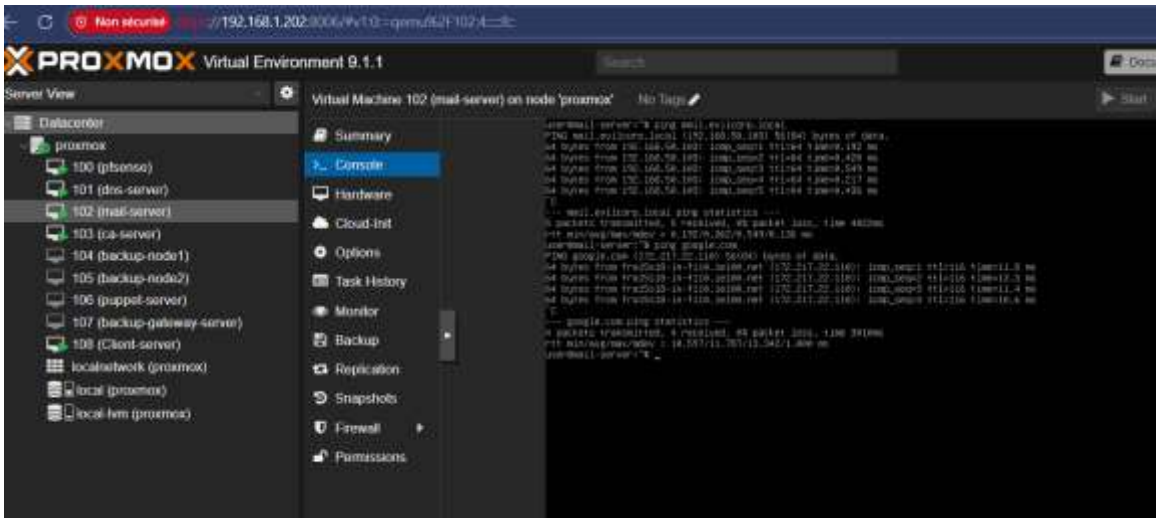
- ✓ bind9 active (running)
- ✓ zone evilcorp.local chargée

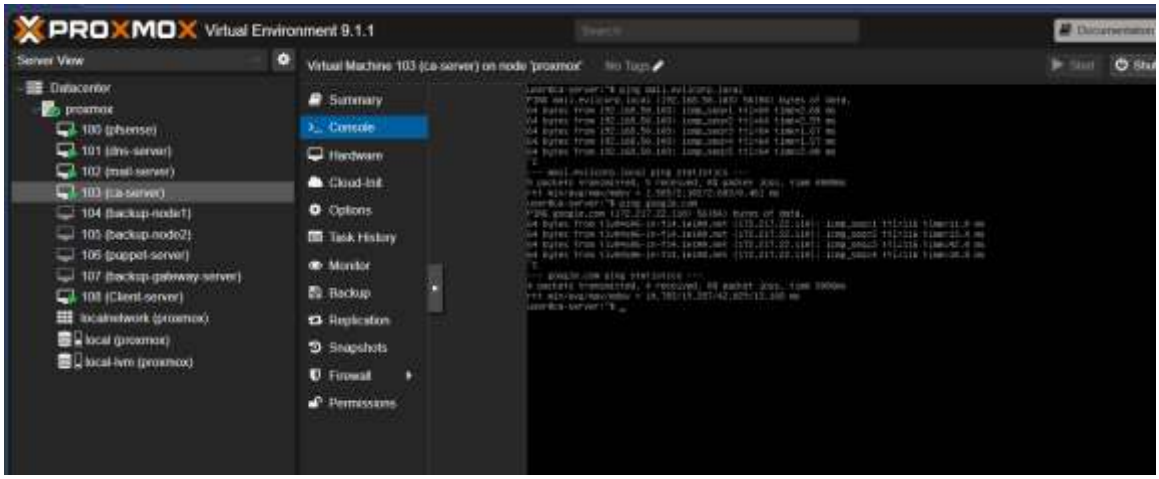
- ✓ nslookup mail.evilcorp.local → OK
- ✓ nslookup google.com → OK

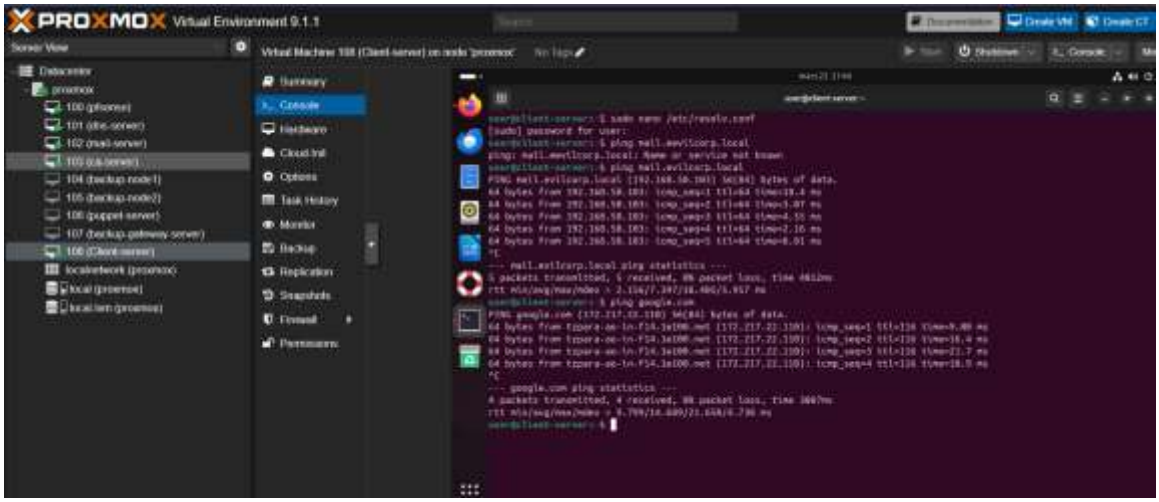
Le DNS marche LOCAL, mais les autres VMs ne l'utilisent PAS encore, toutes les VMs doivent utiliser : DNS = 192.168.50.100.

Config sur toutes les VMs : dns-server, mail-server, ca-server, client : sudo nano /etc/resolv.conf, mets : nameserver 192.168.50.100 / search evilcorp.local :









Résultat Final :

- ✓ résolution interne OK
- ✓ résolution internet OK
- ✓ infra cohérente

- **Serveur Mail**

Le serveur de messagerie constitue le cœur de l'infrastructure. Il gère l'envoi, la réception et le stockage des courriels des utilisateurs.

Logiciels utilisés : Postfix, Dovecot

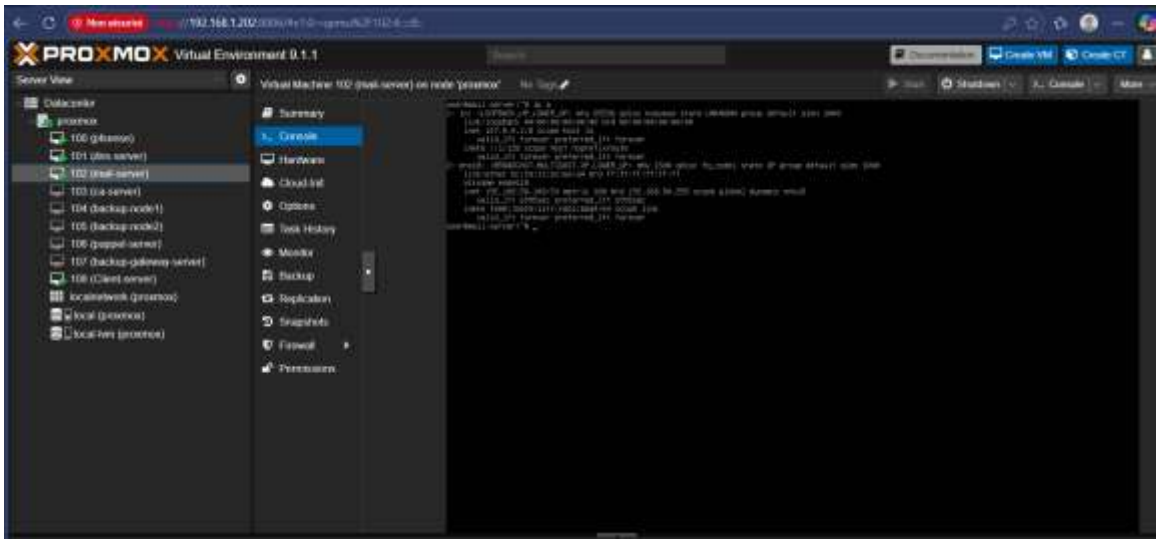
Protocoles supportés :

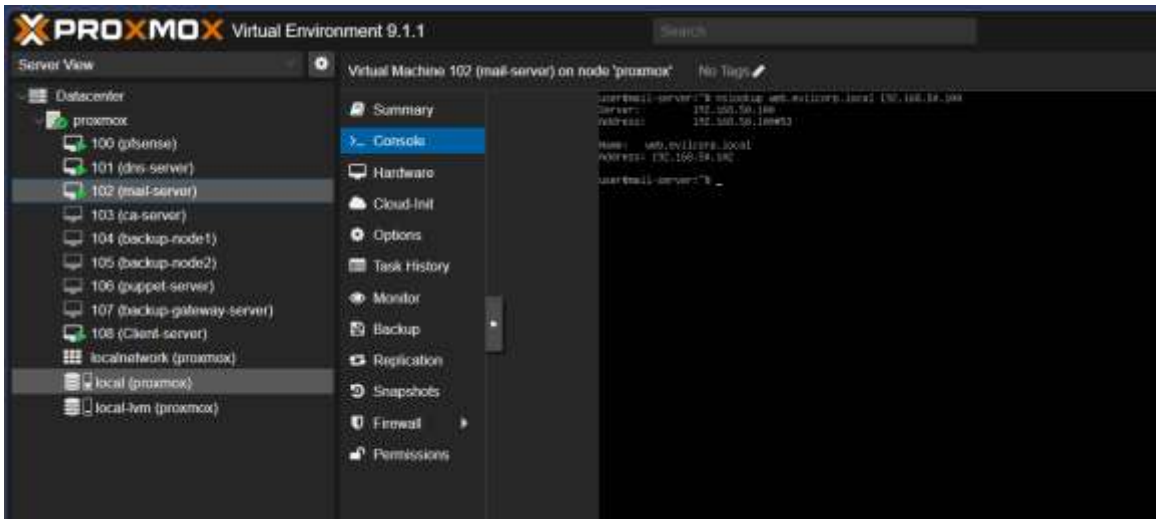
SMTP (envoi de mail)
IMAP (lecture de mail)

Tous les protocoles sont sécurisés via TLS.

Configuration :

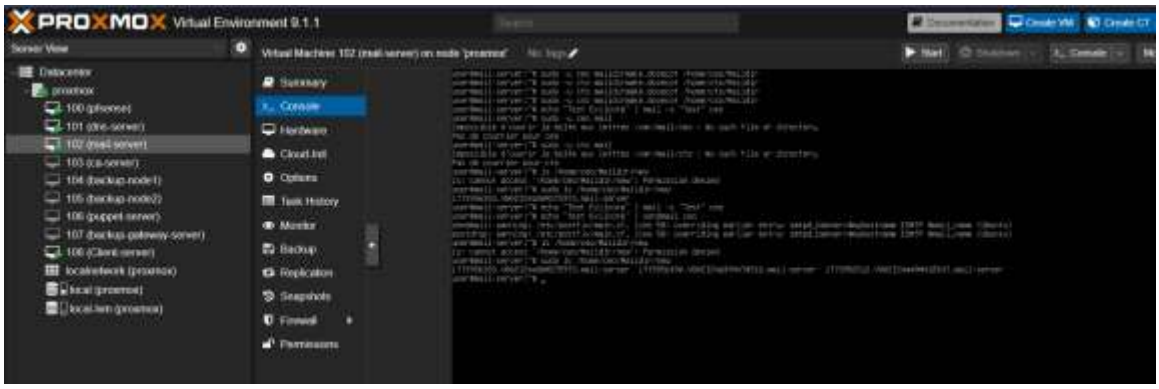
CPU : 2 vCPU
RAM : 2 GB
Disque : 40 GB

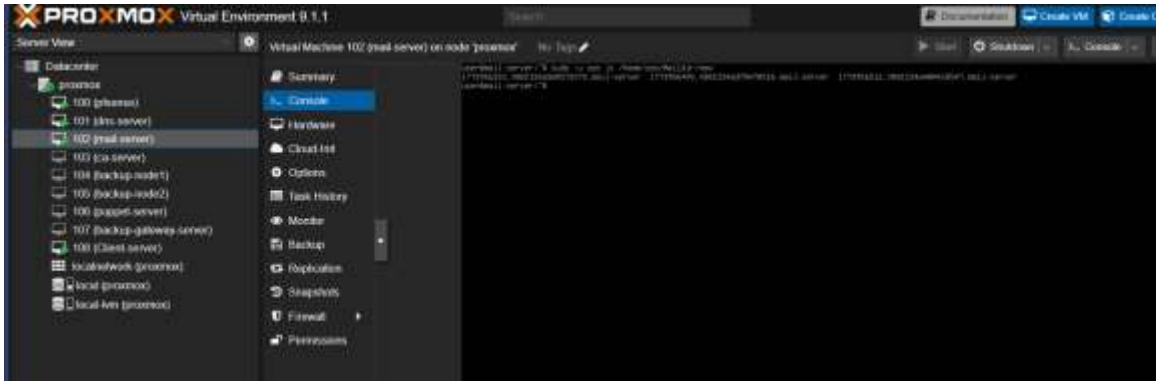




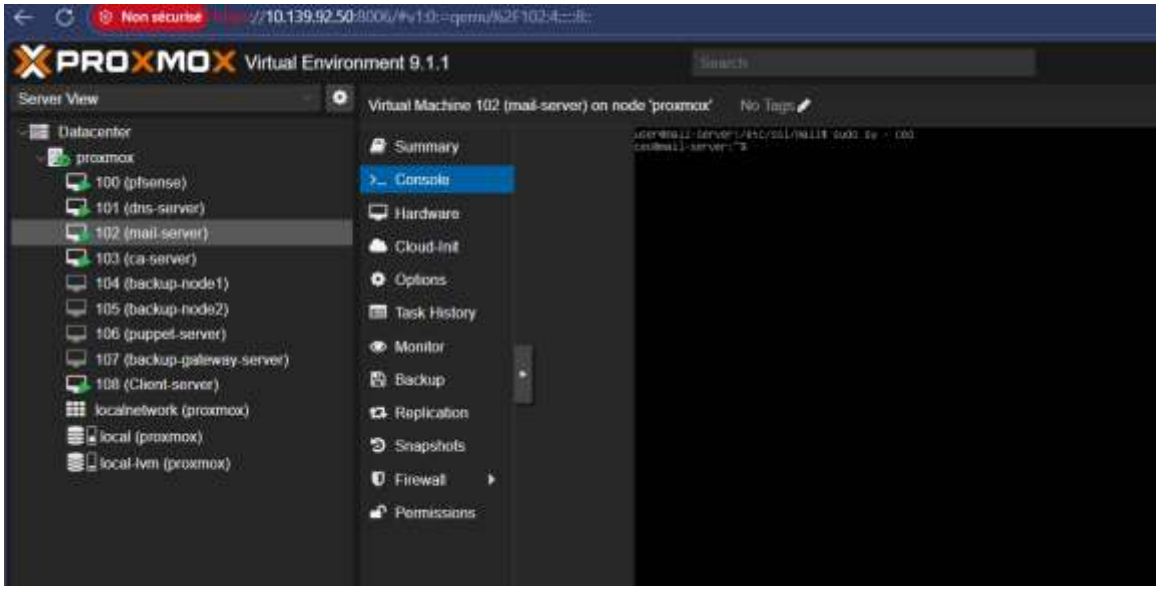
Donc :

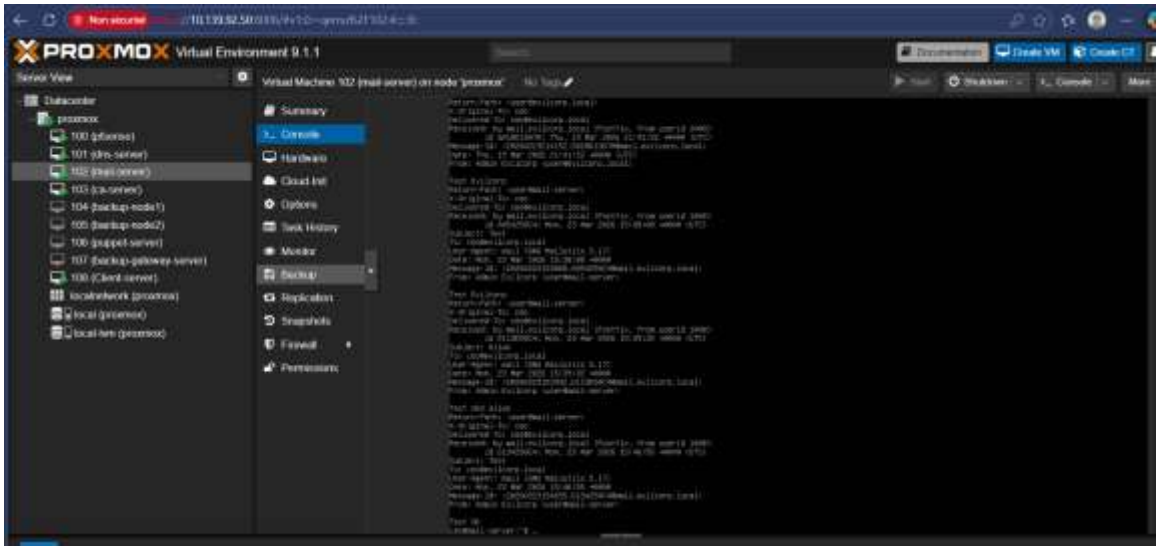
- ✔ DNS accessible depuis le réseau
- ✔ Bind9 correctement configuré
- ✔ Communication inter-VM OK
- ✔ Infrastructure réseau stable





