PR05 : SIO SISR - Déploiement de services dans une DMZ

debian

Sommaires

Contexte : 3
Objectifs : 3
Cahier des charges : 3
Solution : 3
Schéma ASI : 3
Prérequis : 4
Configuration du serveur WEB et FTP : 4
Configuration du serveur FTP : 4
Installation de Proftpd : 4
Configuration du Serveur Web : 5
Installation d'Apache2 : 5
Configuration des deux intranets : 5
Configuration des Virtual hosts : 6
Mise en place du protocole HTTPS : 8
Configuration pour la DMZ : 13
Configuration du DNS : 14
Installation de BIND9 : 14
Déclaration et création des zones : 14
Configuration du serveur MariaDB : 19
Installation de MariaDB : 19
Sécurisation de l'installation MariaDB : 19
Création des bases de données : 22
Configuration des connexions distantes : 23
Configuration pour le LAN : 25
Configuration réseau des postes pour le LAN et le WAN : 26
Configuration pour le LAN : 26
Configuration pour le WAN : 26
Configuration du pare-feu PFSENSE : 27
Installation de PFSENSE : 27
Configuration des interfaces réseaux : 35
Connection depuis l'interface WEB depuis un poste du LAN : 37
Mise en place des règles de filtrages : 40
Mise en place du NAT : 40
Règles de filtrage pour le LAN : 42
Règles de filtrage pour la DMZ : 43
Règles de filtrage pour le WAN : 44
Test de Connexion : 44
Conclusion : 50

Contexte :

Dans le cadre de la sécurisation des accès aux serveurs de l'entreprise, il a été décidé de créer un réseau dit DMZ pour contenir tous les services accessibles depuis l'extérieur. La 1ère partie de ce déploiement ne portera que sur le serveur WEB qui sera déplacé du LAN vers cette DMZ

Objectifs :

- Déployer une DMZ intégrant un serveur web.
- Mettre en place un serveur de bases de données MariaDB
- Configurer des règles de filtrages via PFSENSE

Cahier des charges :

NB : pour tester les connexions depuis l'extérieur, on utilisera un poste configuré dans le même réseau que l'interface WAN de votre Firewall.

Phase 1 :

- Autoriser les accès au serveur Web depuis Internet et depuis le LAN
- Autoriser les accès FTP sur le serveur de la DMZ depuis le LAN
- Autoriser les accès Internet depuis le LAN et la DMZ en passant par le Firewall.
- Interdire tout accès au LAN depuis l'Internet ou la DMZ.

Phase 2 :

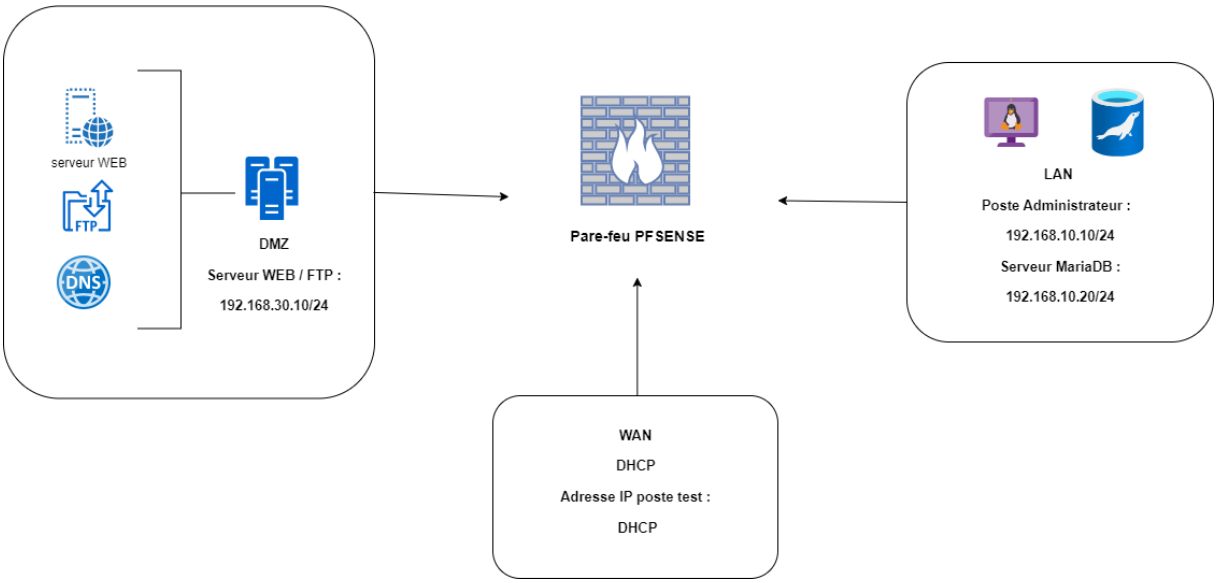
On considère que le site web migré la DMZ est associé à une de BDD dans le LAN.

- Installer un serveur de base de données MariaDB dans le LAN et, autoriser les requêtes SQL du serveur WEB vers le serveur MariaDB

Solution :

Pour répondre à la demande, je vais mettre en place plusieurs VM sous Debian afin d'héberger mon serveur web, mon serveur DNS et mon serveur de base de données. Je vais également mettre en place différentes VM sous Debian afin de tester les connexions depuis la DMZ et le LAN, qui seront sécurisées par un pare-feu.

Schéma ASI :



Prérequis :

Tout d'abord nous allons devoir installer puis paramétrer une machine virtuelle sous Debian qui nous servira de Serveur WEB (Apache2), de serveur DNS (BIND9) et de serveur FTP (Proftpd).

Puis une autre machine virtuelle pour le serveur MariaDB qui sera dans le LAN.

Ensuite on va configurer deux VM sous Debian : une pour le réseau LAN et une autre pour le réseau WAN.

Et enfin nous allons installer et configurer PFSENSE.

Configuration du serveur WEB et FTP :

Configuration du serveur FTP :

Nous devons installer au préalable ProFTPD sur notre VM SERVEUR Debian11

Installation de Proftpd :

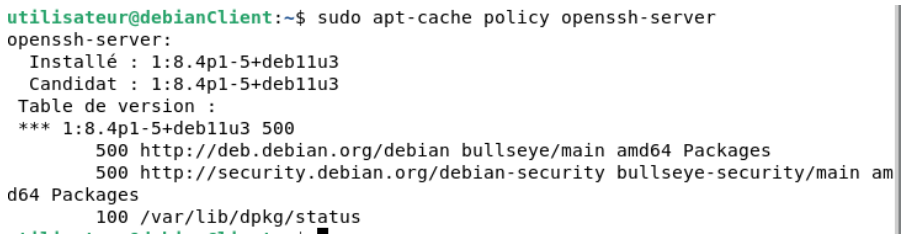
- sudo apt-get update
- sudo apt install proftpd

Une fois installé nous pouvons choisir de créer des utilisateurs mais nous allons nous connecter en mode utilisateur pour nos tests.

FTP n'est pas un protocole sécurisé. Pour éviter la transmission d'informations en clair, il est nécessaire de crypter les données en transit. Nous utiliserons donc principalement le protocole SFTP.

Pour cela, nous devrons installer OpenSSH sur les serveurs ainsi que sur les postes clients via la commande :

- sudo apt install openssh-server



Configuration du Serveur Web :

On va devoir configurer Apache 2 avec nos 2 intranets et mettre en place le protocole HTTPS

Dans un premier temps faire sudo apt-get update afin d'avoir les dernières versions disponibles (il faut le faire avant toutes installations sous Debian)

Installation d'Apache2 :

- sudo apt install Apache2

On constate donc qu'Apache2 est correctement installé.

Configuration des deux intranets :

On va créer deux dossiers MBWay et DigitalSchool :

Aller dans le dossier cd /var/www/html est créé deux dossiers avec la commande mkdir :

```
utilisateur@debianServeur:~$ cd /var/www/html/
utilisateur@debianServeur:/var/www/html$ ls -l
total 20
drwxr-xr-x 2 root root  4096 15 avril 12:46 digitalschool
-rw-r--r-- 1 root root 10701  7 avril 21:19 index.html
drwxr-xr-x 2 root root  4096 13 avril 18:18 mbway
utilisateur@debianServeur:/var/www/html$
```

Pour chaque dossier nous allons créer un fichier index.html que nous allons ensuite configurer :

```
utilisateur@debianServeur:/var/www/html$ cd mbway
utilisateur@debianServeur:/var/www/html/mbway$ ls -l
total 4
-rw-r--r-- 1 root root 123 13 avril 18:18 index.html
utilisateur@debianServeur:/var/www/html/mbway$ cd ..
utilisateur@debianServeur:/var/www/html$ cd digitalschool
utilisateur@debianServeur:/var/www/html/digitalschool$ ls -l
total 8
-rw-r--r-- 1 root root 131 14 avril 11:13 index.html
```

Fichier index.html pour DigitalSchool :

```
utilisateur@debianServeur:/var/www/html/digitalschool$ cat index.html
<!DOCTYPE html>
<html>
<body>
        <h1>Bienvenue sur le site de Digitalschool</h1>
        <p>Ceci est la page d'acceuil.</p>
</body>
</html>
```

On va créer le même type pour MBWay.

Nous avons donc créé un dossier où nous importerons nos fichiers html, PHP, JavaScript etc… respectifs à chaque site. Ici nous n'avons déposé qu'un « index.html » pour le moment.

Configuration des Virtual hosts :

Grâce à l'étape précédente nous avons créé nos intranets.

De ce fait pour y accéder nous sommes obligés d'écrire : http://AdresseIP/mbway/ pour accéder au site d'MBWay par exemple.

Pour corriger cela nous allons donc configurer des Virtual Host pour accéder à nos sites depuis l'adresse http://www.mbway.lan

On va aller dans le dossier /etc/apache2/sites-available/

On a un fichier 000-default.conf avec une configuration par default que l'on va copier (commande cd) et créé deux fichiers un pour MBWay et un autre pour DigitalSchool.

```
utilisateur@debianServeur:~$ cd /etc/apache2/sites-available/
utilisateur@debianServeur:/etc/apache2/sites-available$ ls -l
total 36
-rw-r--r-- 1 root root 1332  2 avril  2023 000-default.conf
-rw-r--r-- 1 root root 6387 14 avril 19:53 default-ssl.conf
-rw-r--r-- 1 root root 1351 13 avril 15:31 digitalschool.conf
-rw-r--r-- 1 root root 6390 16 avril 09:20 digitalschool-ssl.conf
-rw-r--r-- 1 root root 1636 14 avril 19:38 mbway.conf
-rw-r--r-- 1 root root 6374 16 avril 09:19 mbway-ssl.conf
utilisateur@debianServeur:/etc/apache2/sites-available$
```

On va ensuite les configurer :

```
utilisateur@debianServeur:/etc/apache2/sites-available$ cat mbway.conf
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port that
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) this
        # value is not decisive as it is used as a last resort host regardless.
        # However, you must set it for any further virtual host explicitly.
        ServerName www.mbway.lan

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html/mbway
```

```
utilisateur@debianServeur:/etc/apache2/sites-available$ cat digitalschool.conf
<VirtualHost *:80>
        # The ServerName directive sets the request scheme, hostname and port
        # the server uses to identify itself. This is used when creating
        # redirection URLs. In the context of virtual hosts, the ServerName
        # specifies what hostname must appear in the request's Host: header to
        # match this virtual host. For the default virtual host (this file) th
        # value is not decisive as it is used as a last resort host regardless
        # However, you must set it for any further virtual host explicitly.
        ServerName www.digitalschool.lan

        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html/digitalschool
```

On modifie ServerName par le nom de nos serveurs : www.mbway.lan et www.digitalschool.lan.

On lui indique la route avec DocumentRoot : /var/www/html/mbway et /var/www/html/digitalschool .

On va activer nos sites avec la commande a2ensite + le nom du virtual Host

Maintenant si on ajoute www.mbway.lan en alias à notre serveur dans le fichier /etc/hosts de notre client, vous pouvez voir la page d'accueil de notre site dans notre navigateur.

On vérifie que nos sites fonctionnent :

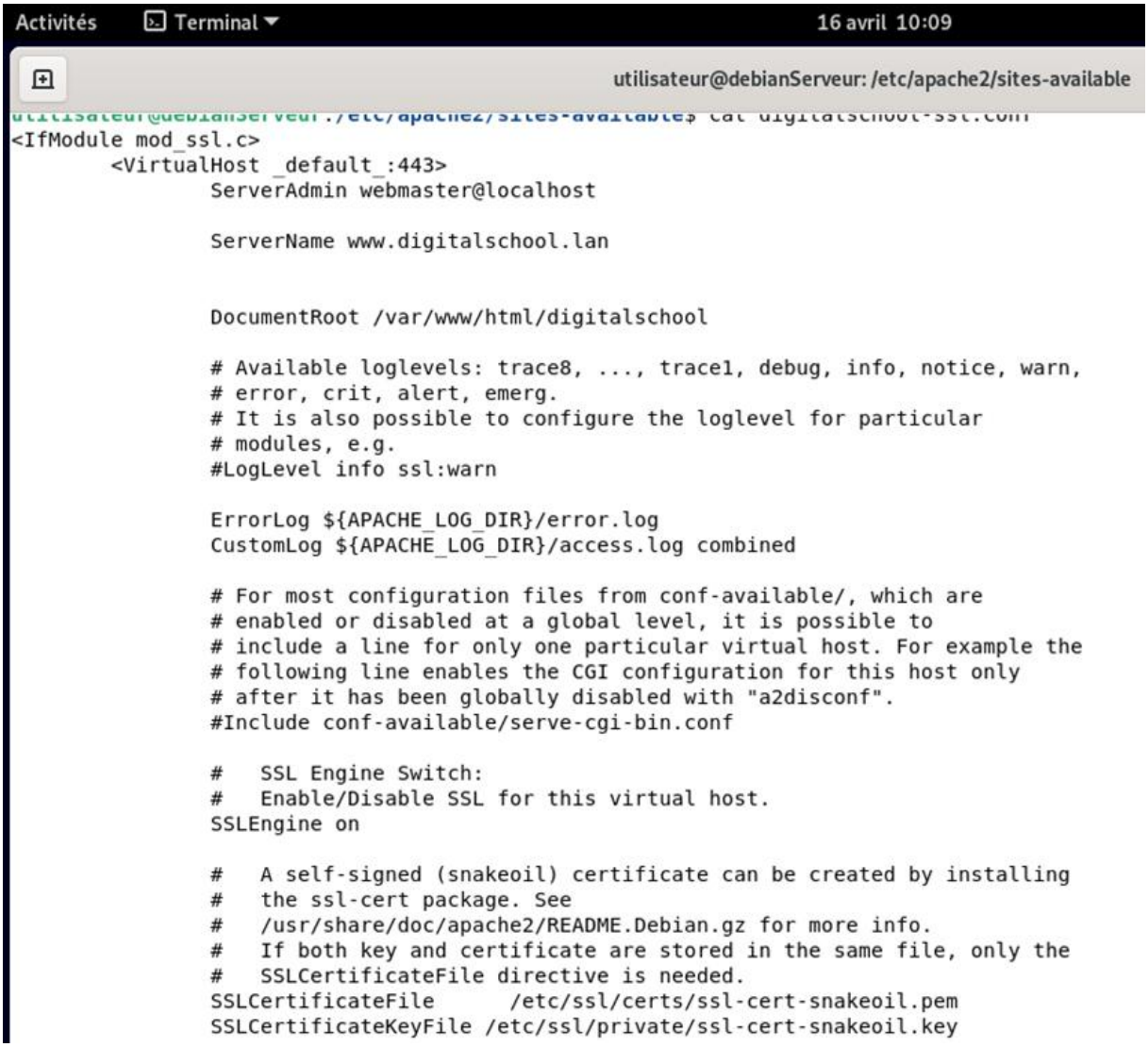
Mise en place du protocole HTTPS :

La connexion à nos sites intranets ne se font de base qu'en HTTP ce qui n'est absolument pas sécurisé. Pour éviter de subir une attaque informatique de type « Man in the middle » nous devons crypter les données qui transitent par le biais du protocole HTTPS.

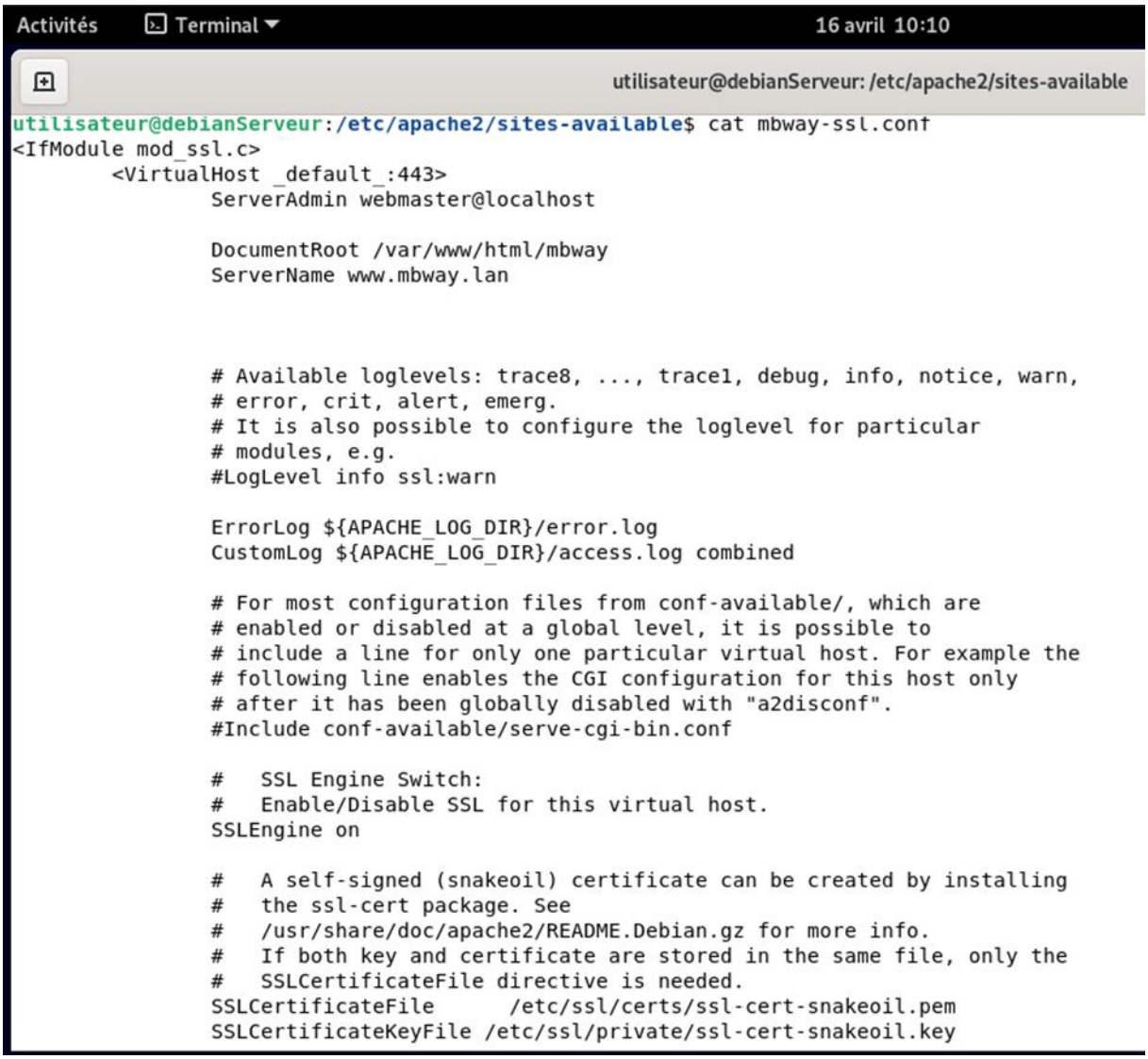
On a un fichier avec une configuration déjà présente sous Debian :

```
utilisateur@debianServeur:/etc/apache2/sites-available$ ls -l
total 36
-rw-r--r-- 1 root root 1332  2 avril  2023 000-default.conf
-rw-r--r-- 1 root root 6387 14 avril 19:53 default-ssl.conf
-rw-r--r-- 1 root root 1351 13 avril 15:31 digitalschool.conf
-rw-r--r-- 1 root root 6390 16 avril 09:20 digitalschool-ssl.conf
-rw-r--r-- 1 root root 1636 14 avril 19:38 mbway.conf
-rw-r--r-- 1 root root 6374 16 avril 09:19 mbway-ssl.conf
utilisateur@debianServeur:/etc/apache2/sites-available$
```