- ✔ mails envoyés
- ✔ mails stockés
- ✔ Maildir créé
- ✔ postfix OK



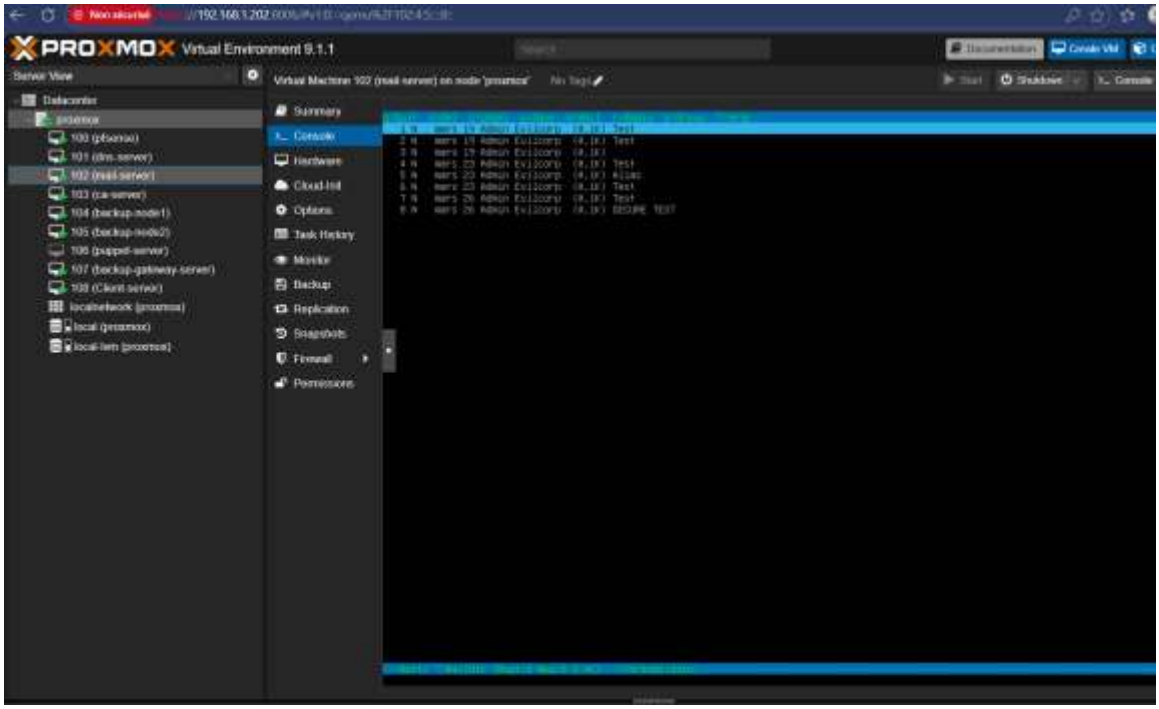


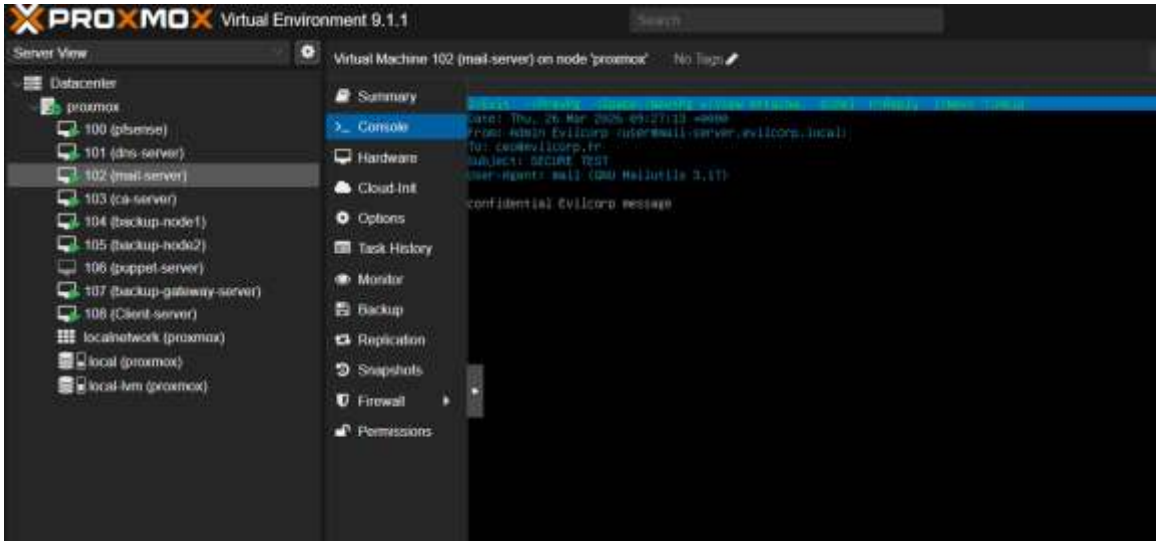
- ✔ Envoi OK
- ✔ Réception OK
- ✔ Alias OK
- ✔ Domaine evilcorp.local OK
- ✔ Postfix fonctionne
- ✔ Maildir fonctionne
- ✔ Certificat utilisé

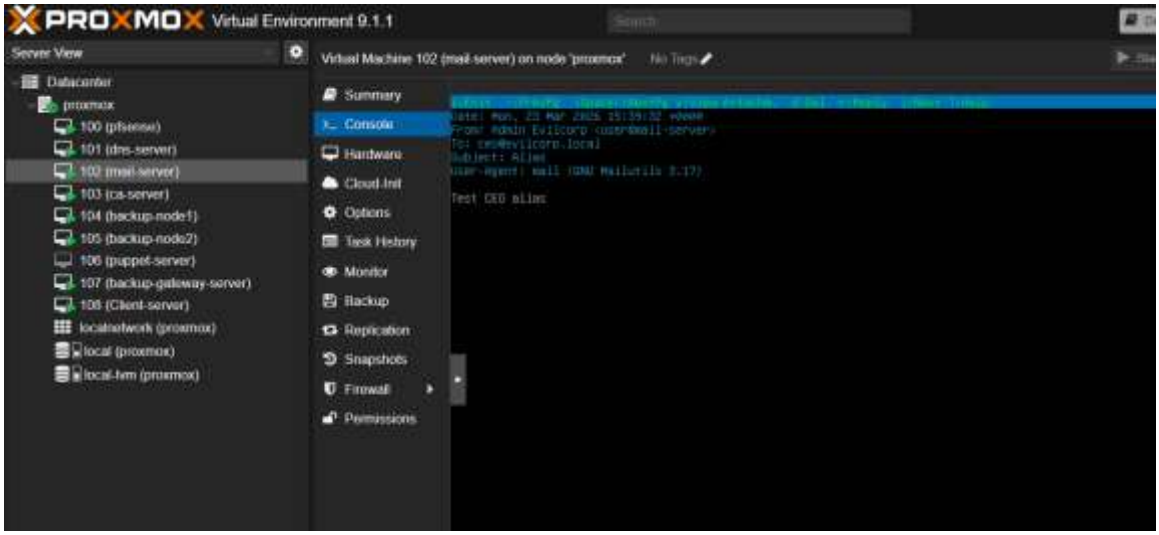
Infrastructure mail complète :

- ✔ MTA → Postfix
- ✔ Stockage → Maildir
- ✔ Utilisateurs locaux
- ✔ Alias (/etc/aliases)
- ✔ TLS (certificat CA)

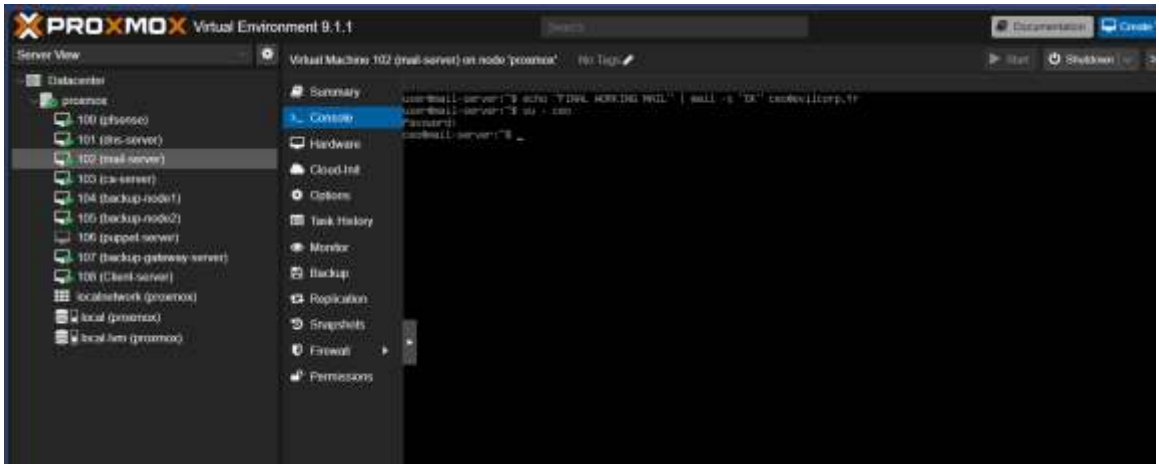
Ouverture des mails de l'utilisateur ceo avec mutt : mutt -f ~/Maildir

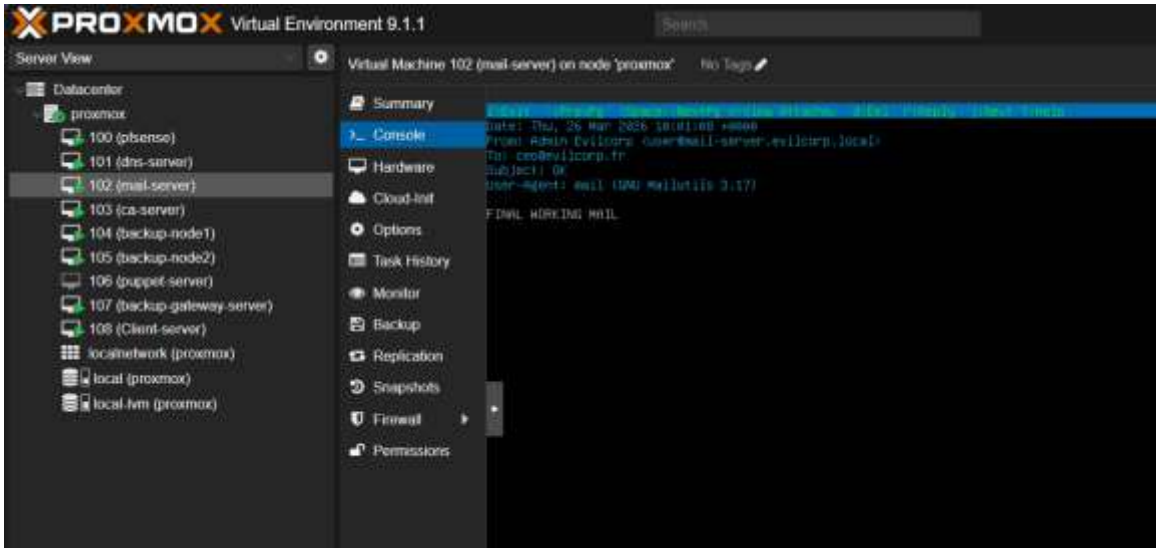






Test Final : envoie de mail vers ceo@evilcorp





- ✅ SMTP
- ✅ réception
- ✅ Maildir
- ✅ client mutt

- **Serveur Autorité de Certification (CA)**

Une autorité de certification interne est déployée afin de gérer les certificats de l'infrastructure. Cette machine est isolée car elle représente un élément critique de sécurité.

Rôle :

- Génération du certificat racine
- Signature des certificats des services
- Gestion du cycle de vie des certificats

Certificats générés :

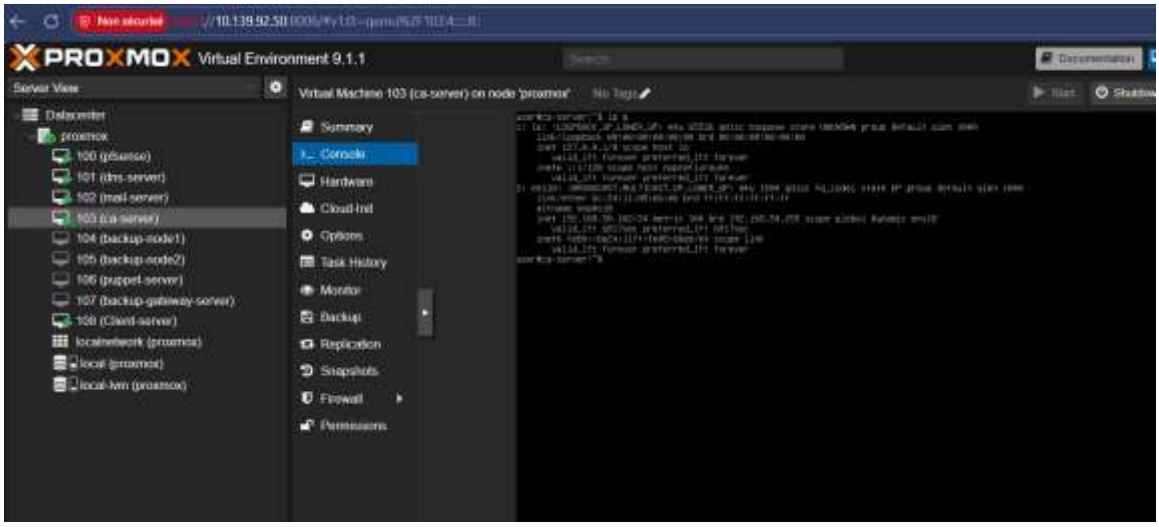
- Certificat serveur mail
- Certificat IPSec
- Certificat Puppet
- Certificat DNS si nécessaire

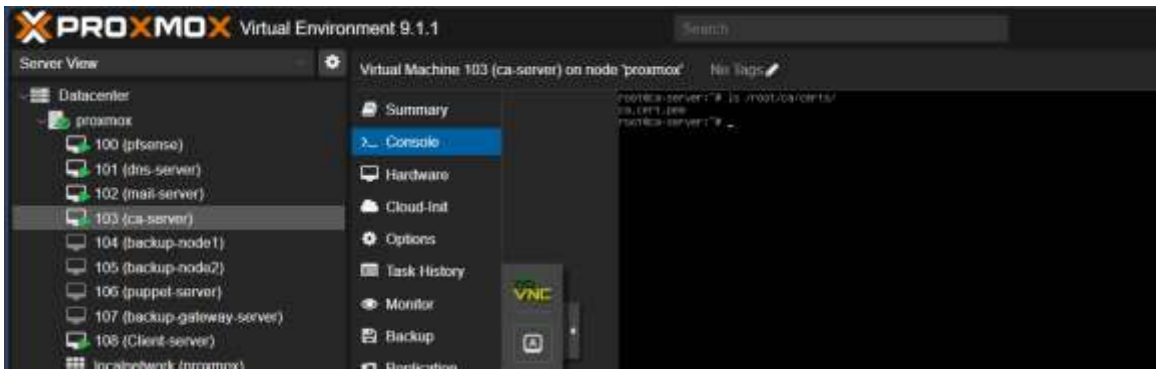
Logiciels utilisés :

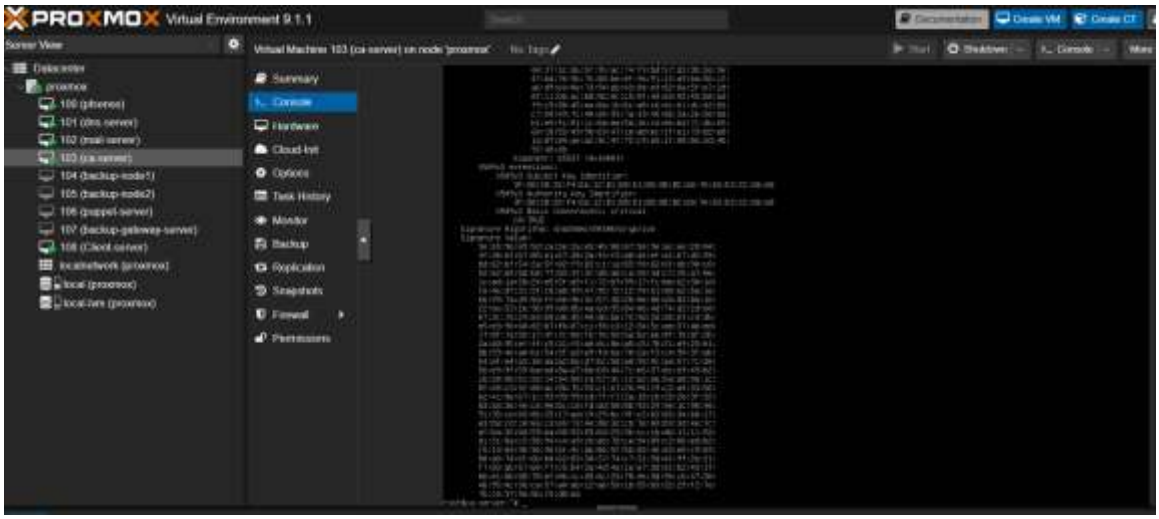
OpenSSL
EasyRSA

Configuration :

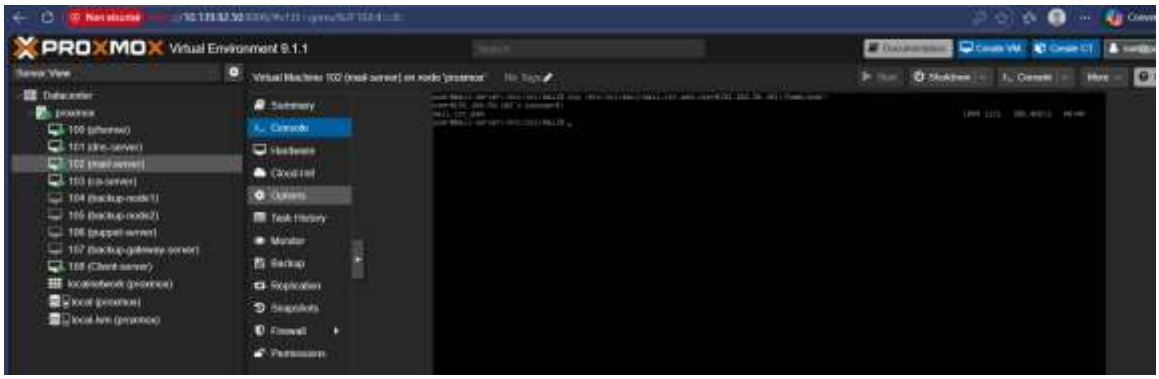
CPU : 1 vCPU
RAM : 1 GB
Disque : 10 GB

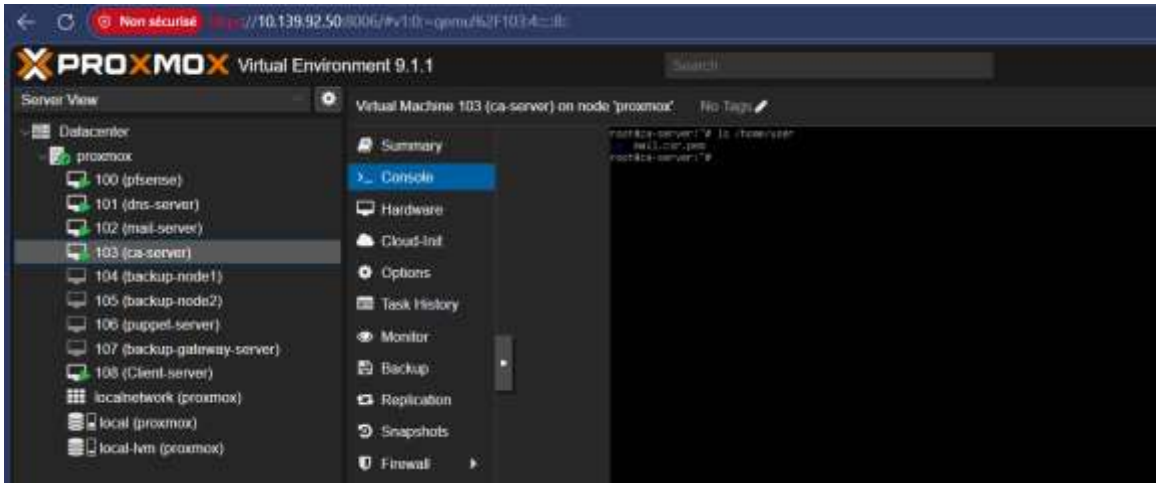

PROXMOX Virtual Environment 9.1.1
Server View
Datacenter
proxmox
100 (pfsense)
101 (dns-server)
102 (mail-server)
103 (ca-server)
104 (backup-node1)
105 (backup-node2)
106 (puppet-server)
107 (backup-gateway-server)
108 (Client-server)
localnetwork (proxmox)
local (proxmox)
local-lvm (proxmox)
Virtual Machine 103 (ca-server) on node 'proxmox'
No Tags
Summary
Console
Hardware
Cloud-Init
Options
Task History
Monitor
Backup
Replication
Snapshots
Firewall
Permissions