On va donc copier le fichier default-ssl et créer deux fichiers pour chaque site puis les configurer :




```
<IfModule mod_ssl.c>
        <VirtualHost _default_:443>
                ServerAdmin webmaster@localhost

                ServerName www.digitalschool.lan


                DocumentRoot /var/www/html/digitalschool

                # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
                # error, crit, alert, emerg.
                # It is also possible to configure the loglevel for particular
                # modules, e.g.
                #LogLevel info ssl:warn

                ErrorLog ${APACHE_LOG_DIR}/error.log
                CustomLog ${APACHE_LOG_DIR}/access.log combined

                # For most configuration files from conf-available/, which are
                # enabled or disabled at a global level, it is possible to
                # include a line for only one particular virtual host. For example the
                # following line enables the CGI configuration for this host only
                # after it has been globally disabled with "a2disconf".
                #Include conf-available/serve-cgi-bin.conf

                #   SSL Engine Switch:
                #   Enable/Disable SSL for this virtual host.
                SSLEngine on

                #   A self-signed (snakeoil) certificate can be created by installing
                #   the ssl-cert package. See
                #   /usr/share/doc/apache2/README.Debian.gz for more info.
                #   If both key and certificate are stored in the same file, only the
                #   SSLCertificateFile directive is needed.
                SSLCertificateFile      /etc/ssl/certs/ssl-cert-snakeoil.pem
                SSLCertificateKeyFile /etc/ssl/private/ssl-cert-snakeoil.key
```



On donne le nom de notre serveur :

ServerName www.mbway.lan et ServerName www.digitalschool.lan ainsi que la route avec DocumentRoot /var/www/html/mbway et /var/www/html/digitalschool .

On va activer a2enmod ssl puis a2ensite default-ssl

On va activer nos sites avec la commande a2ensite + le nom du virtual Host

Puis redemarrer apache2 : service apache2 reload

La recommandation est de créer/acheter un certificat pour chaque site plutôt que d'utiliser la configuration de base d'Apache2.

C'est donc ce que nous allons faire :

Dans un premier temps nous allons verifier que le module SSL est bien activée :

- sudo a2enmod ssl

Nous allons ensuite générer nos clés privées et des CSR(certificate signing request) :

Pour Mbway : openssl req -newkey rsa:2048 -nodes -keyout /etc/ssl/private/mbway.key -out /etc/ssl/certs/mbway.csr

Et pour Digitalschool : openssl req -newkey rsa:2048 -nodes -keyout /etc/ssl/private/digitalschool.key -out /etc/ssl/certs/ digitalschool.csr

Nous devrons répondre à une suite de question :

```
Mot de passe :
root@debian11:/home/utilisateur# openssl req -newkey rsa:2048 -nodes -keyout /etc/ssl/private/mbway.key -out /etc/ssl/certs/mbway.csr
Generating a RSA private key
.....+++++
.......+++++
writing new private key to '/etc/ssl/private/mbway.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:fr
State or Province Name (full name) [Some-State]:paris
Locality Name (eg, city) []:paris
Organization Name (eg, company) [Internet Widgits Pty Ltd]:thomasit
Organizational Unit Name (eg, section) []:it
Common Name (e.g. server FQDN or YOUR name) []:thomas
Email Address []:thomas.fr

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:root
An optional company name []:
root@debian11:/home/utilisateur#
```

Nous allons enfin passer a la création de nos certificats auto-signés :

Pour Mbway : sudo openssl req x509 -req -days 365 -in /etc/ssl/certs/mbway.csr -signkey /etc/ssl/private/mbway.key -out /etc/ssl/certs/mbway.crt

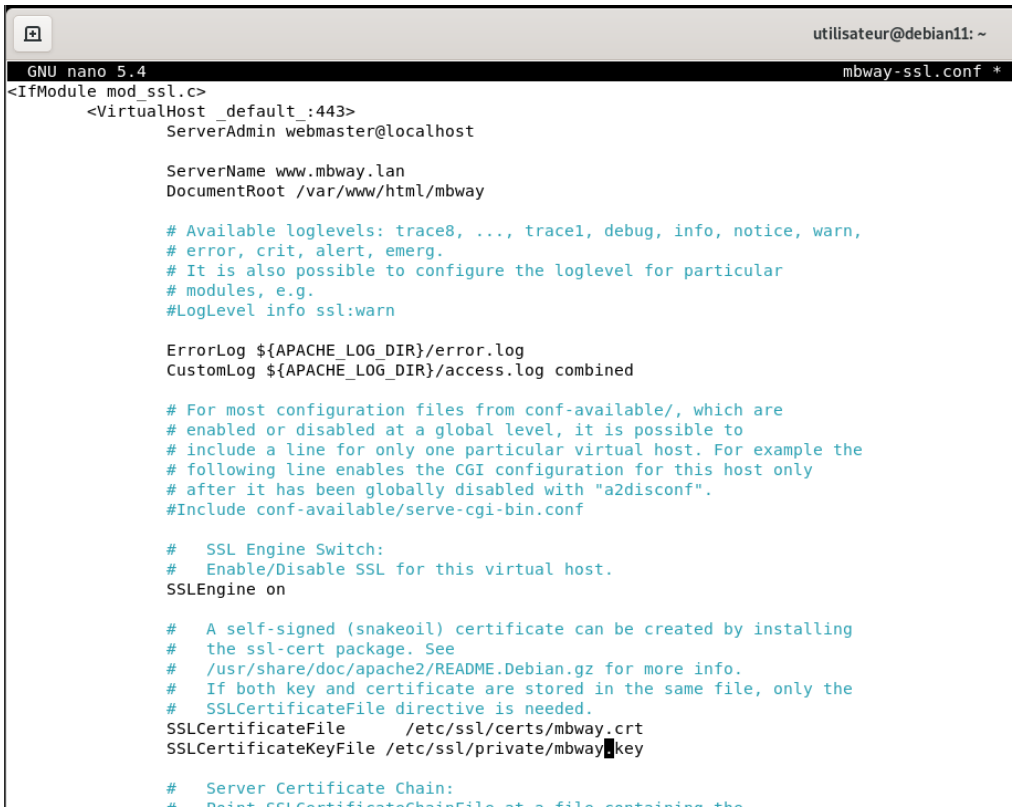
Et pour digitalschool : sudo openssl req x509 -req -days 365 -in /etc/ssl/certs/digitalschool.csr -signkey /etc/ssl/private/digitalschool.key -out /etc/ssl/certs/digitalschool.crt