PROXMOX Virtual Environment 9.1.1
Search
Server View
Datacenter
proxmox
100 (pfsense)
101 (dns-server)
102 (mail-server)
103 (ca-server)
104 (backup-node1)
105 (backup-node2)
106 (puppet-server)
107 (backup-gateway-server)
108 (Client-server)
Virtual Machine 103 (ca-server) on node 'proxmox'
No Tags
Summary
Console
Hardware
Cloud-Init
Options
Task History
Monitor
Backup
VNC

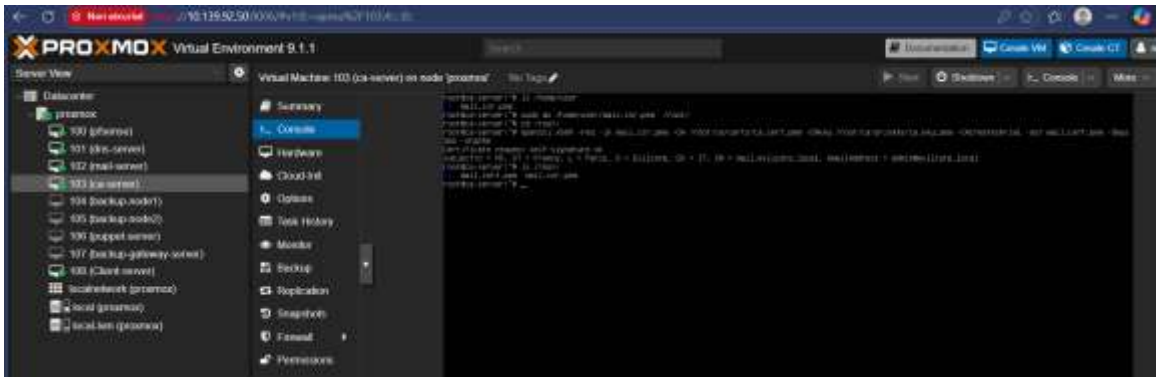


- ✔ ca.cert.pem présent
- ✔ affichage openssl x509 OK
- ✔ clé RSA générée
- ✔ certificat valide



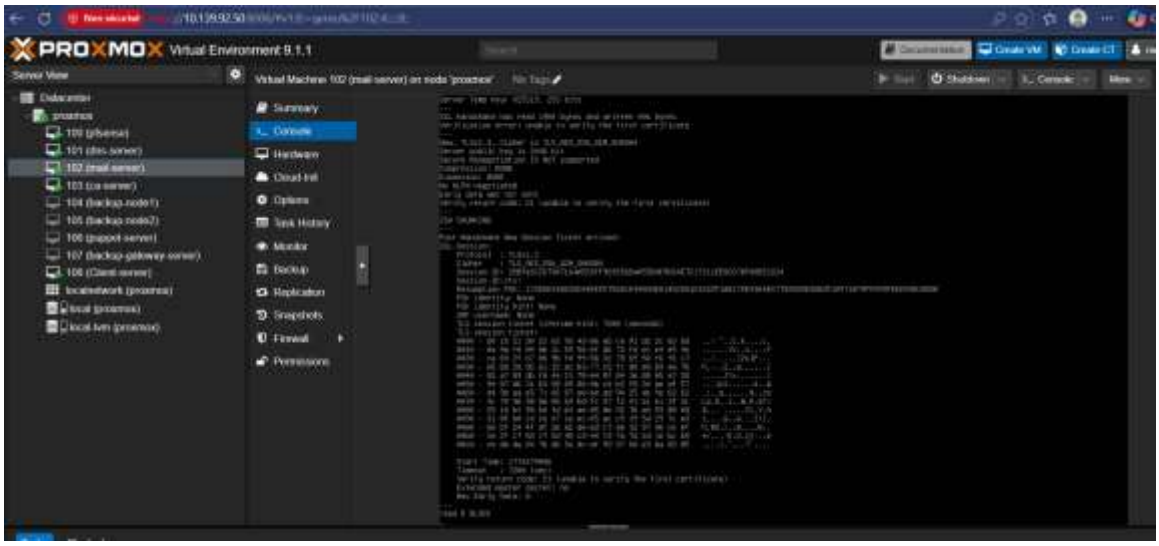


- Signature du Certificat :



Certificat signé avec succès ✅.

- Renvoi du Certificat vers mail-server (102) :



Le TLS est opérationnel.

Le screen montre :

- TLSv1.3 ✅
- Cipher OK ✅
- Handshake OK ✅

Le message : Verify return code: 21 (unable to verify the first certificate)

INTERPRÉTATION : Ce message = 100% NORMAL

Pourquoi ?

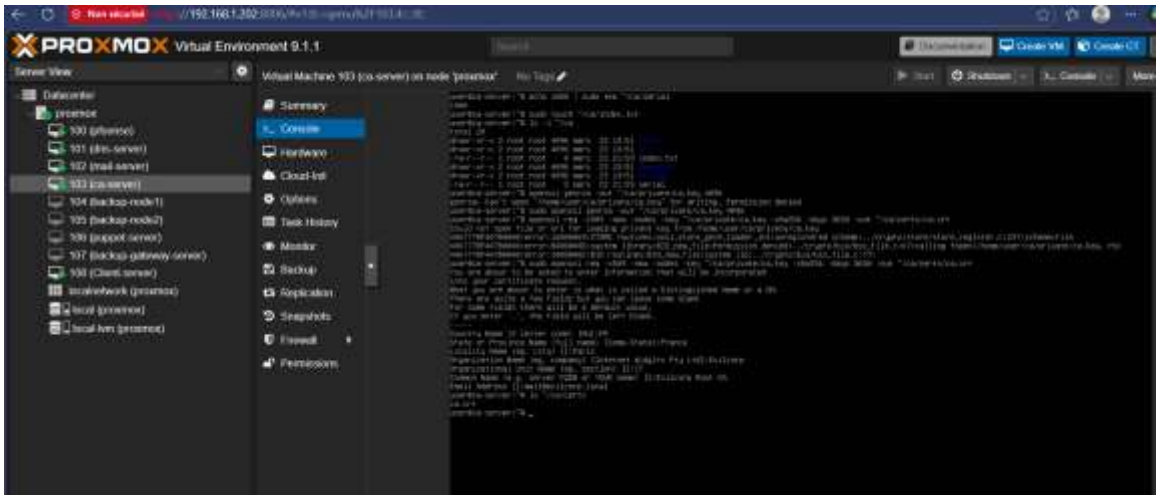
✔ on utilise une CA interne

✔ le système ne la connaît pas

❌ donc il ne peut pas vérifier

Résumé :

- ✔ SMTP over TLS
- ✔ chiffrement
- ✔ certificat signé
- ✔ communication sécurisée



- ✓ ca.crt créé
- ✓ structure CA complète
- ✓ CN = EvilCorp Root CA

- **Serveur Puppet**

Le serveur Puppet permet d'automatiser la configuration et l'administration des machines de l'infrastructure. L'utilisation de Puppet permet de centraliser la gestion de la sécurité et d'éviter les erreurs de configuration manuelle.

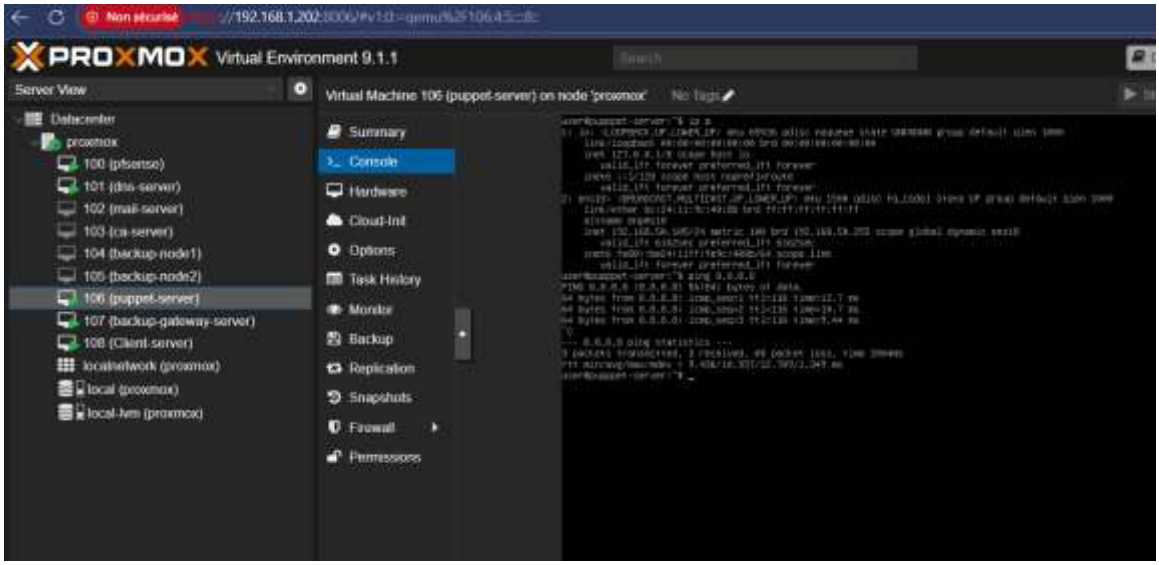
Fonctions principales :

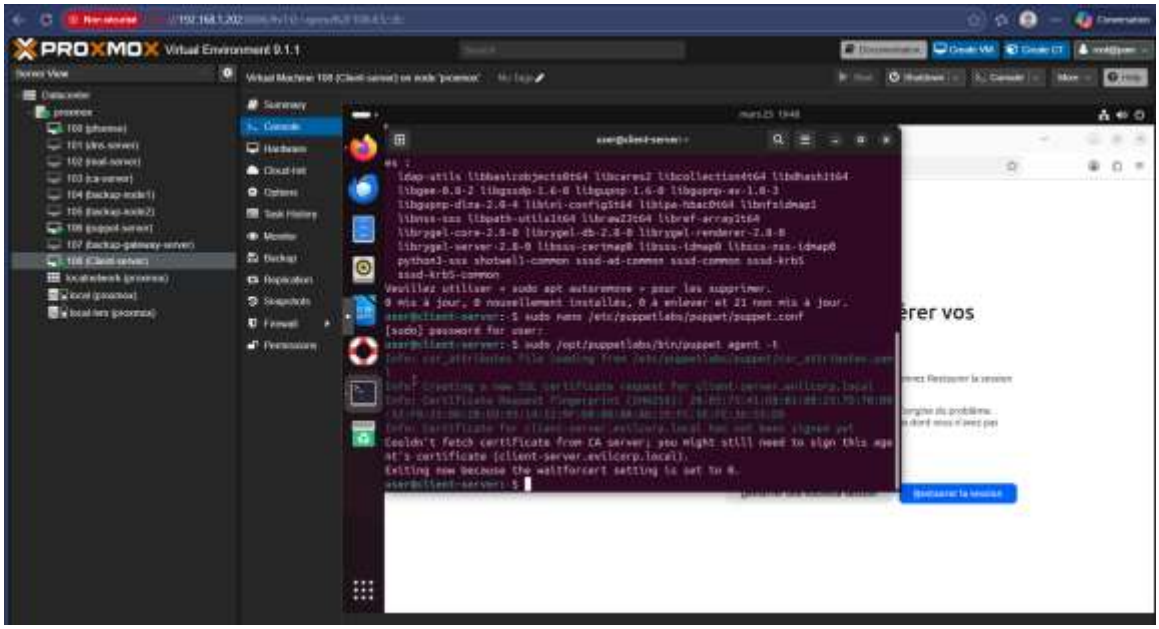
- Déploiement automatique du certificat racine
- Gestion des mises à jour de sécurité
- Gestion de la configuration des machines
- Gestion centralisée des clients

Puppet est particulièrement utile pour garantir que tous les systèmes de l'infrastructure possèdent le certificat racine de l'autorité de certification.

Configuration :

CPU : 2 vCPU
RAM : 2 GB
Disque : 20 GB





CE QUE MONTRE LE SCREEN :

Message clé :

Certificate for client-server.evilcorp.local has not been signed yet

Traduction :

➞ le client a bien contacté Puppet Server

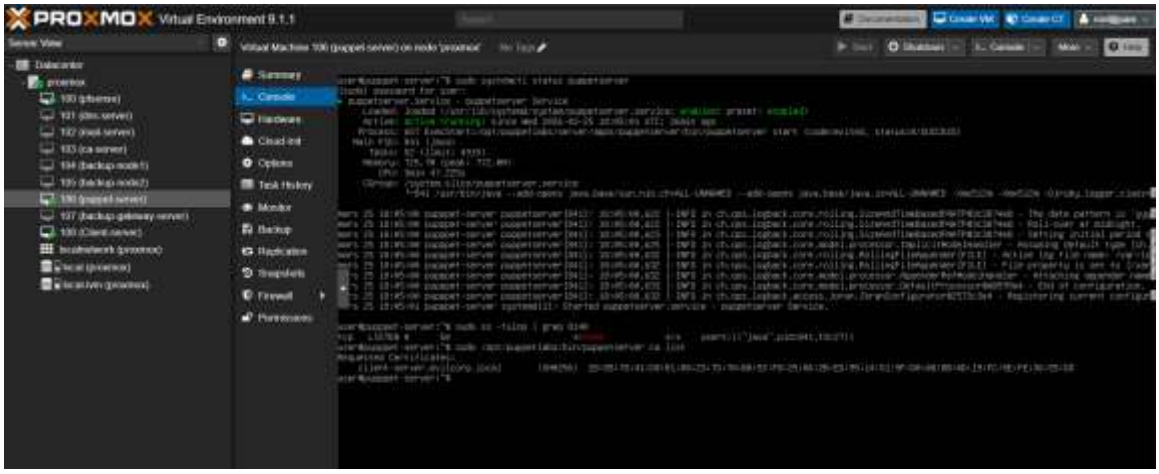
➞ le certificat a été créé

➞ MAIS il n'est pas encore validé

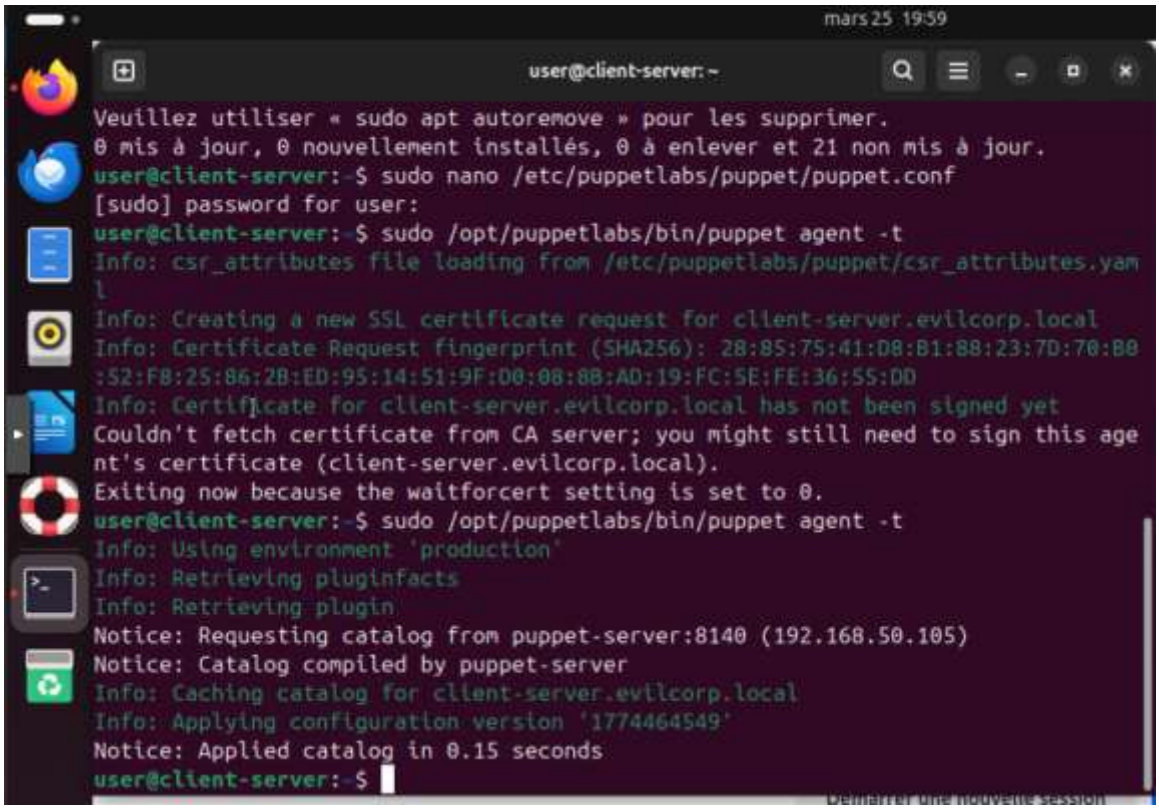
Donc tout fonctionne

Il manque juste la signature côté serveur

Etape suivante, voir les certificats en attente sur la Vm Puppet-server : sudo /opt/puppetlabs/bin/puppetserver ca list



Certificat signé :



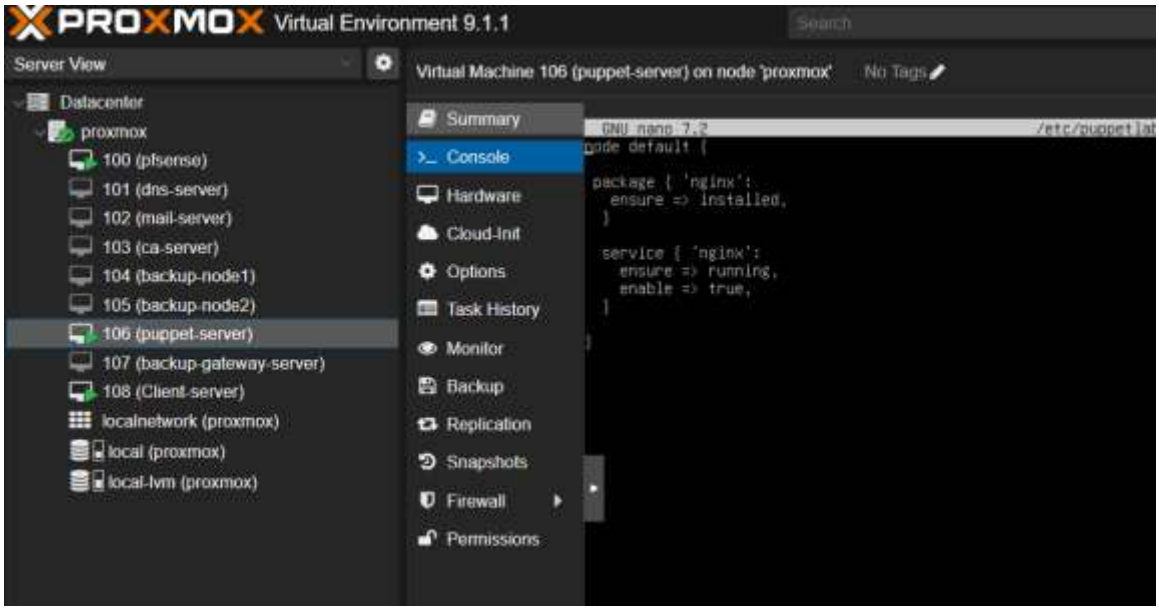
Le Puppet fonctionne.

Test Automatisation :

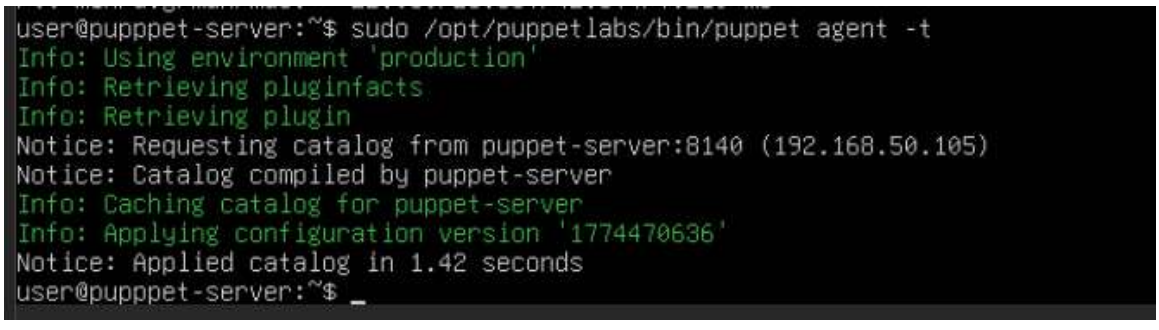
Automatisation Ok

Déploiement automatique d'un service sur client-server à partir de puppet-server :

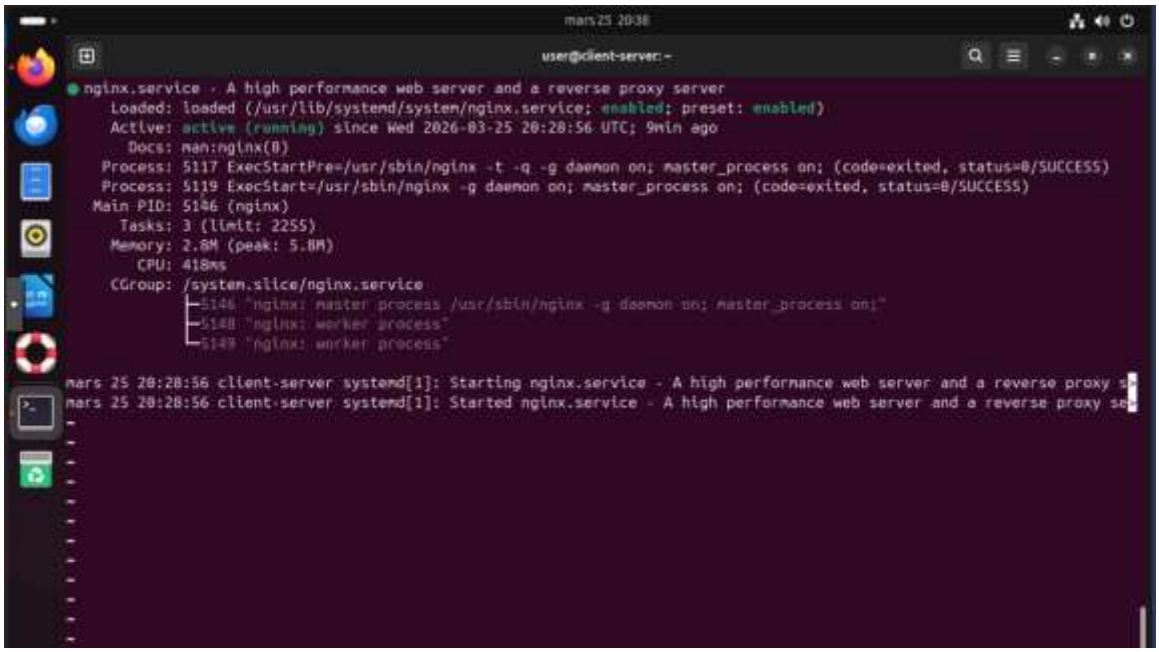
Configurer le manifest : sudo nano /etc/puppetlabs/code/environments/production/manifests/site.pp

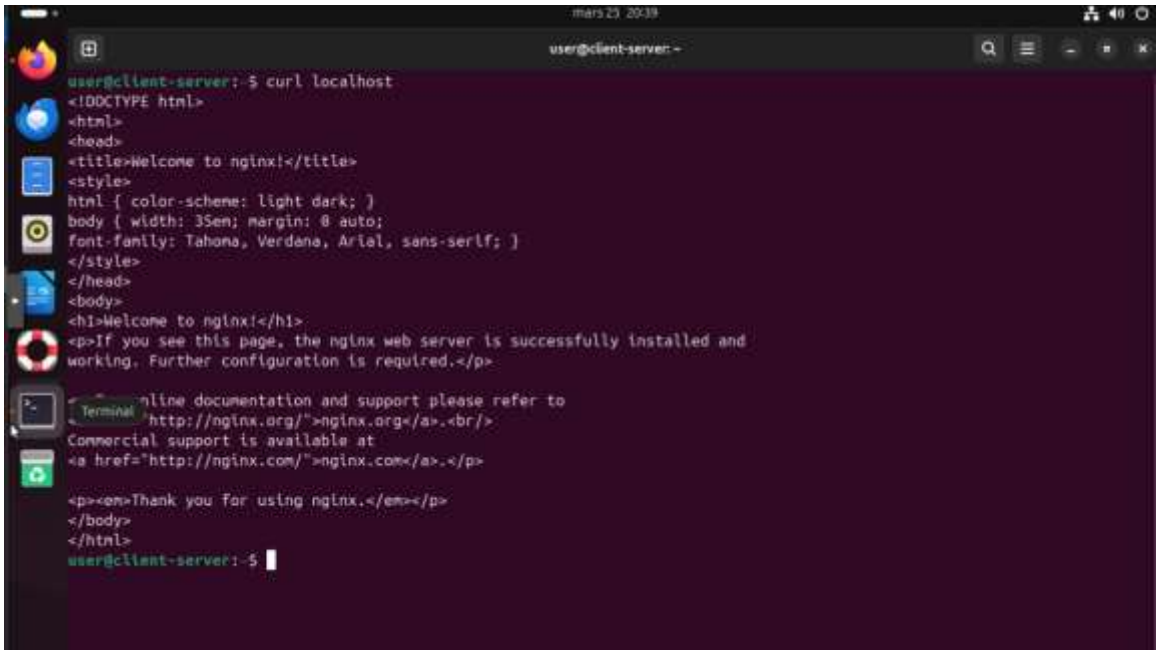


Puppet installe + démarre nginx.



Vérifier service dans Client-server :





Tout fonctionne parfaitement.

Etape suivante :

OBJECTIF : puppet-server gère :

- Mail Server → postfix + dovecot
- DNS Server → bind9
- Clients → nginx / config

= automatisation complète du projet

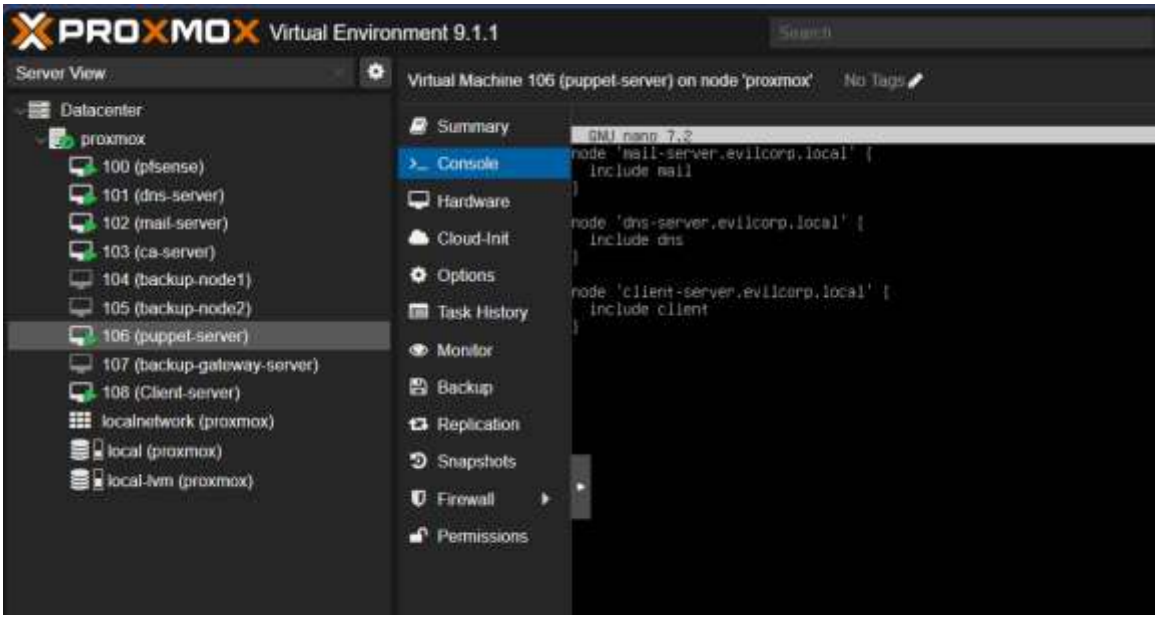
BUT : Transformer le Puppet en :

- Pilotage par rôle (mail / dns / client)
- Déploiement automatique par machine

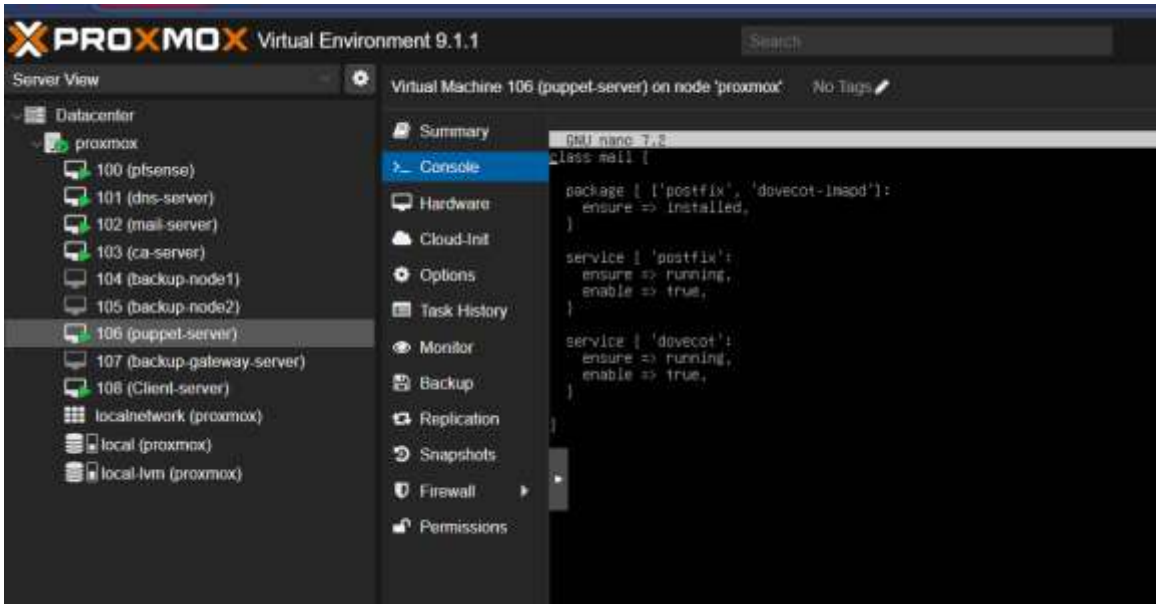
1- Créer les fichiers :

- sudo nano site.pp
- sudo nano mail.pp
- sudo nano dns.pp
- sudo nano client.pp

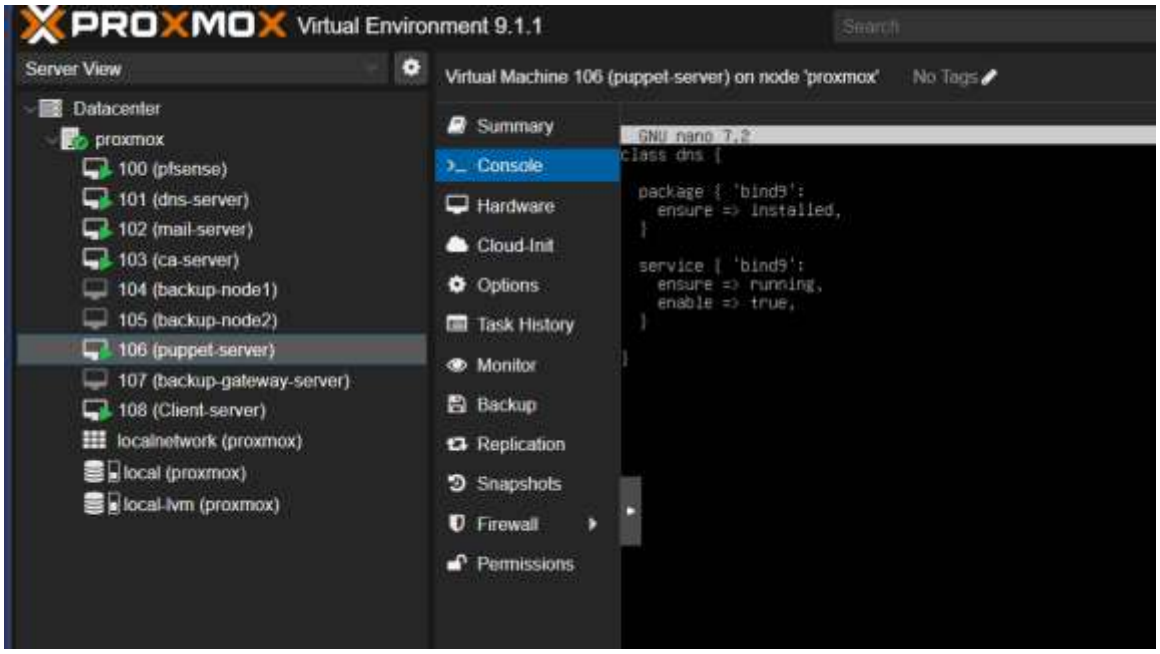
2- Configuration principale (site.pp) :



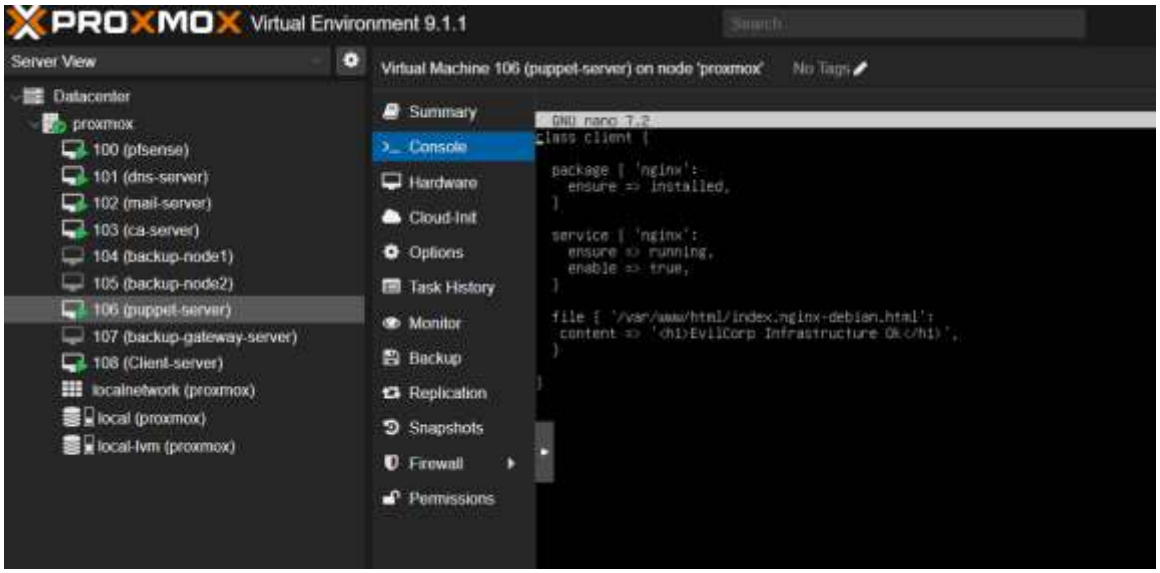
3- Configuration server mail :



4- Configuration dns server :



5- Configuration Client :

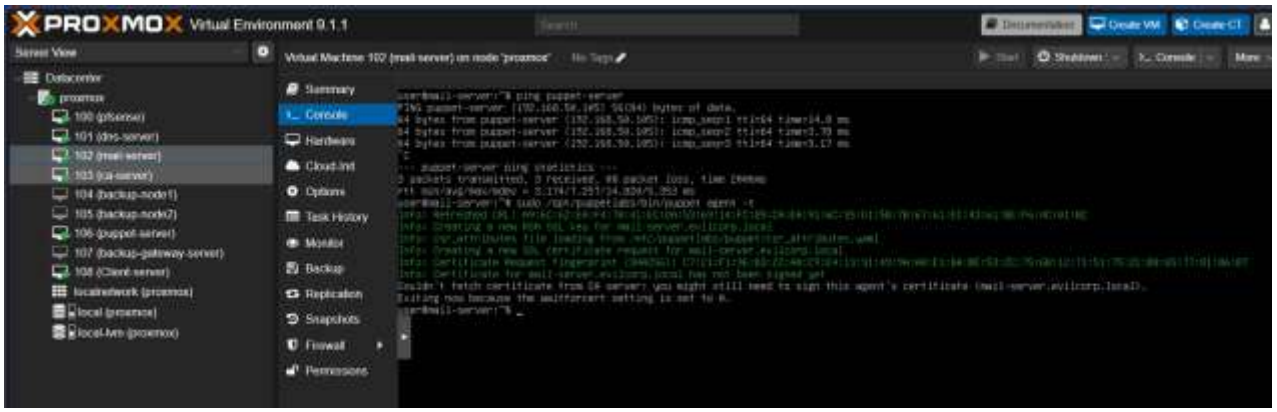


Sur chaque machine :

- Client : hostnamectl set-hostname client-server.evilcorp.local
- Mail : hostnamectl set-hostname mail-server.evilcorp.local

- Dns : hostnamectl set-hostname dns-server.evilcorp.local

Configuration Client-server :



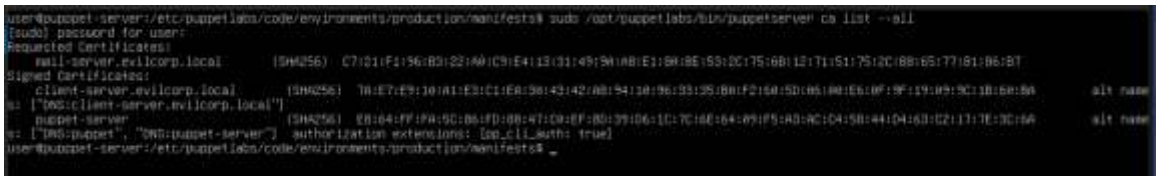
- ✓ ping OK → réseau OK
- ✓ résolution OK → /etc/hosts fonctionne
- ✓ connexion Puppet OK
- ✓ certificat généré

Donc : Communication CLIENT → PUPPET SERVER = OK

Étape suivante : Certificate has not been signed yet => Le Puppet Server doit valider le client.