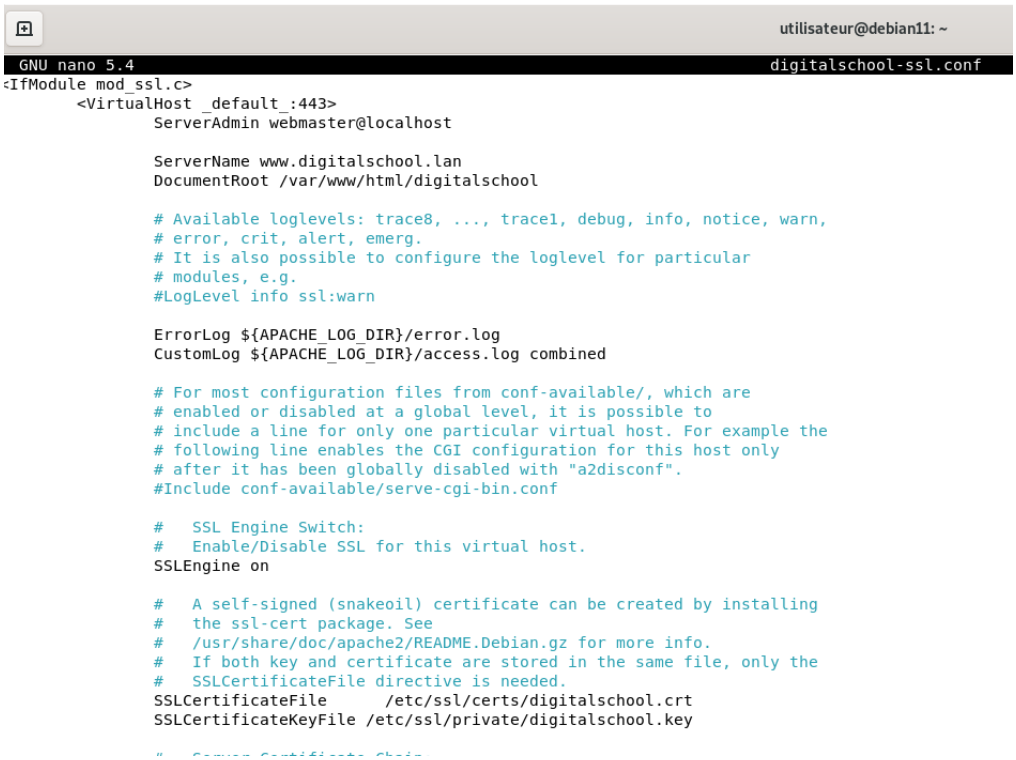
Nous devons ensuite modifier les lignes :

- SSLCertificateFile

- SSLCertificateKeyFile

Pour les sites Mbway et Digitaschool dans mbway-ssl.conf et digitalschool-ssl.conf





On va activer nos sites avec la commande a2ensite + le nom de notre site

Puis redemarrer apache2 : sudo systemctl restart apache2

Maintenant si on ajoute www.mbway.lan en alias à notre serveur dans le fichier /etc/hosts de notre client, vous pouvez voir la page d'accueil de notre site dans notre navigateur :

Nous arrivons donc bien a nous connecter en https !

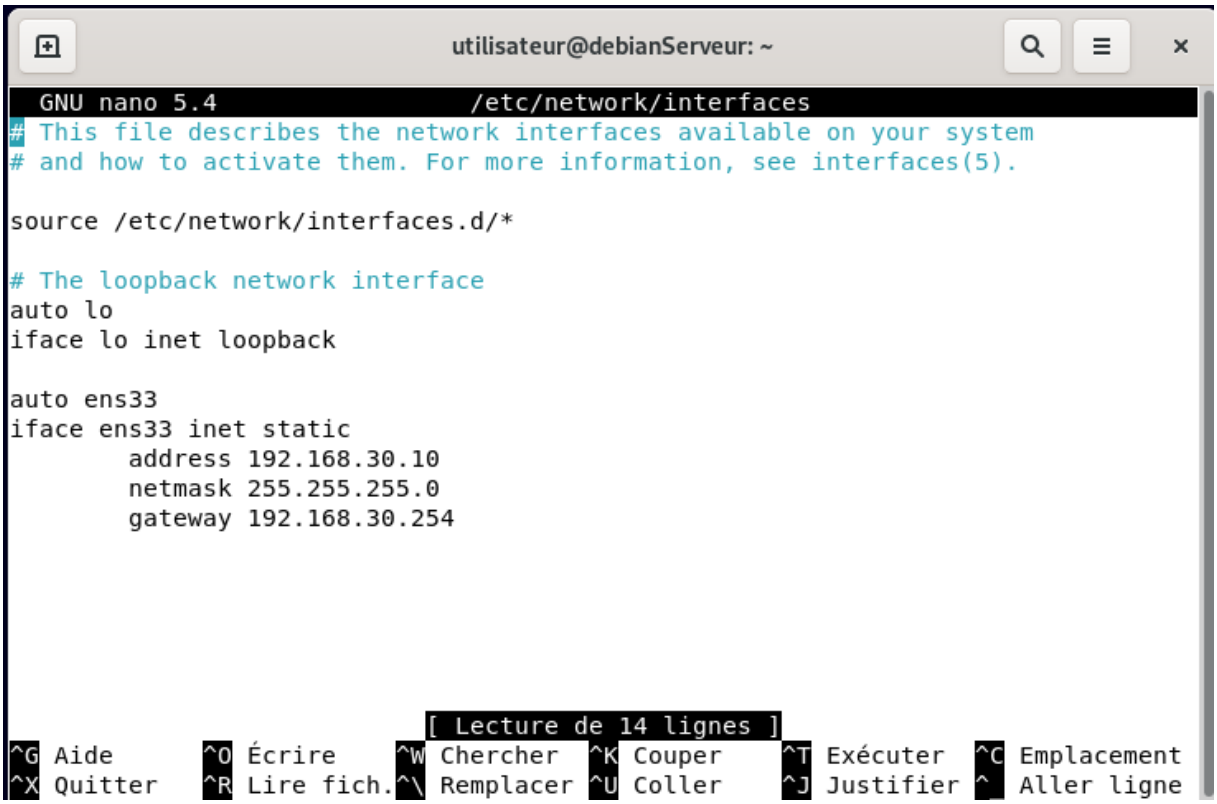
Configuration pour la DMZ :

On va d'abord aller dans le fichier /etc/network/interfaces afin de paramétrer notre carte réseaux avec la commande sudo nano /etc/network/interfaces.

On redémarrera les interfaces avec la commande sudo systemctl restart networking.service pour que les changements prennent effet.

Il ne faut pas oublier d'activer l'accès par pont.

Pour notre serveur on lui définit comme adresse IP : 192.168.30.10



Maintenant que l'interface réseau est prête on pourra finir de configurer notre DMZ via PFsense.

Configuration du DNS :

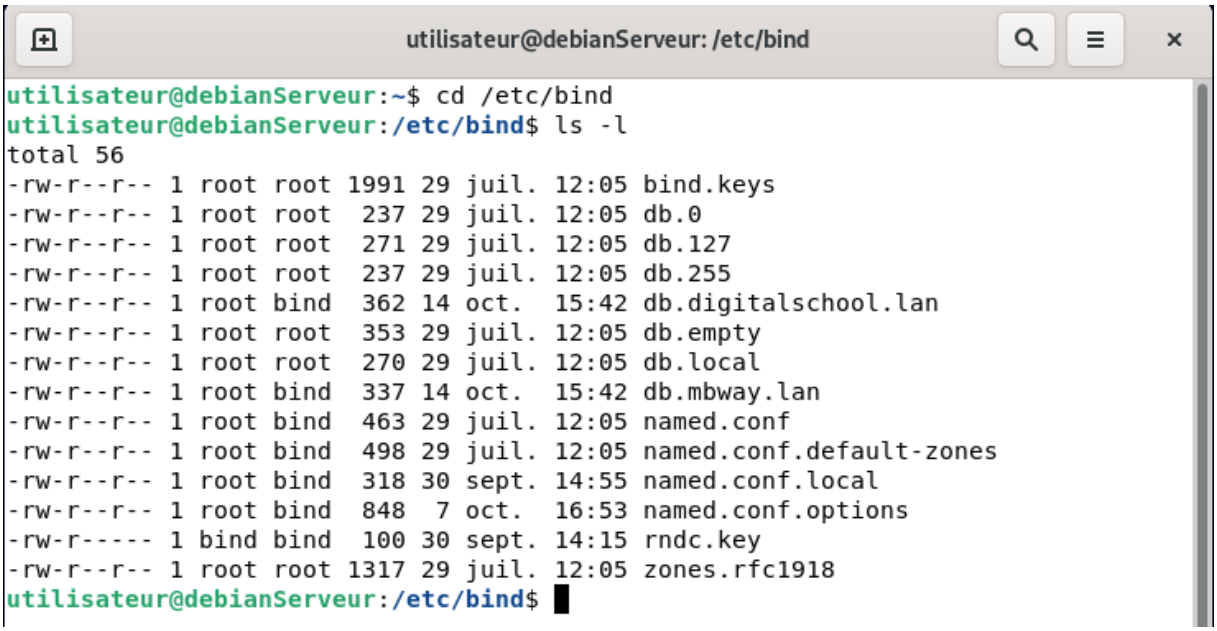
On commence par faire apt-get upgrade

Installation de BIND9 :

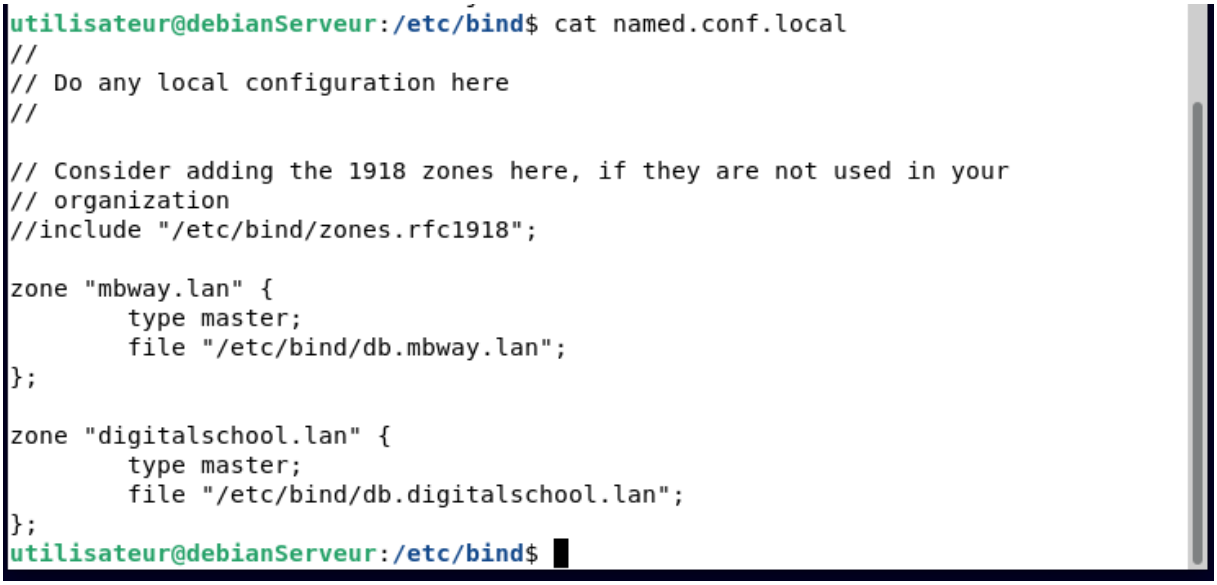
On fait ensuite sudo apt install Bind9 puis sudo apt install bind9utils.