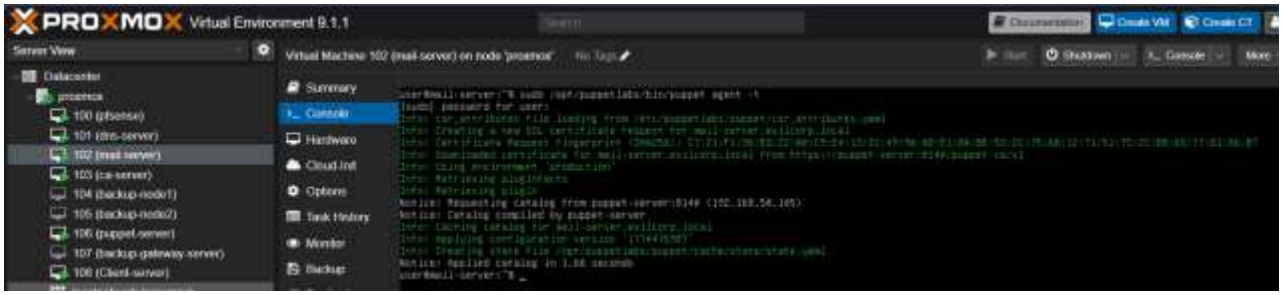
Sur puppet-server :

Le serveur de messagerie est identifié comme : mail-server.evilcorp.local.



Certificat signé :

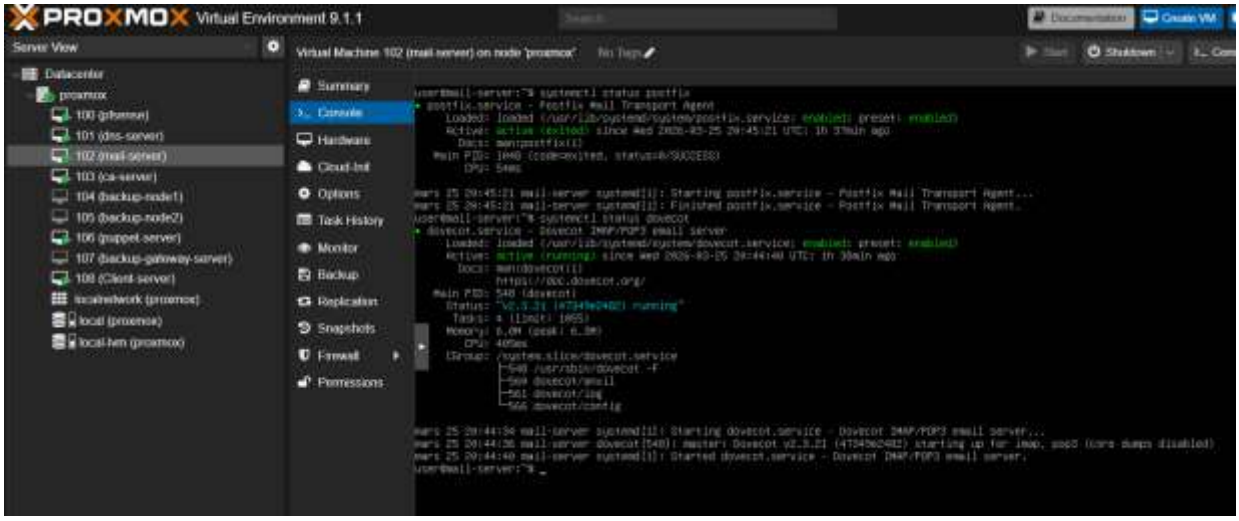
Retour sur mail-server :



Résultat :

- ✓ postfix installé
- ✓ dovecot installé
- ✓ services démarrés

Test final de vérification sur mail-server :



OBJECTIF : puppet-server gère :

- Mail Server → postfix + dovecot ✅
- DNS Server → bind9
- Clients → nginx / config

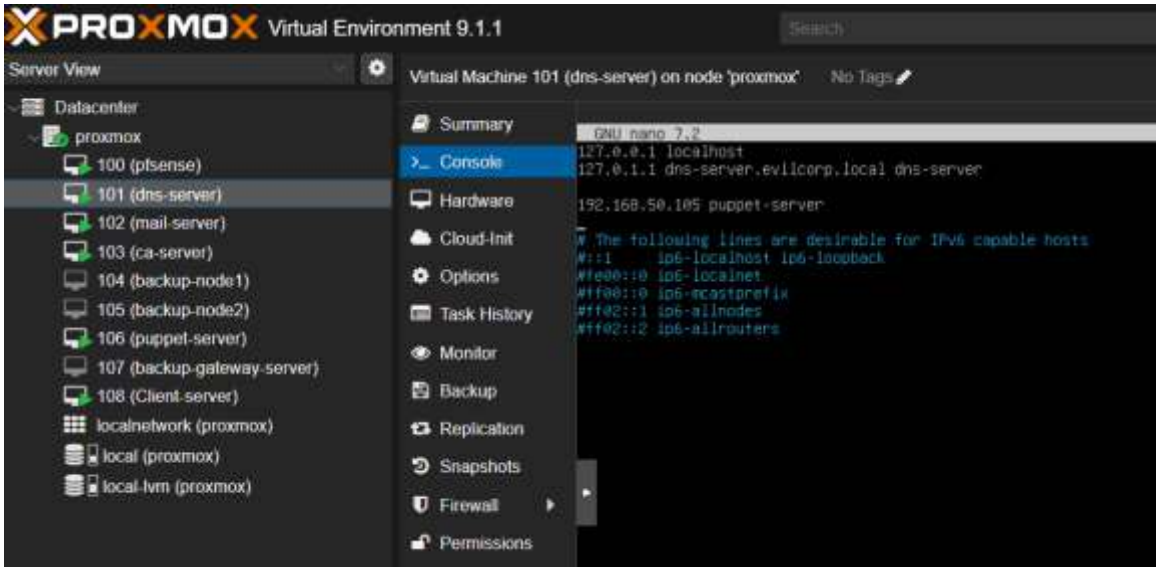
= automatisation complète du projet

- **Étape suivante :** DNS Server → bind9

OBJECTIF : Puppet doit automatiquement :

- Installer bind9
- Démarrer le service
- Configurer une zone DNS evilcorp.local

Etape 1 : Ajout dans /etc/hosts :



Étape 2 : installer Puppet agent :

- wget https://apt.puppet.com/puppet8-release-noble.deb
- sudo dpkg -i puppet8-release-noble.deb
- sudo apt update
- sudo apt install puppet-agent -y

Étape 3 : configure Agent

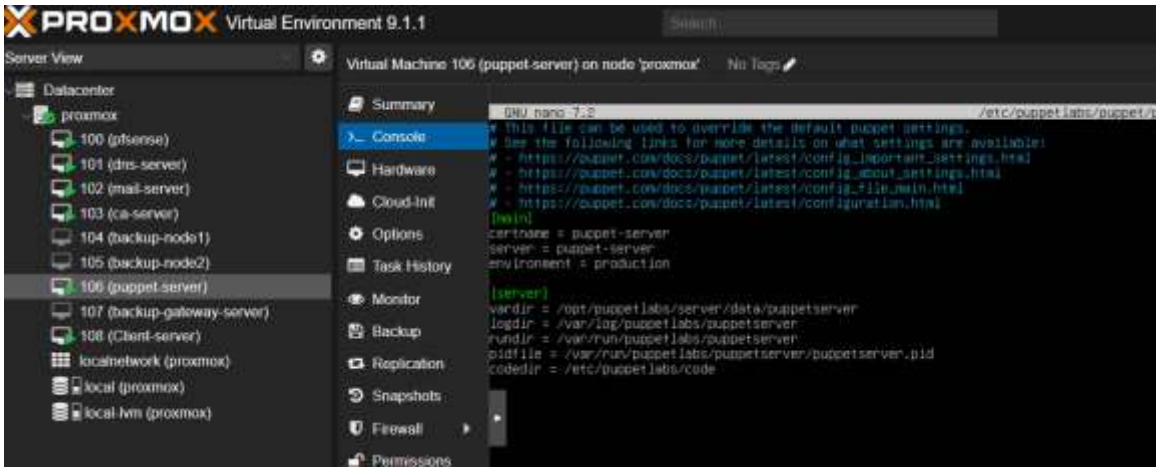
sudo nano /etc/puppetlabs/puppet/puppet.conf

Ajouter :

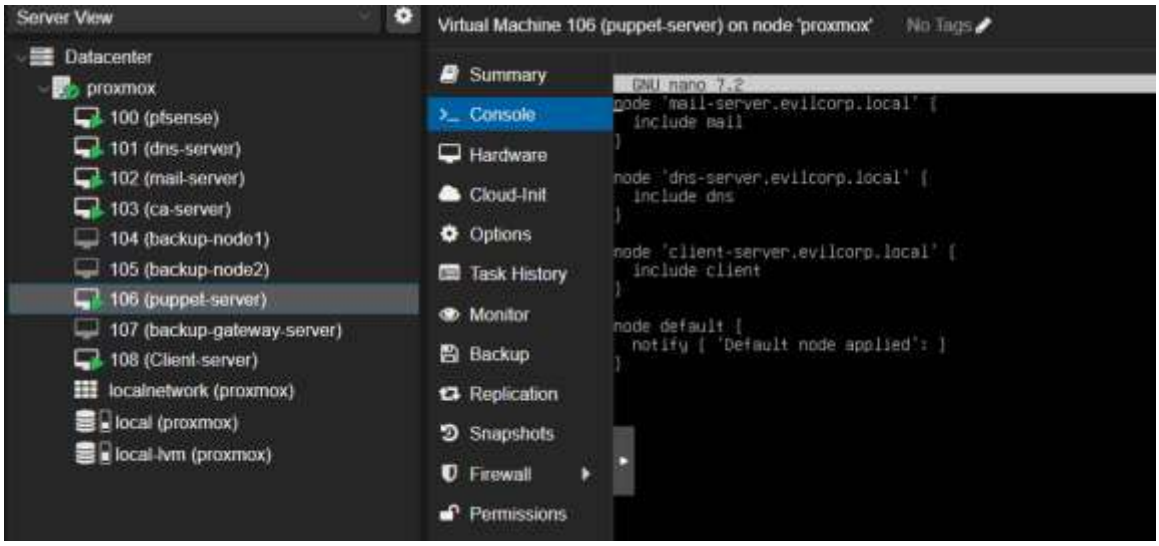
[main]

server = puppet-server

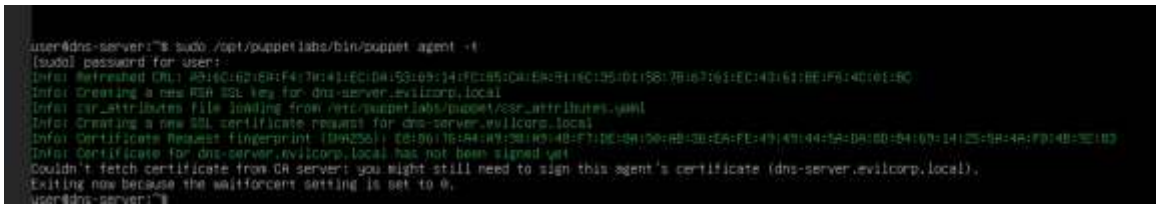
environment = production



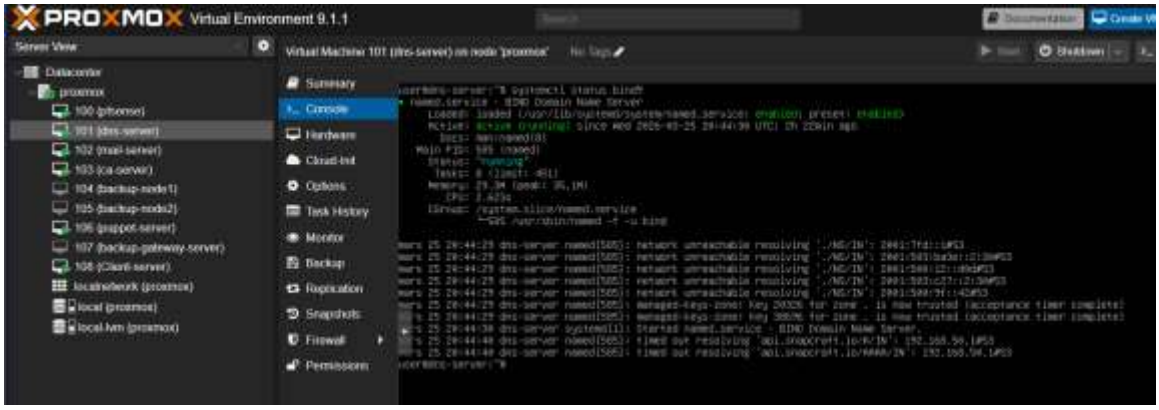
Modifier Puppet-server : Ajout Dns node



Lancer agent Dns sur dns-server :



Signature sur puppet-server :

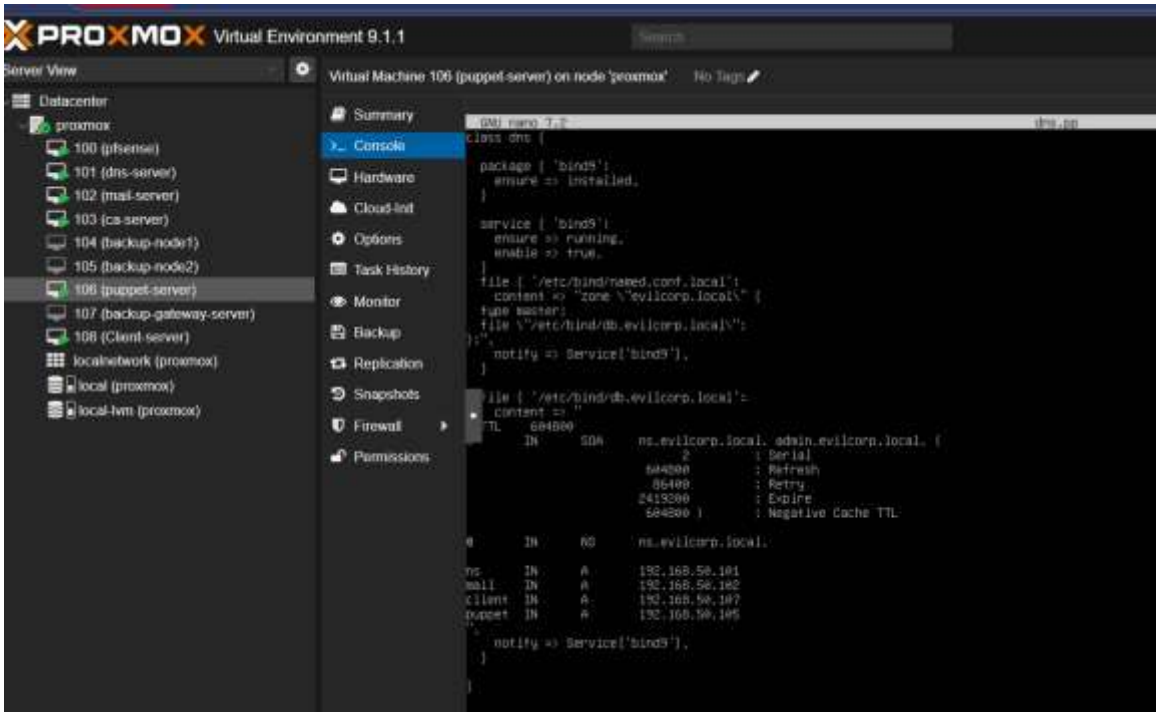


Résultat : Running, Enable tout est Ok.

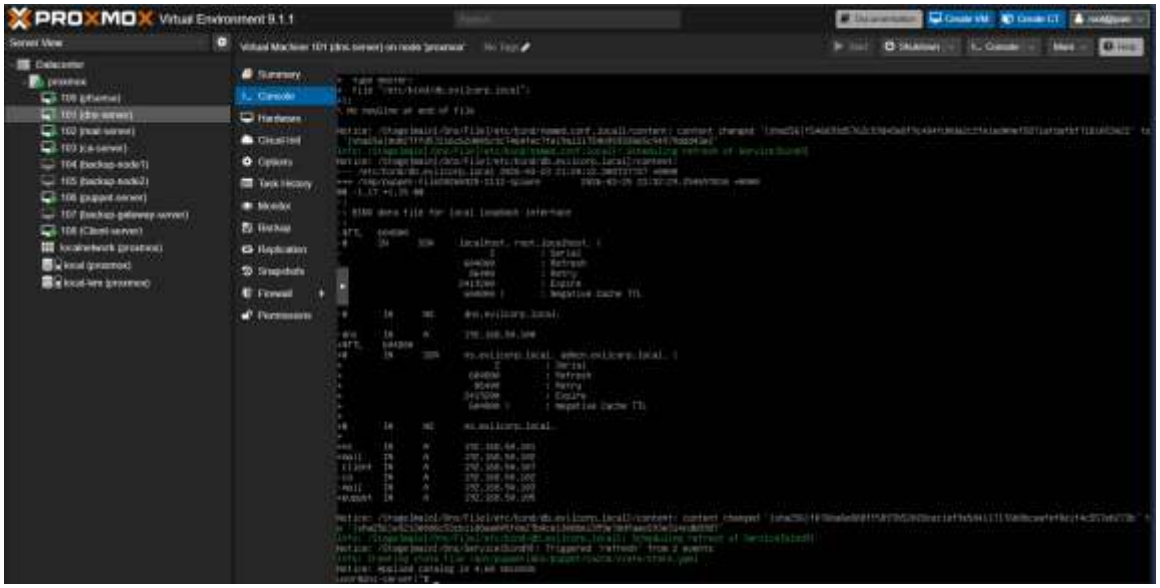
Etape suivante : Automatiser avec Puppet

- Config de la zone evilcorp.local
- Création du fichier DNS
- Démarrage bind9
- Résolution fonctionnelle

Etape 1 : mettre la configuration complète dans dns.pp



2. Relancer Puppet : sudo /opt/puppetlabs/bin/puppet agent -t

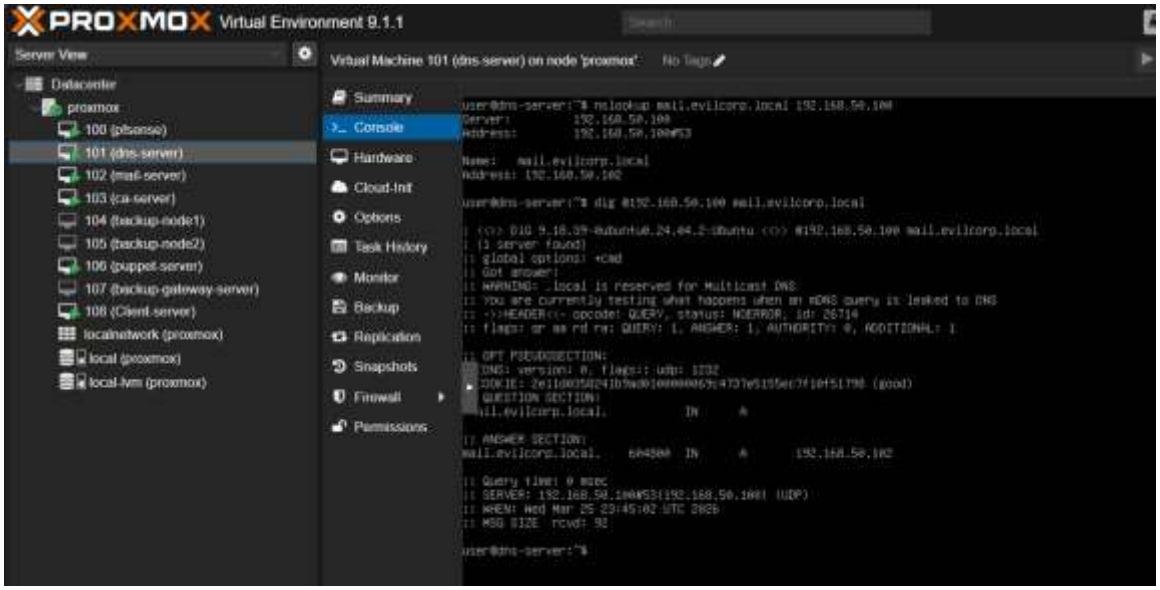


3. Vérification BIND :

```
user@dns-server:~$ systemctl status bind9
● named.service - BIND Domain Name Server
     Loaded: loaded (/usr/lib/systemd/system/named.service; enabled; preset: enabled)
     Active: active (running) since Wed 2026-03-25 23:32:31 UTC; 4min 59s ago
       Docs: man:named(8)
   Main PID: 2207 (named)
     Status: "running"
      Tasks: 8 (limit: 451)
     Memory: 24.8M (peak: 24.8M)
        CPU: 639ms
     CGroup: /system.slice/named.service
             └─2207 /usr/sbin/named -f -u bind

mars 25 23:32:31 dns-server.evilcorp.local systemd[1]: Started named.service - BIND Domain Name Server.
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: network unreachable resolving './NS/IN': 2001:500:12::d0d#53
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: network unreachable resolving './NS/IN': 2001:503:c27::2:30#53
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: network unreachable resolving './NS/IN': 2001:500:2f::f#53
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: network unreachable resolving './NS/IN': 2001:503:ba3e::2:30#53
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: network unreachable resolving './NS/IN': 2001:500:2d::d#53
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: network unreachable resolving './NS/IN': 2001:500:1::53#53
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: running
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: managed-keys-zone: Key 20326 for zone . is now trusted (acceptance timer complete)
mars 25 23:32:31 dns-server.evilcorp.local named[2207]: managed-keys-zone: Key 38696 for zone . is now trusted (acceptance timer complete)
user@dns-server:~$ _
```

Test DNS sur mail-server :



RÉSULTAT VALIDÉ :

- ✓ nslookup OK

Name: mail.evilcorp.local
Address: 192.168.50.102

```
user@dns-server:~$ nslookup mail.evilcorp.local 192.168.50.100
Server:         192.168.50.100
Address:        192.168.50.100#53

Name:   mail.evilcorp.local
Address: 192.168.50.102
```

- ✓ dig OK

ANSWER SECTION: mail.evilcorp.local. IN A 192.168.50.102

```
user@dns-server:~$ dig @192.168.50.100 mail.evilcorp.local

; <<>> DiG 9.18.39-0ubuntu0.24.04.2-Ubuntu <<>> @192.168.50.100 mail.evilcorp.local
; (1 server found)
;; global options: +cmd
;; Got answer:
;; WARNING: .local is reserved for Multicast DNS
;; You are currently testing what happens when an mDNS query is leaked to DNS
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 26714
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
;EDNS: version: 0, flags:; udp: 1232
;COOKIE: 2e11d0358241b9ad0100000069c4737e5155ec7f10f51798 (good)
;; QUESTION SECTION:
;mail.evilcorp.local.           IN      A

;; ANSWER SECTION:
mail.evilcorp.local.    604800  IN      A       192.168.50.102

;; Query time: 0 msec
;; SERVER: 192.168.50.100#53(192.168.50.100) (UDP)
;; WHEN: Wed Mar 25 23:45:02 UTC 2026
;; MSG SIZE  rcvd: 92

user@dns-server:~$
```

Donc :

✓ résolution DNS fonctionne

✓ zone evilcorp.local correcte

✓ Puppet a bien déployé

✓ bind9 fonctionne sur le réseau

OBJECTIF : puppet-server gère :

- Mail Server → postfix + dovecot ✅
- DNS Server → bind9 ✅
- Clients → nginx / config

= automatisation complète du projet

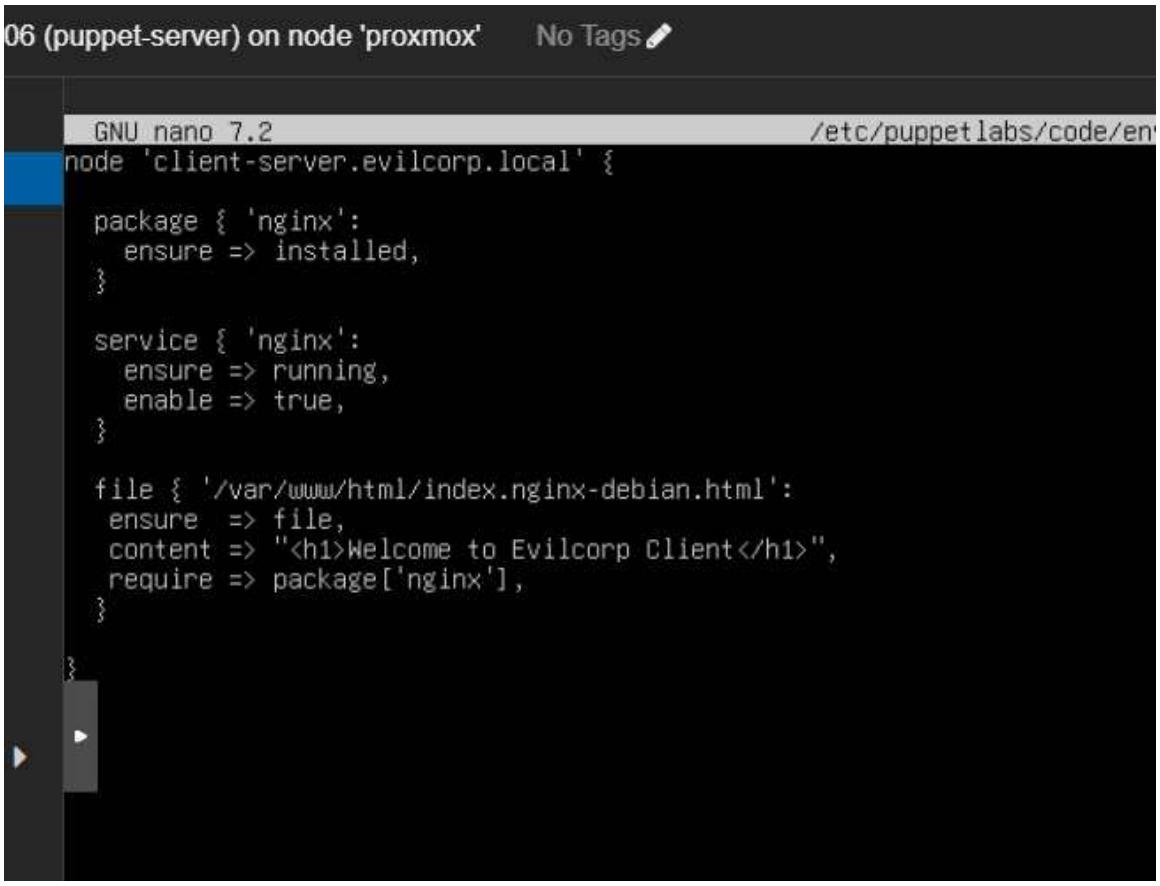
- **Étape suivante :** Clients → nginx / config

  On va faire :

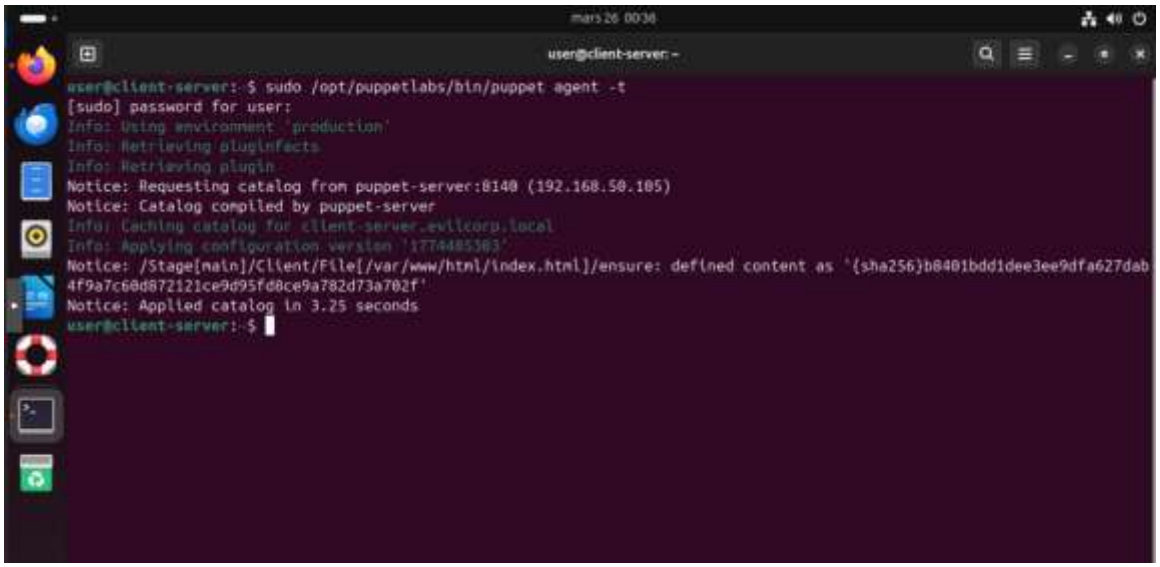
  - Installer nginx
  - Démarrer le service
  - Créer une page web custom
  - Automatiser via Puppet

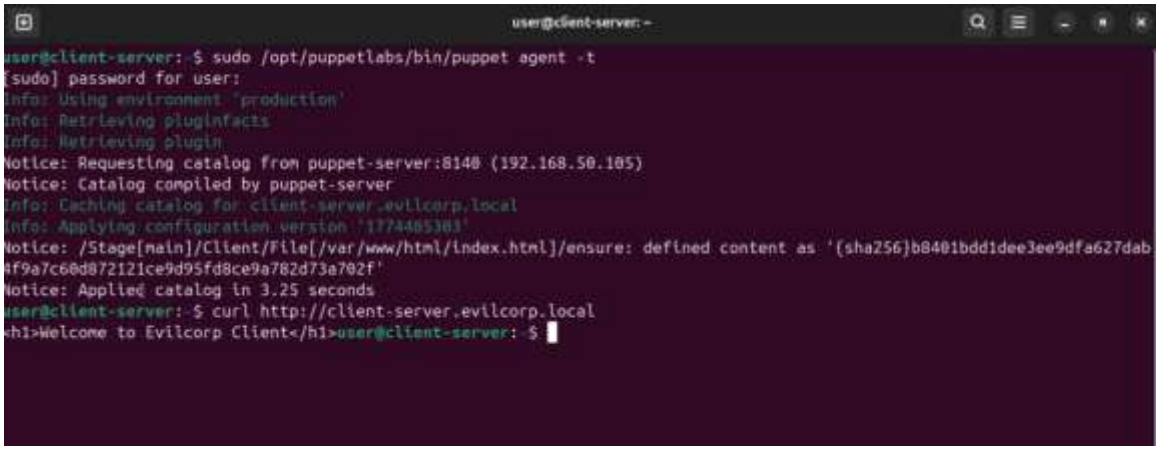
  Etape 1 : Configurer Manifest Puppet

  Sur Puppet-server : sudo nano /etc/puppetlabs/code/environments/production/manifests/client.pp



Étape 2 : Lancer l'agent





✓ Puppet fonctionne : Applied catalog in 3.25 seconds
✓ Déploiement automatique OK : File[/var/www/html/index.html] → content applied
✓ Service nginx opérationnel
✓ DNS fonctionnel

OBJECTIF : puppet-server gère :

- Mail Server → postfix + dovecot ✅
- DNS Server → bind9 ✅
- Clients → nginx / config ✅

= automatisation complète du projet ✅

Nous avons mis en place une infrastructure automatisée avec Puppet permettant de déployer dynamiquement des services sur les machines clientes. L'ensemble de l'infrastructure est piloté de manière déclarative, garantissant cohérence, reproductibilité et automatisation.

- **Backup Gateway**

Le serveur Backup Gateway joue un rôle essentiel dans la sécurisation des sauvegardes. Il sert de passerelle dédiée pour les sauvegardes entre le réseau interne et le site de sauvegarde. Cette machine garantit que seules les données de sauvegarde transitent dans le tunnel IPSec, conformément aux exigences du projet.

Fonctions principales :

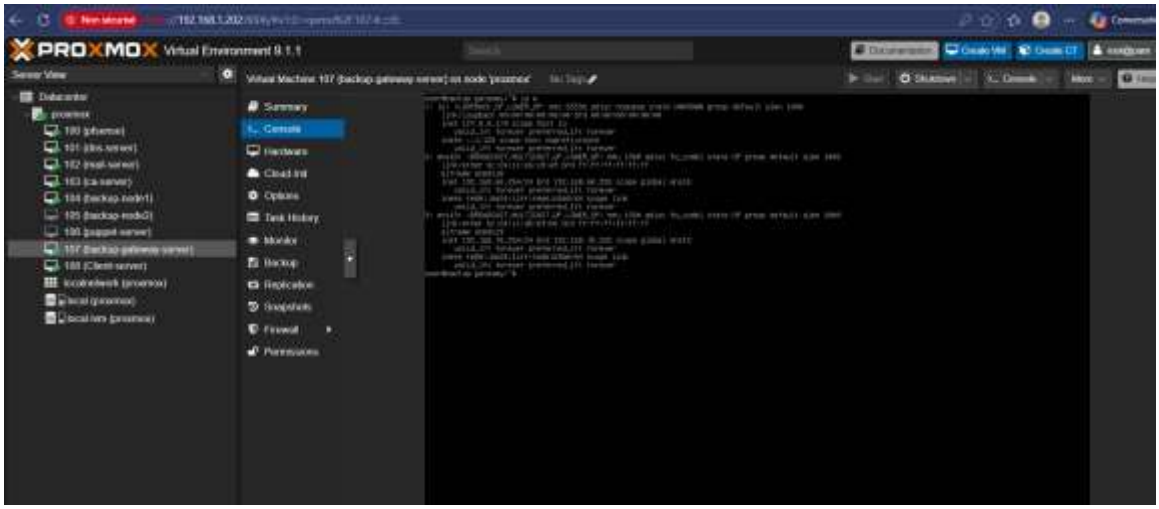
- Transmission des sauvegardes
- Gestion du tunnel IPSec
- Routage du trafic de sauvegarde
- Chiffrement des communications

Logiciels utilisés :

strongSwan (IPSec)
rsync (synchronisation des sauvegardes)

Configuration :