Déclaration et création des zones :

On va commencer pas déclarer les zones dans /etc/bind



On va aller dans le fichiers named.conf.local et le configurer :



On le configure en mode master et ont créé deux zones pour chacun de nos sites et on respecte la convention db.nomDomaine pour le déclarer.

On va ensuite créer nos fichiers de zones :

```
  GNU nano 5.4                    db.mbway.lan
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     debianServer.mbway.lan. root.debianServer.mbway.lan. (
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      debianServer.mbway.lan.
debianServer    IN      A       192.168.30.10
www     IN      A       192.168.30.10

                        [ Lecture de 14 lignes ]
^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^\ Remplacer ^U Coller    ^J Justifier ^_ Aller ligne
```

```
  GNU nano 5.4                  db.digitalschool.lan
;
; BIND data file for local loopback interface
;
$TTL    604800
@       IN      SOA     debianServer.digitalschool.lan. root.debianServer.digit>
                              2         ; Serial
                         604800         ; Refresh
                          86400         ; Retry
                        2419200         ; Expire
                         604800 )       ; Negative Cache TTL
;
@       IN      NS      debianServer.digitalschool.lan.
debianServer    IN      A       192.168.30.10
www     IN      A       192.168.30.10

                        [ Lecture de 14 lignes ]
^G Aide      ^O Écrire    ^W Chercher  ^K Couper    ^T Exécuter  ^C Emplacement
^X Quitter   ^R Lire fich.^\ Remplacer ^U Coller    ^J Justifier ^_ Aller ligne
```

On lui donne les bons noms et adresse IP : debianRouteur.mbway.lan. et debianRouteur.digitalschool.lan.

On ajoute WWW pour avoir le www.mbway.lan et on voit également que l'on a un enregistrement de type A.

On va également configurer le fichier named.conf.options avec le DNS de Google.

Les forwarders sont d'autres serveurs DNS vers lesquels le serveur DNS local peut envoyer les requêtes qu'il ne peut pas résoudre.

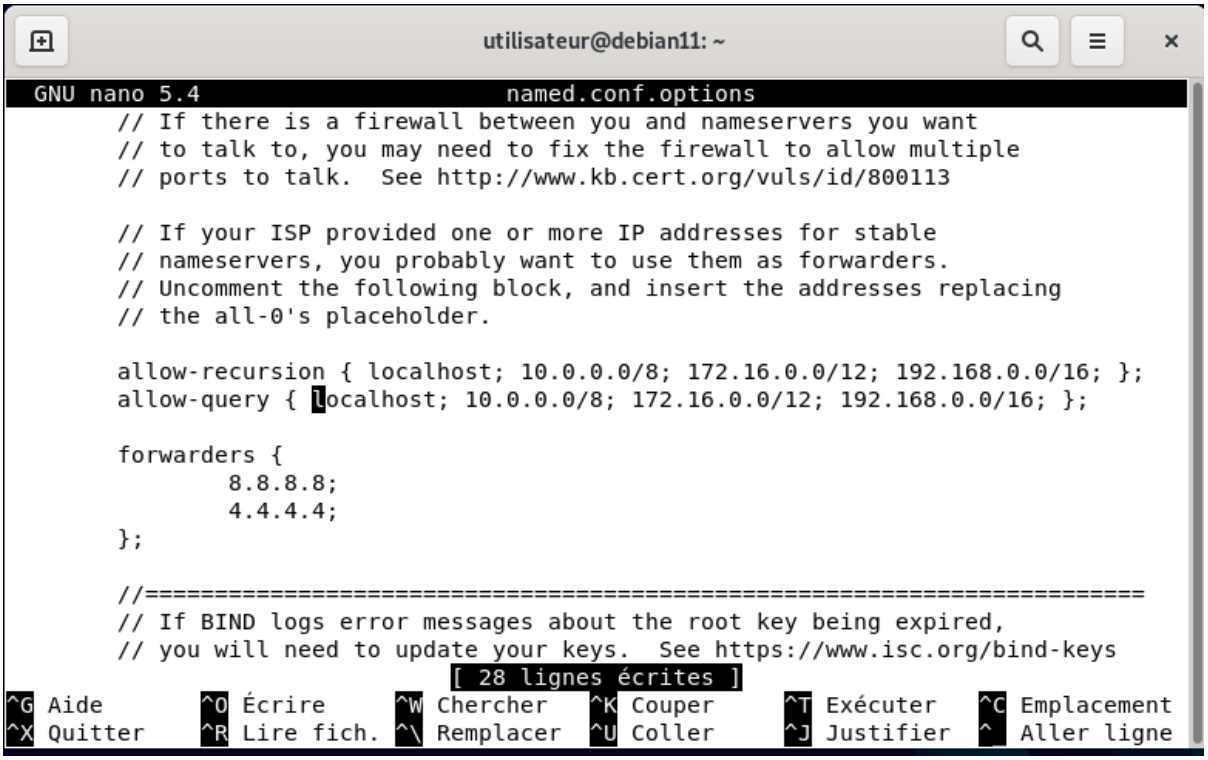
On doit aussi configurer Bind9 pour accepter toutes les requêtes provenant d'adresses privées définies par les standards RFC1918 (qui inclut les réseaux LAN).

Voici les plages d'adresses locales selon RFC1918 :

- 10.0.0.0/8 : Pour les réseaux de classe A.
- 172.16.0.0/12 : Pour les réseaux de classe B.
- 192.168.0.0/16 : Pour les réseaux de classe C.

Dans notre fichier named.conf.options, remplacez la directive allow-recursion par une liste incluant ces plages avec :

- allow-query : qui contrôle qui peut poser n'importe quelle question DNS (locale ou externe).
- allow-recursion : qui contrôle qui peut poser des questions récursives (requêtes nécessitant que le serveur DNS interroge d'autres serveurs).



On redémarre Bind9 afin que les modifications prennent effet : sudo systemctl restart bind9

Dans nos machines Debian on va devoir leur indiquer le nouveau DNS dans le fichier

Nano /etc/resolv.conf :

Nameserver : 192.168.30.10

On peut vérifier notre configuration grâce à la commande nslookup :

```
utilisateur@debianClient2:~$ nslookup www.mbway.lan
Server:         192.168.30.10
Address:        192.168.30.10#53

Name:   www.mbway.lan
Address: 192.168.30.10
```

On peut donc maintenant taper https://www.mbway.lan/

On teste la connexion :

Configuration du serveur MariaDB :

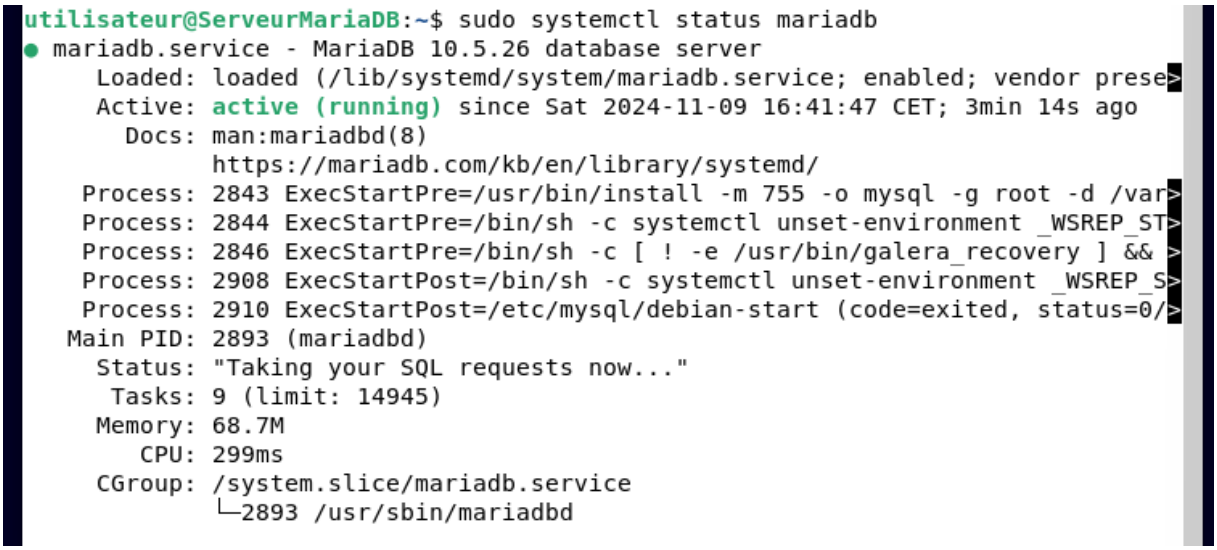
On va devoir installer et configurer MariaDB puis mettre en place notre base de données.