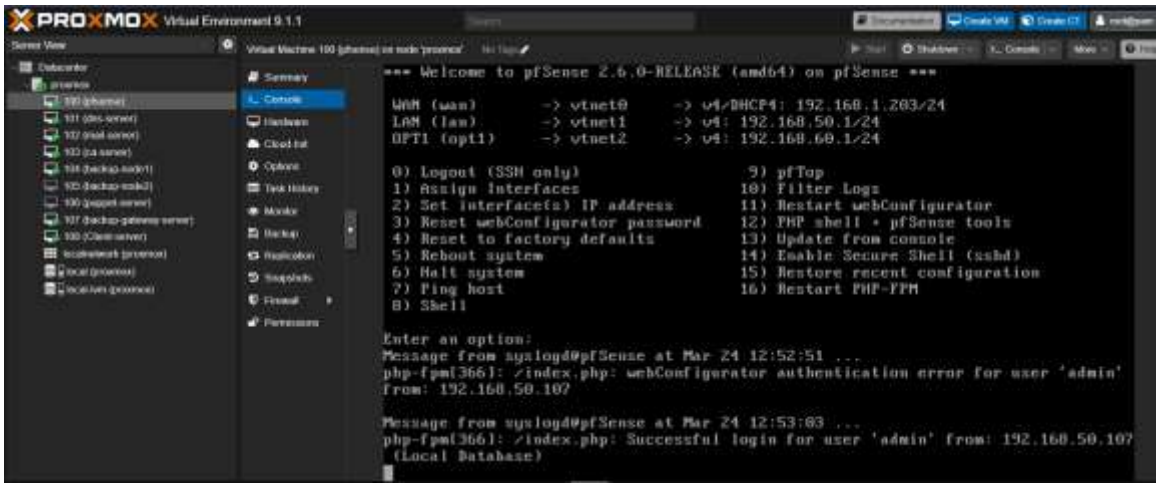
CPU : 1 vCPU
RAM : 1 GB
Disque : 10 GB

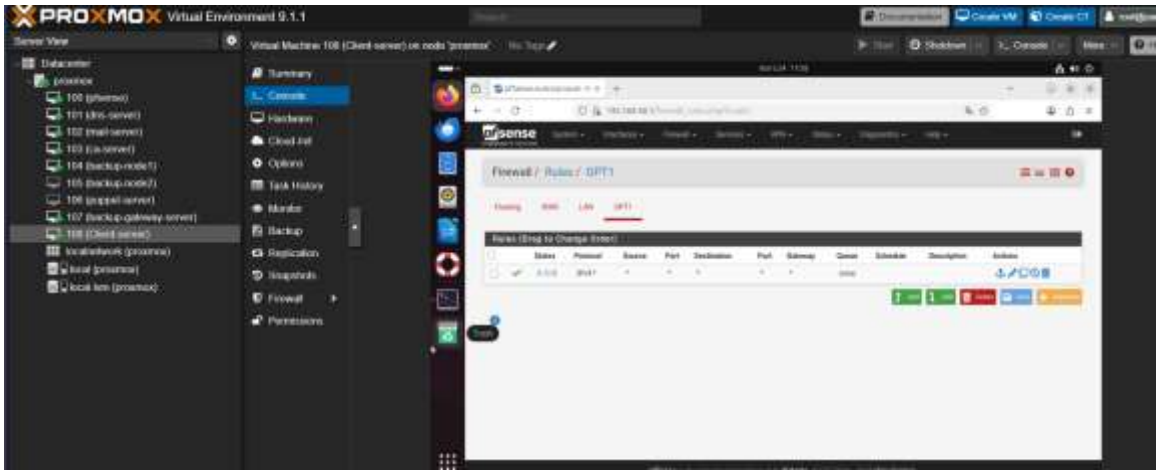


Là actuellement :

- ❌ pas de routage
- ❌ pas de route vers LAN

Activer le routage :

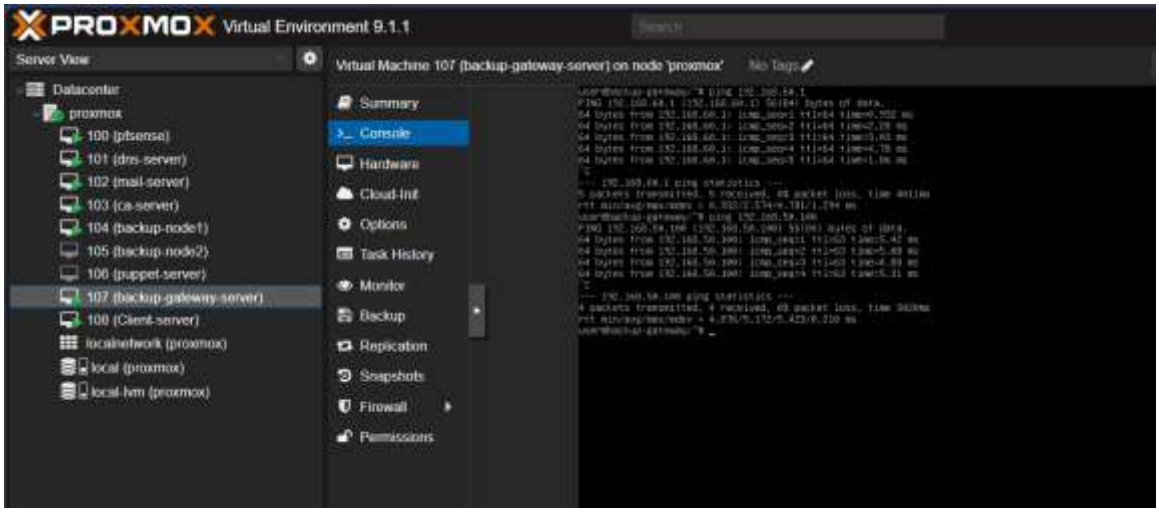




depuis backup-gateway :

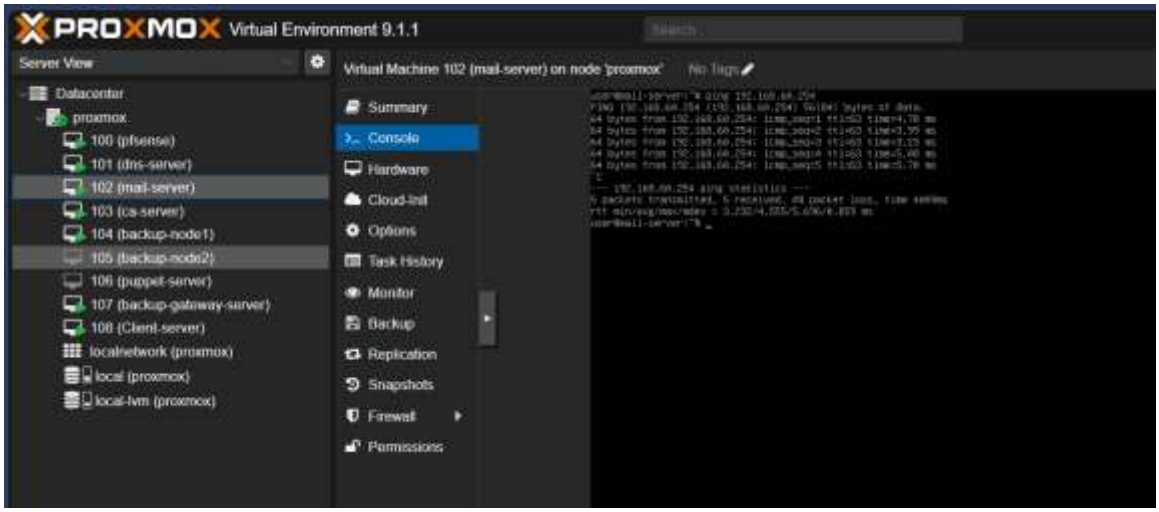
ping 192.168.60.1 # pfSense

ping 192.168.50.100 # DNS

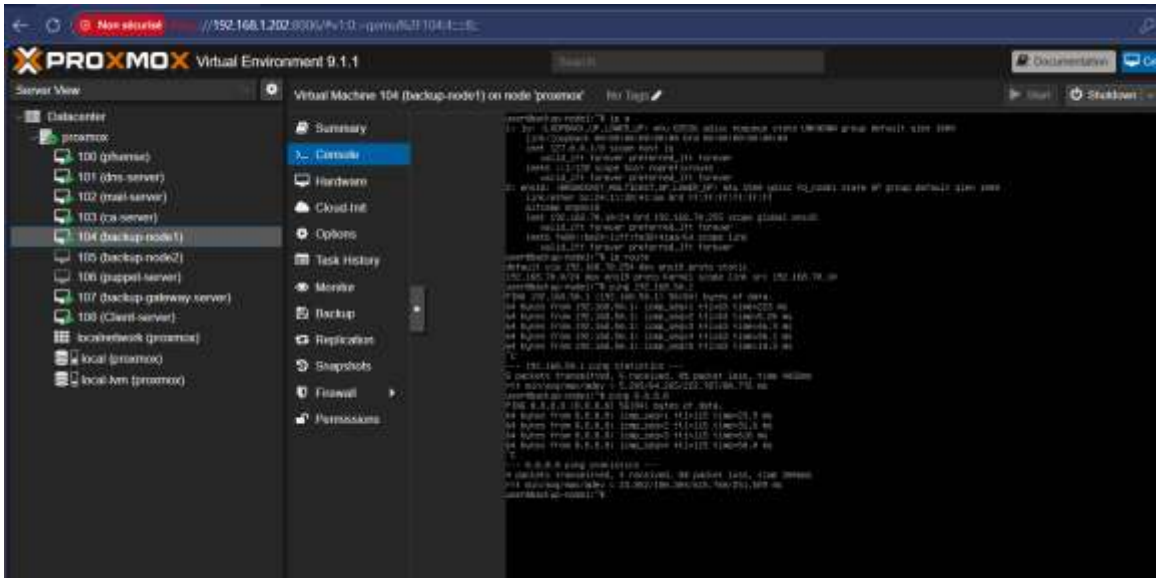


Depuis mail-server :

Ping 192.168.60.254



Depuis backup-node1 : ping 192.168.50.1, ping 8.8.8.8



- ✓ routage multi-réseaux OK
- ✓ architecture complète OK
- ✓ prêt pour IPSec

ROUTAGE COMPLET FONCTIONNEL ✅

- **Backup Node 1**

Le serveur Backup Node 1 constitue le nœud principal du système de stockage de sauvegarde. Il contient une copie des données du serveur mail. Les données sont synchronisées avec le second serveur via DRBD.

Fonctions principales :

- Stockage des sauvegardes
- Réplication DRBD
- Stockage primaire

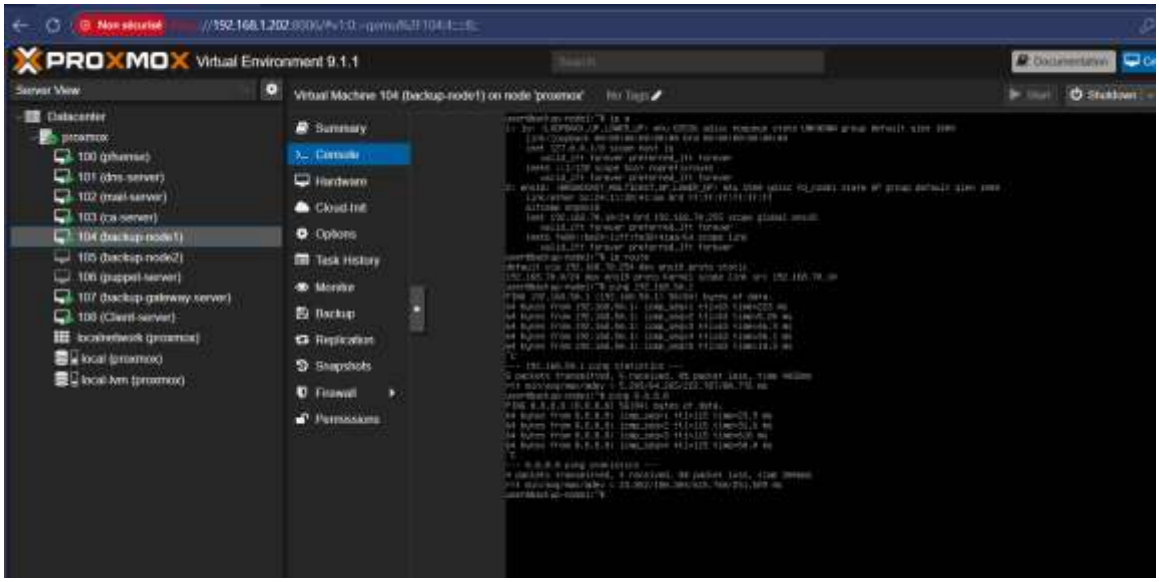
Logiciels utilisés :

DRBD
Pacemaker
Corosync

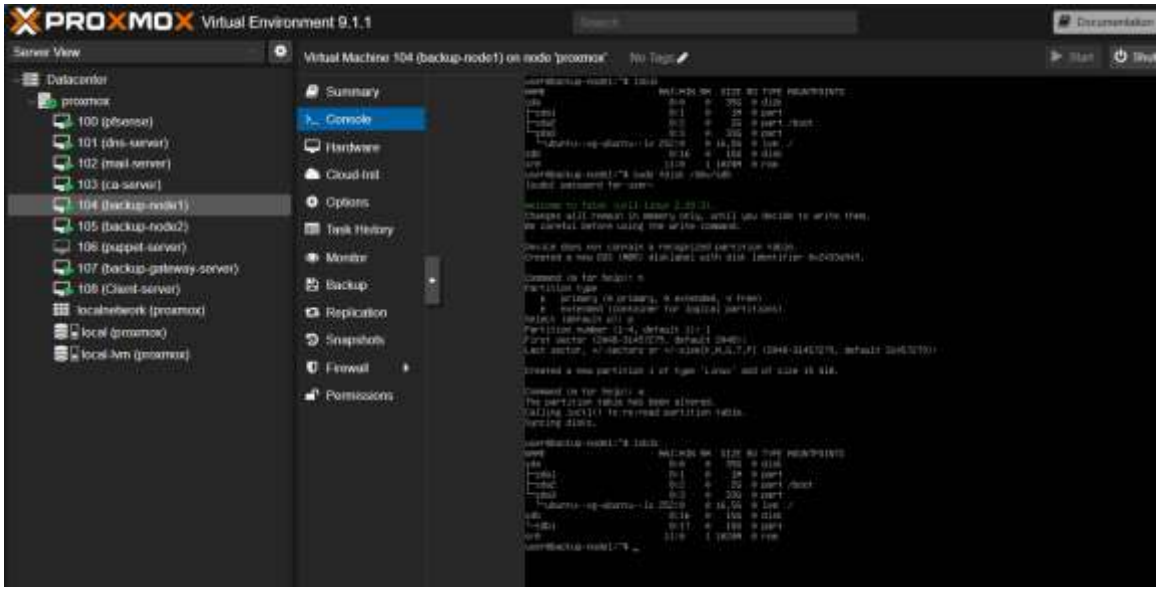
Configuration :

CPU : 2 vCPU
RAM : 2 GB
Disque : 50 GB

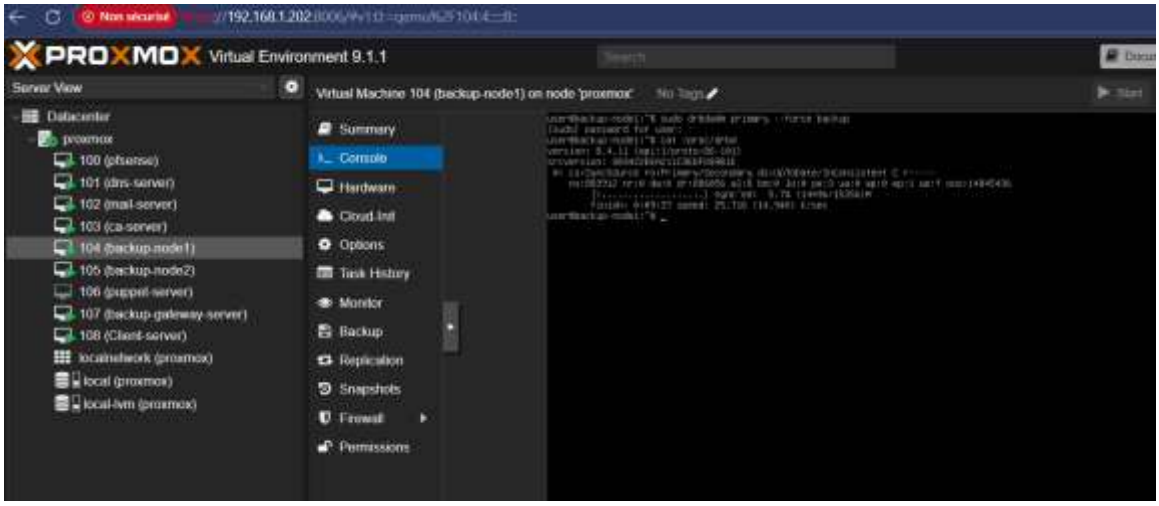
Un espace disque plus important est nécessaire pour stocker les sauvegardes.



- ✓ IP OK → 192.168.70.10
- ✓ route OK → via 192.168.70.254
- ✓ ping 192.168.50.1 OK (LAN)
- ✓ ping 8.8.8.8 OK (Internet)



- ✓ /dev/sdb détecté
- ✓ partition créée → sdb1
- ✓ taille OK (~15GB)
- ✓ fdisk terminé correctement (w)



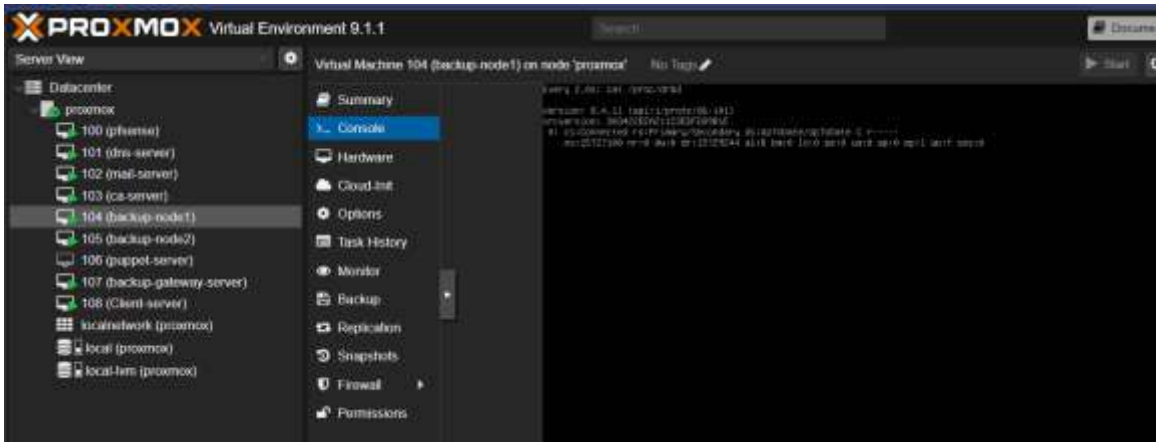
Signification :

- SyncSource : sync EN COURS

- Primary/Secondary : rôles OK
- UpToDate/Inconsistent : normal pendant sync

Conclusion : DRBD Fonctionne Parfaitement, synchronisation node 1 -> node 2

Pour surveiller la synchronisation : watch cat /proc/drbd



CONCLUSION :

DRBD = VALIDÉ À 100%

Stockage HA = OK

Réplication temps réel = OK

On vient de mettre en place : un RAID1 distribué réseau

- **Backup Node 2**

Le serveur Backup Node 2 constitue le nœud secondaire du système de sauvegarde. Il maintient une copie synchronisée des données du nœud principal. En cas de panne du premier serveur, ce serveur prend automatiquement le relais.