Dans un premier temps faire sudo apt-get update afin d'avoir les dernières versions disponibles.

Installation de MariaDB :

Utiliser la commande sudo apt-get install mariadb-server mariadb-client

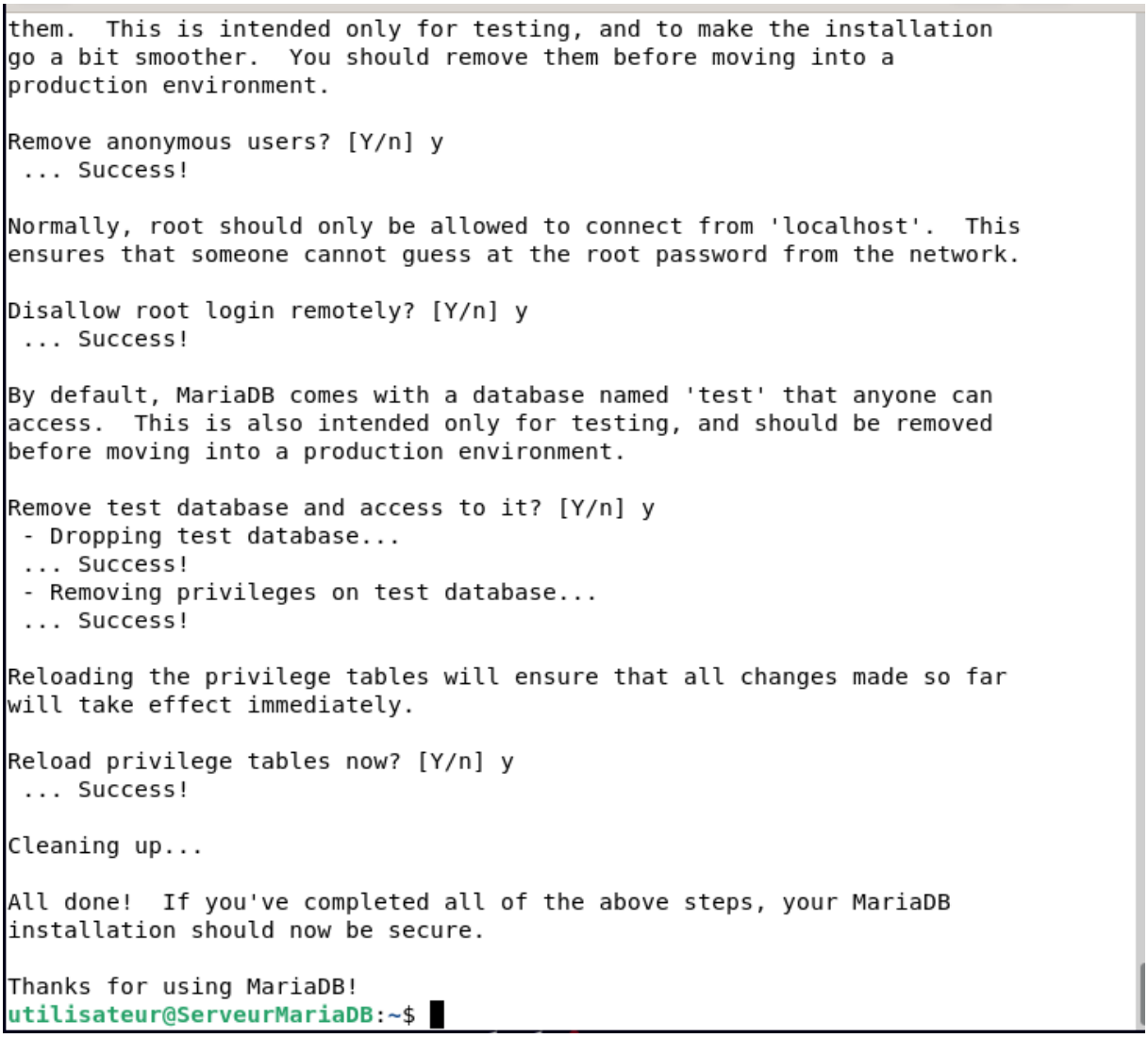
Puis vérifier qu'il se soit bien installé : sudo systemctl status mariadb



Sécurisation de l'installation MariaDB :

Nous allons utiliser un script afin de sécuriser un minimum notre installation via la commande : sudo mariadb-secure-installation :

```
utilisateur@ServeurMariaDB:~$ sudo mariadb-secure-installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
 ... skipping.

You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] y
New password:
Re-enter new password:
Sorry, passwords do not match.

New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
```



Nous avons répondu N à la question Switch to unix_socket authentication. Cela authorisera les connexions sur la base de données MariaDB avec un nom d'utilisateur et un mot de passe.

Puis répondre Y à la question suivante Change the root password ? pour spécifier le mot de passe de l'utilisateur root de MariaDB.

Cet utilisateur root de la base de données aura tous les droits d'accès.

A la question Remove anonymous users ? Répondre Y pour désactiver les connexions anonymes

Pour désactiver les connexions root depuis un serveur autre que le nôtre répondre Y a la question : Disallow root login remotely ?

Répondre Y a : Remove test database and access to it pour supprimer la base de données de test et l'accès.

Et enfin répondre Y a Reload privilege tables now pour recharger les tables de privilèges.

```
utilisateur@ServeurMariaDB:~$ sudo mariadb -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 40
Server version: 10.5.26-MariaDB-0+deb11u2 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

Maintenant que notre Serveur est installé nous allons pouvoir créer nos bases de données

Création des bases de données :

Connexion a MariaDB :

- Sudo mariadb -u root -p

```
utilisateur@ServeurMariaDB:~$ sudo mariadb -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 40
Server version: 10.5.26-MariaDB-0+deb11u2 Debian 11

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Puis création des bases de données pour Mbway et Digitalschool :

- Create database mbway ;
- Create database digitalschool ;

```
MariaDB [(none)]> create database mbway;
Query OK, 1 row affected (0,001 sec)

MariaDB [(none)]> create database digitalschool;
Query OK, 1 row affected (0,000 sec)

MariaDB [(none)]>
```

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| digitalschool      |
| information_schema |
| mbway              |
| mysql              |
| performance_schema |
+--------------------+
5 rows in set (0,000 sec)