Fonctions principales :

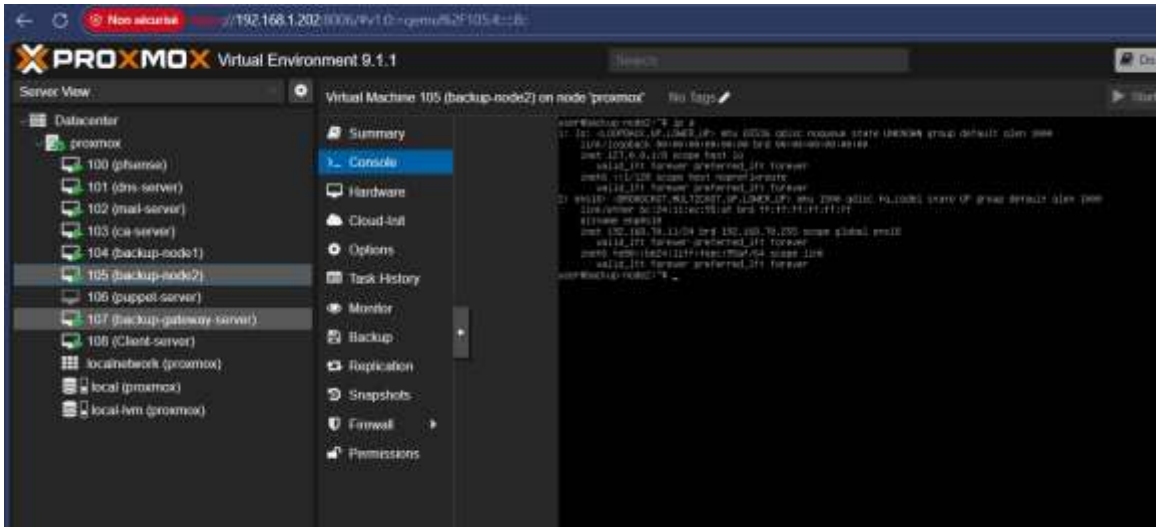
- Stockage secondaire
- Réplication DRBD
- Bascule automatique

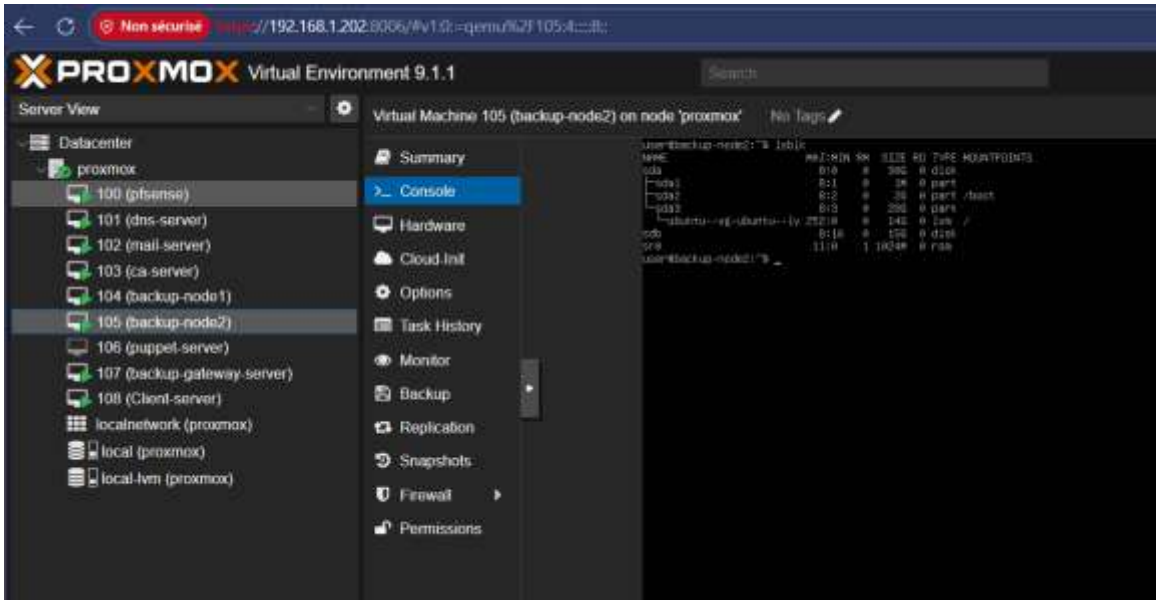
Logiciels utilisés :

DRBD
Pacemaker
Corosync

Configuration :

CPU : 2 vCPU
RAM : 2 GB
Disque : 50 GB





VALIDATION :

backup-node2 :

✓ IP → 192.168.70.11

✓ réseau OK

✓ ping internet OK

✓ ping DNS OK

Donc :

✓ node1 ↔ node2 réseau OK

✓ gateway OK

✓ pfSense OK

Tableau récapitulatif des machines virtuelles :

Machine	CPU	RAM	Disque	Rôle
pfSense	2 vCPU	2 GB	20 GB	Firewall / IPSec
DNS	1 vCPU	1 GB	10 GB	Résolution DNS
Mail	2 vCPU	2 GB	40 GB	Messagerie
CA	1 vCPU	1 GB	10 GB	Autorité de certification
Puppet	2 vCPU	2 GB	20 GB	Automatisation
Backup Gateway	1 vCPU	1 GB	10 GB	Tunnel IPSec
Backup Node 1	2 vCPU	2 GB	50 GB	Sauvegarde primaire
Backup Node 2	2 vCPU	2 GB	50 GB	Sauvegarde secondaire

Organisation réseau des machines :

Machine	Réseau
pfSense	WAN / LAN / Backup
DNS	LAN
Mail	LAN
CA	LAN
Puppet	LAN
Backup Gateway	LAN + Backup

Machine	Réseau
Backup Node 1	Backup
Backup Node 2	Backup

9. Gestion des utilisateurs :

Quatre utilisateurs seront créés sur le serveur mail.

Utilisateur	Email
CEO	ceo@evilcorp.fr
CTO	cto@evilcorp.fr
CFO	cfo@evilcorp.fr
CSO	cso@evilcorp.fr

Chaque utilisateur possède également un alias email.

10. Autorité de certification :

Une autorité de certification interne sera déployée.

Rôle :

- Créer le certificat racine
- Signer les certificats des services

Les certificats seront utilisés pour :

- Le serveur mail

- Le tunnel IPSec
- Les communications TLS

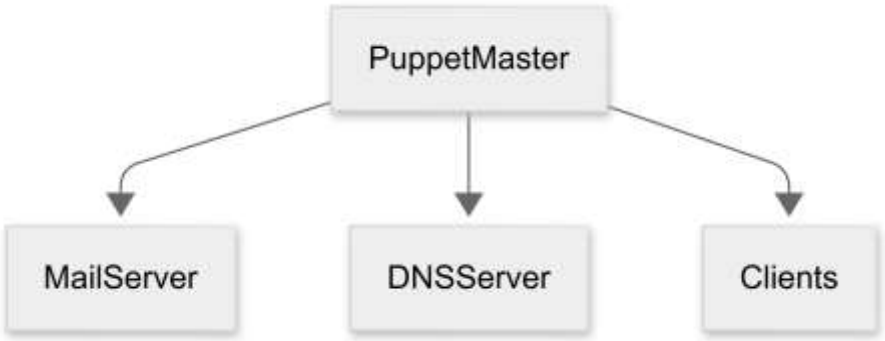
Cette autorité de certification est isolée sur une machine dédiée afin de garantir la sécurité du système.

11. Automatisation avec Puppet :

Puppet permet l'automatisation de la configuration des machines.

Fonctions principales :

- Déployer le certificat racine
- Appliquer les mises à jour de sécurité
- Gérer la configuration des clients



12. Infrastructure de sauvegarde :

La sauvegarde constitue un élément critique du projet.

Les emails doivent être copiés régulièrement vers un site distant.

Objectifs :

- Protéger les données
- Permettre la restauration
- Éviter toute perte d'informations

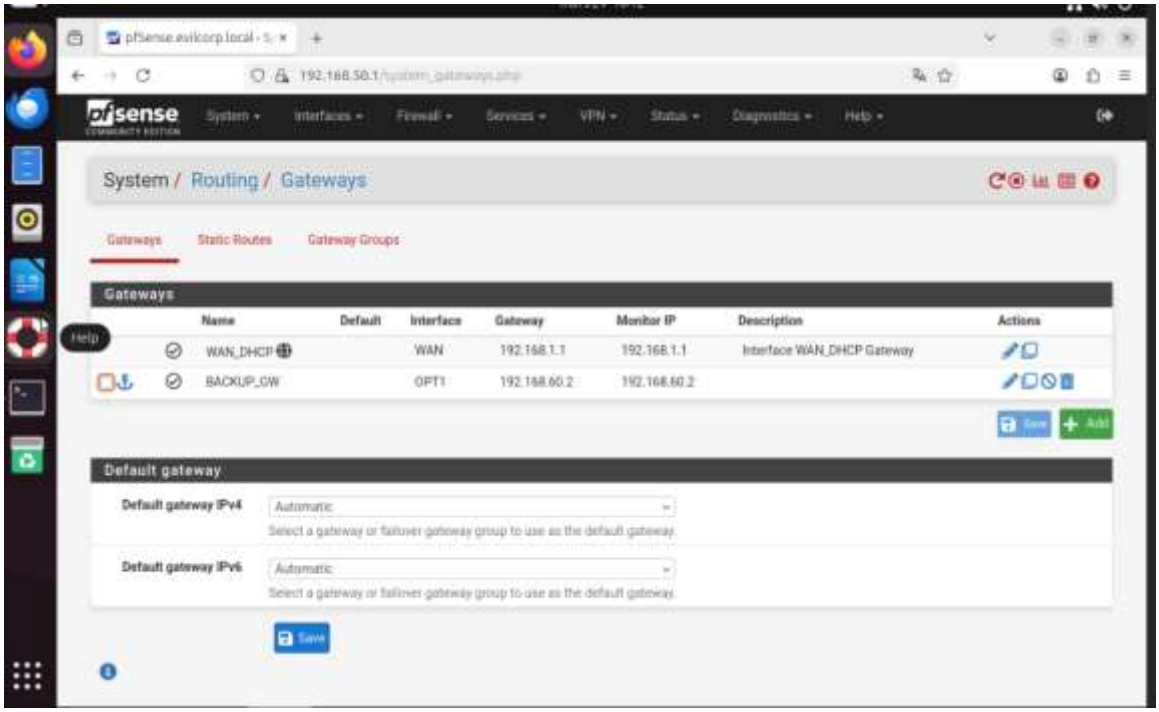
13. Tunnel IPSec :

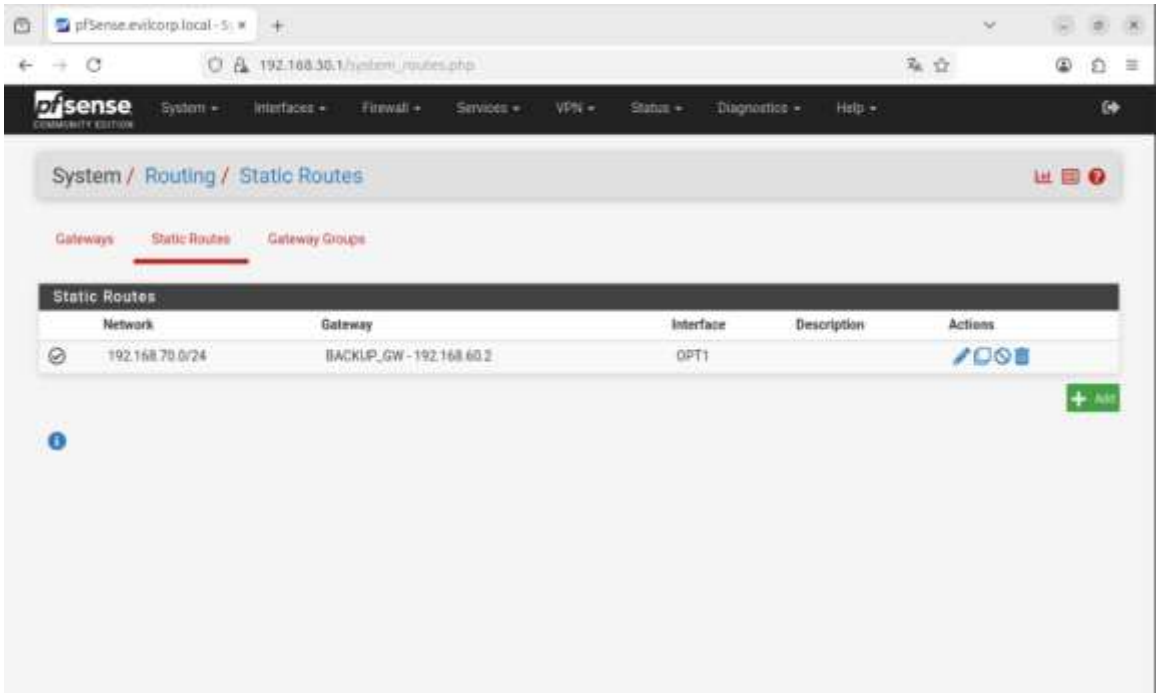
Les sauvegardes sont transmises via un tunnel **IPSec**.

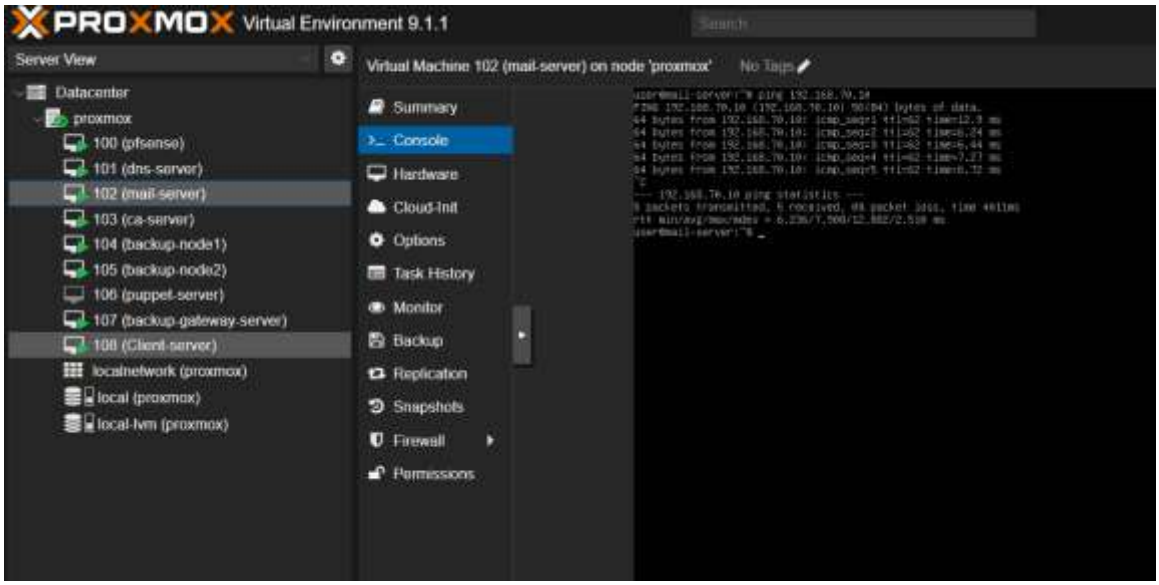
Ce tunnel assure :

- Le chiffrement du trafic
- L'authentification des machines
- La protection contre l'interception

Seules les données de sauvegarde transitent dans ce tunnel.





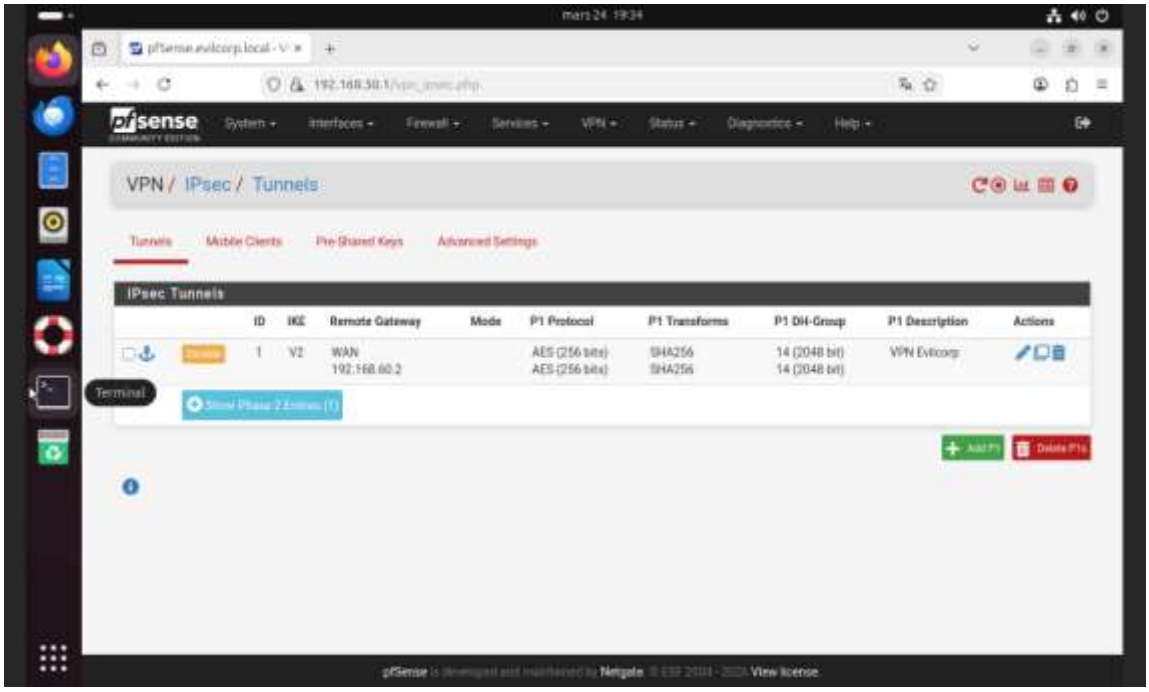


Mettre en place un VPN IPSec entre :

- pfSense (site principal)
- backup-gateway (Linux StrongSwan)

OBJECTIF VPN :

- Permettre : 192.168.50.0/24 <--> 192.168.70.0/24



14. Stockage haute disponibilité :

La haute disponibilité du stockage est assurée via DRBD. DRBD permet de répliquer les données entre deux serveurs.

Mode utilisé : active/passive
En cas de panne du serveur principal, le serveur secondaire prend automatiquement le relais.

15. Sécurisation avec SELinux :

SELinux applique un contrôle d'accès obligatoire.

Chaque processus possède un contexte de sécurité : ***user:role:type:level***

Le service mail ne pourra accéder qu'aux fichiers nécessaires à son fonctionnement.

Cela empêche les accès non autorisés au système.

16. Politique de sauvegarde :

Les sauvegardes sont :

- Automatiques
- Quotidiennes
- Chiffrées
- Stockées sur le cluster DRBD

17. Tests et validation :

Plusieurs tests permettront de vérifier le bon fonctionnement de l'infrastructure.

Tests réalisés :

1. Envoi d'un email
2. Réception d'un email
3. Vérification TLS
4. Test du tunnel IPSec

5. Synchronisation DRBD
6. Arrêt du node primaire
7. Bascule automatique vers node secondaire

Conclusion :

Cette infrastructure permet de mettre en œuvre plusieurs technologies essentielles de l'administration système et réseau :

- Virtualisation
- Sécurisation des communications
- Gestion des certificats
- Réplication des données
- Haute disponibilité
- Automatisation de la configuration

La solution proposée garantit :

- La sécurité des communications
- La protection des données
- La continuité du service
- La résilience face aux pannes

Elle représente une architecture proche de celles utilisées dans des environnements professionnels modernes.