MariaDB [(none)]>
```

Nous allons maintenant configurer notre serveur MariaDB pour autoriser les requêtes SQL provenant du serveur web.

Configuration des connexions distantes :

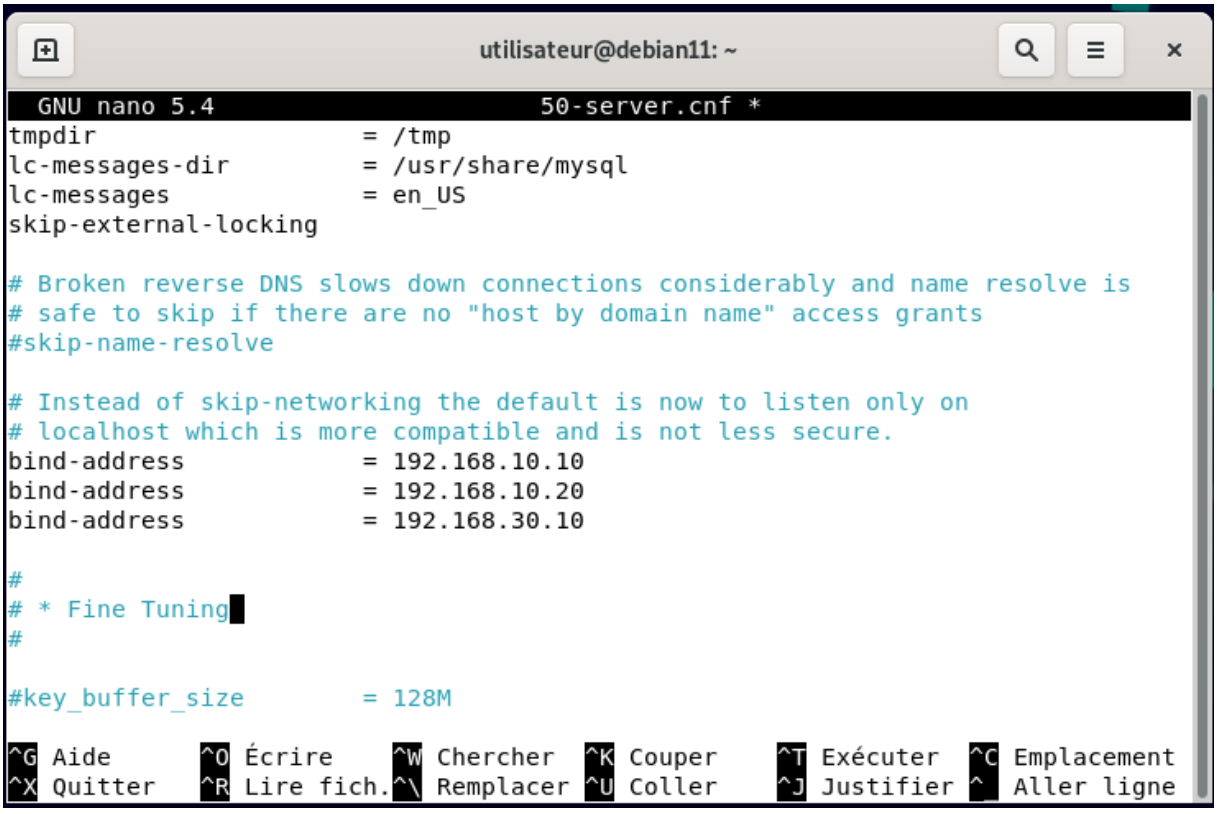
Tout d'abord, nous devons configurer le serveur de façon qu'il écoute sur une autre adresse IP que "127.0.0.1". On doit modifier le fichier :

- sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf

Puis modifier la ligne :

- bind-address = 192.168.10.20 pour la base de données
- bind-address = 192.168.10.10 pour le poste administrateur
- bind-address = 192.168.30.10 pour le serveur Web

On doit lui indiquer l'adresse IP du poste depuis lequel on pourra interroger MariaDB :



Nous allons devoir maintenant configurer un utilisateur et lui accorder les droits nécessaires à la gestion des bases de données du campus.

Nous allons utiliser la commande :

- GRANT ALL PRIVILEGES ON mbway.* TO 'administrateur'@'192.168.10.10' IDENTIFIED BY 'root' WITH GRANT OPTION;
- su

- GRANT ALL PRIVILEGES ON mbway.* : Donne tous les privilèges (lecture, écriture, modification, suppression, etc.) sur toutes les tables de la base mbway.

- TO 'administrateur'@'192.168.10.10' : Spécifie l'utilisateur administrateur qui se connecte depuis l'adresse IP 192.168.10.10.

- IDENTIFIED BY 'root' : Définit le mot de passe de l'utilisateur administrateur comme root. Lors de la mise en production, nous utiliserons un mot de passe robuste.

- WITH GRANT OPTION : Permet à l'utilisateur administrateur de déléguer (accorder) ces privilèges à d'autres utilisateurs.

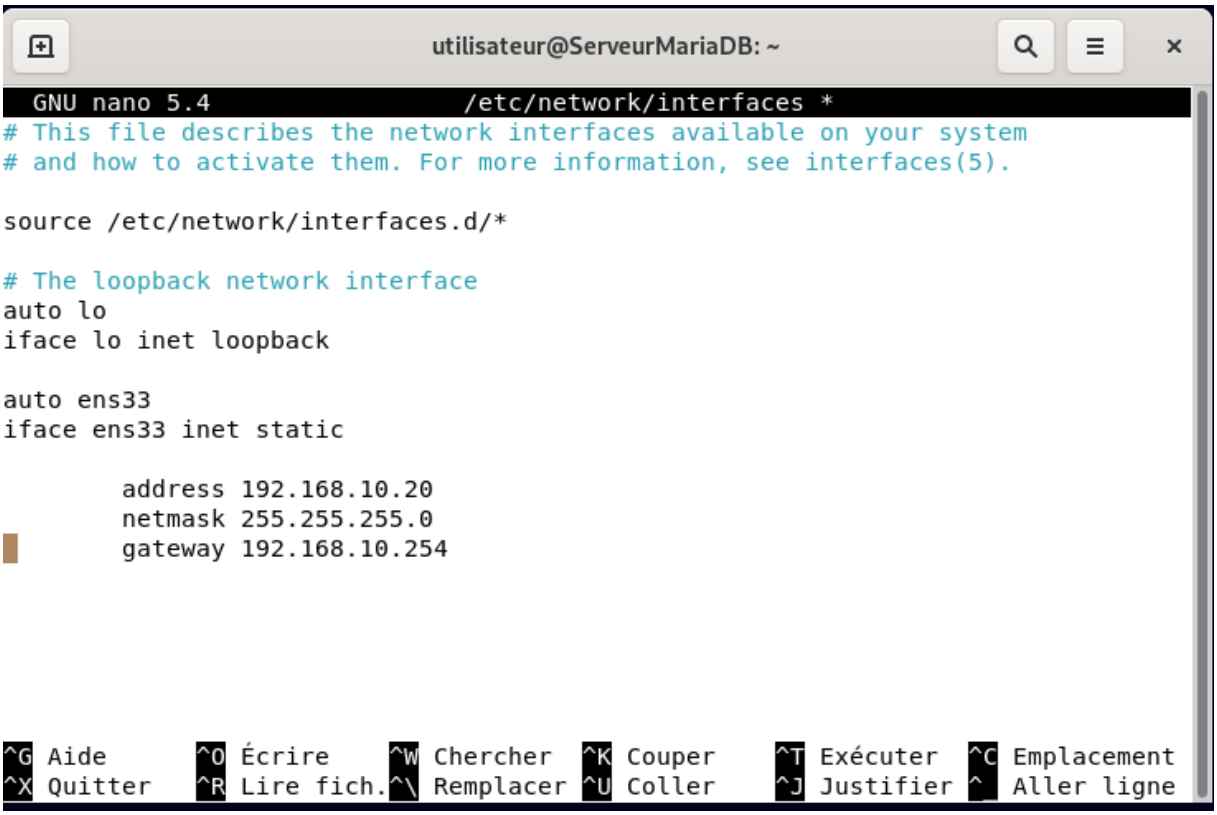
Puis, on met à jour les privilèges avant de quitter :

- FLUSH PRIVILEGES;
- EXIT;

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| digitalschool      |
| information_schema |
| mbway              |
| mysql              |
| performance_schema |
+--------------------+
5 rows in set (0,003 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON mbway.* TO 'administrateur'@'192.168.1
0.10' IDENTIFIED BY 'root' WITH GRANT OPTION;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON digitalschool.* TO 'administrateur'@'1
92.168.10.10' IDENTIFIED BY 'root' WITH GRANT OPTION;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]>
```

Configuration pour le LAN :

On va d'abord aller dans le fichier /etc/network/interfaces afin de paramétrer notre carte réseaux avec la commande sudo nano /etc/network/interfaces.



On redémarrera les interfaces avec la commande sudo systemctl restart networking.service pour que les changements prennent effet.

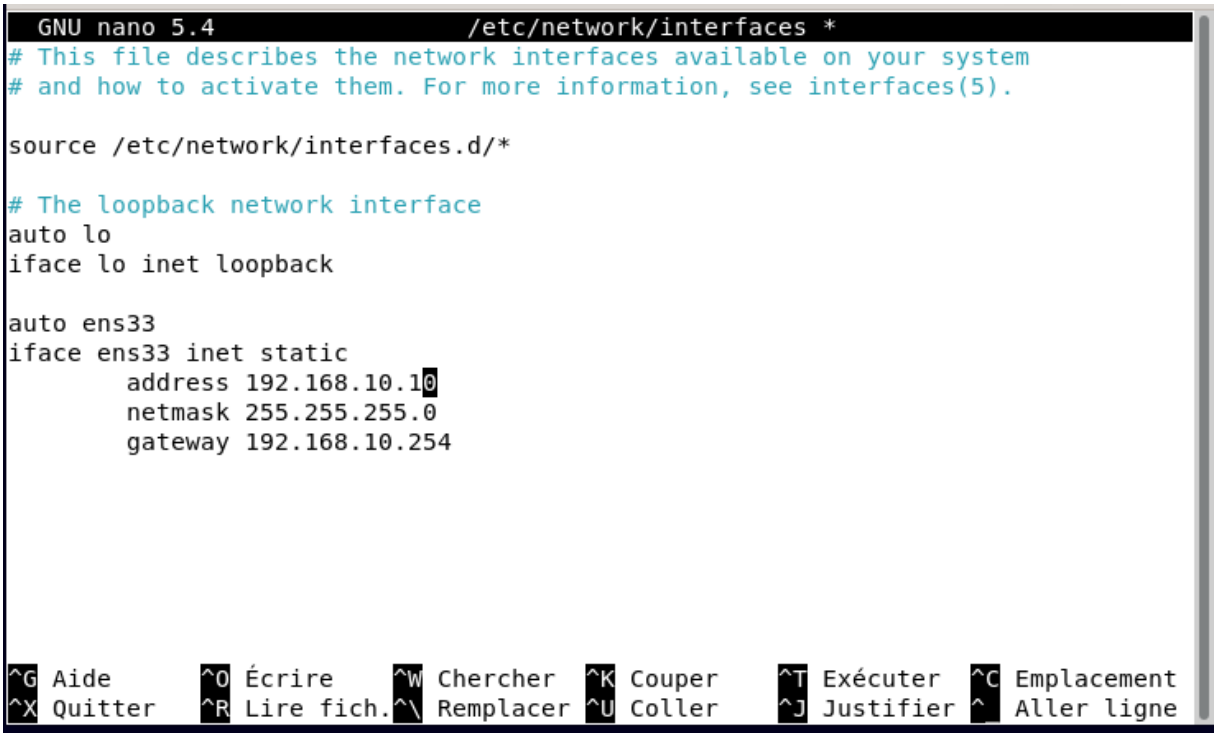
Il ne faut pas oublier d'activer l'accès par pont.

Pour notre serveur on lui définit comme adresse IP : 192.168.10.20

Configuration réseau des postes pour le LAN et le WAN :

Configuration pour le LAN :

Pour notre poste administrateur on lui définit comme adresse IP : 192.168.10.10



Configuration pour le WAN :

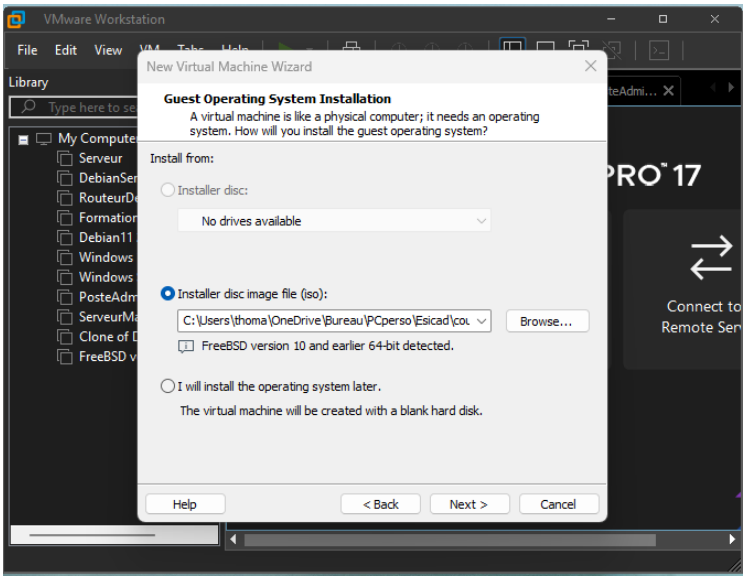
Pour notre poste situer dans le WAN on sera connecté en DHCP.

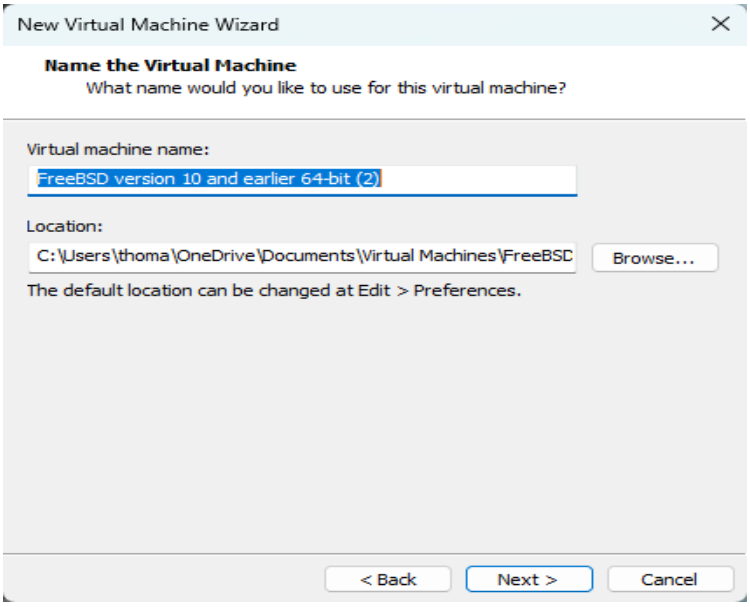
```
  GNU nano 5.4                  /etc/network/interfaces *
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

auto ens33
iface ens33 inet dhcp

^G Aide       ^O Écrire      ^W Chercher  ^K Couper     ^T Exécuter  ^C Emplacement
^X Quitter    ^R Lire fich.  ^\ Remplacer ^U Coller     ^J Justifier ^_ Aller ligne
```

Configuration du pare-feu PFSENSE :

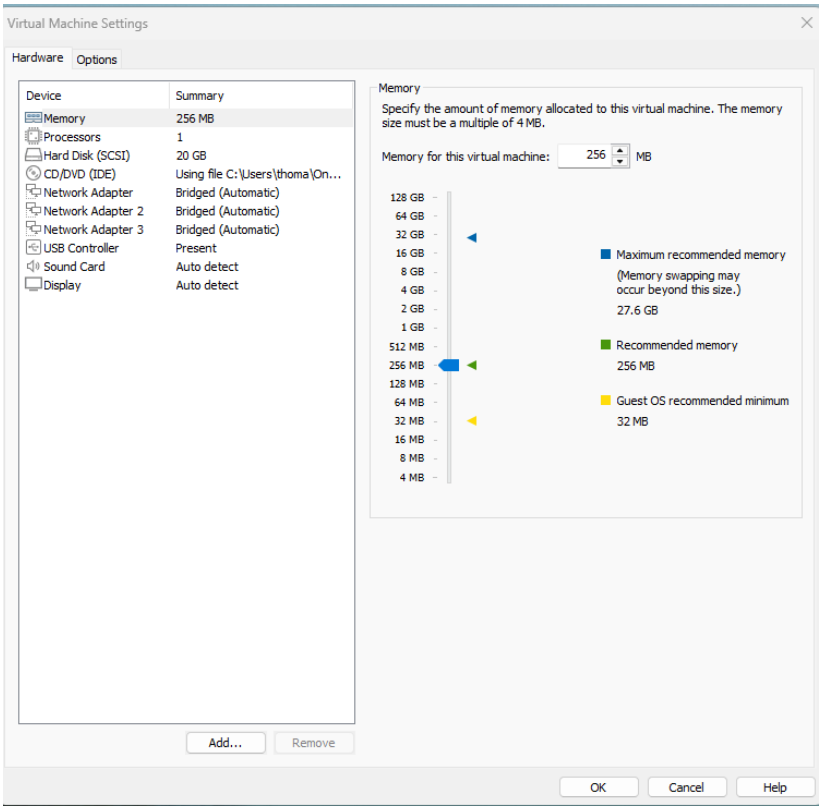
Installation de PFSENSE :

Nous allons dans un premier temps réaliser l'installation d'une machine virtuelle sous Pfense :



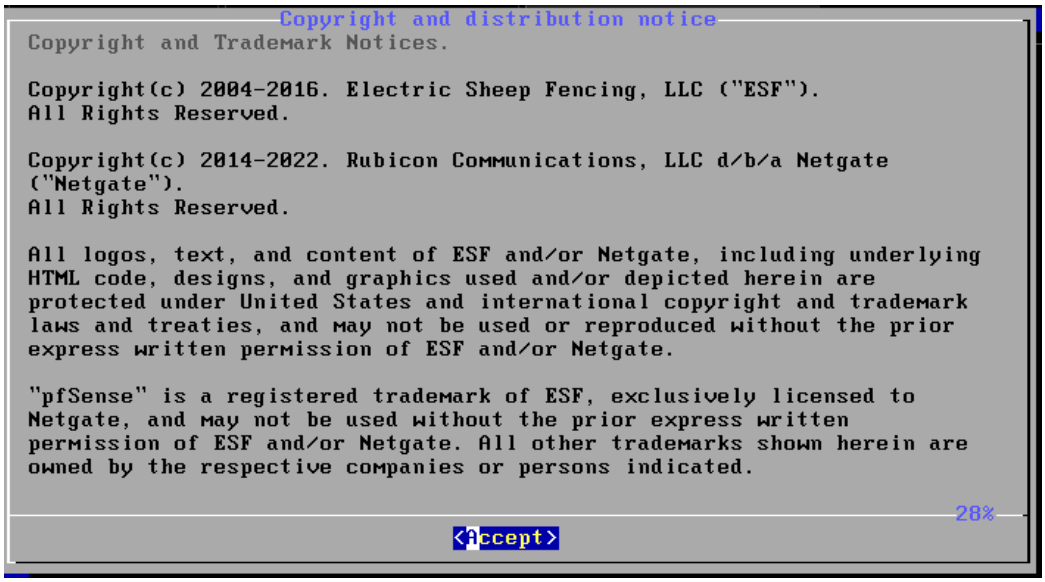


Choix de l’image ISO

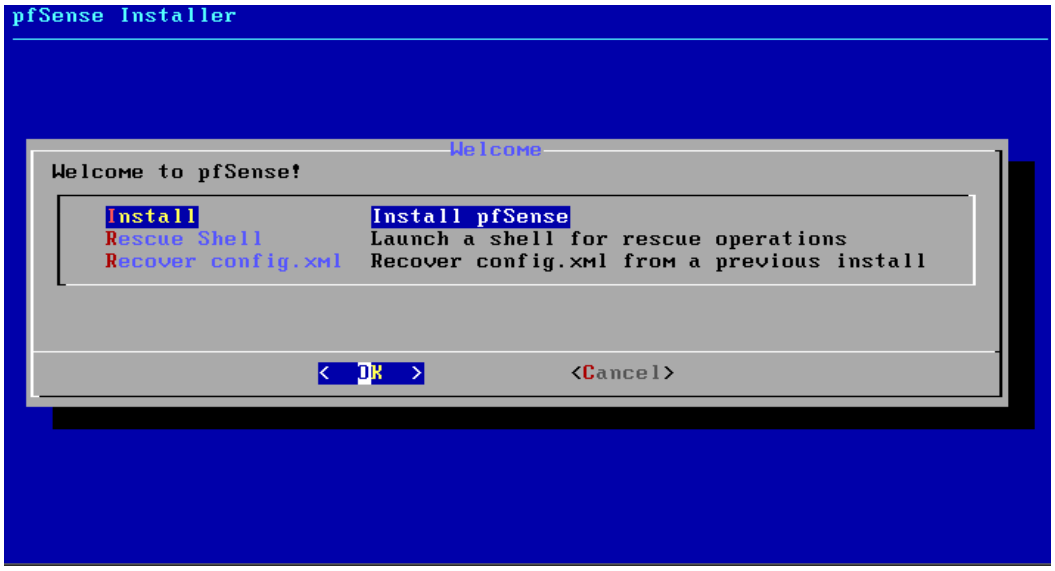


Créations des 3 interfaces réseaux

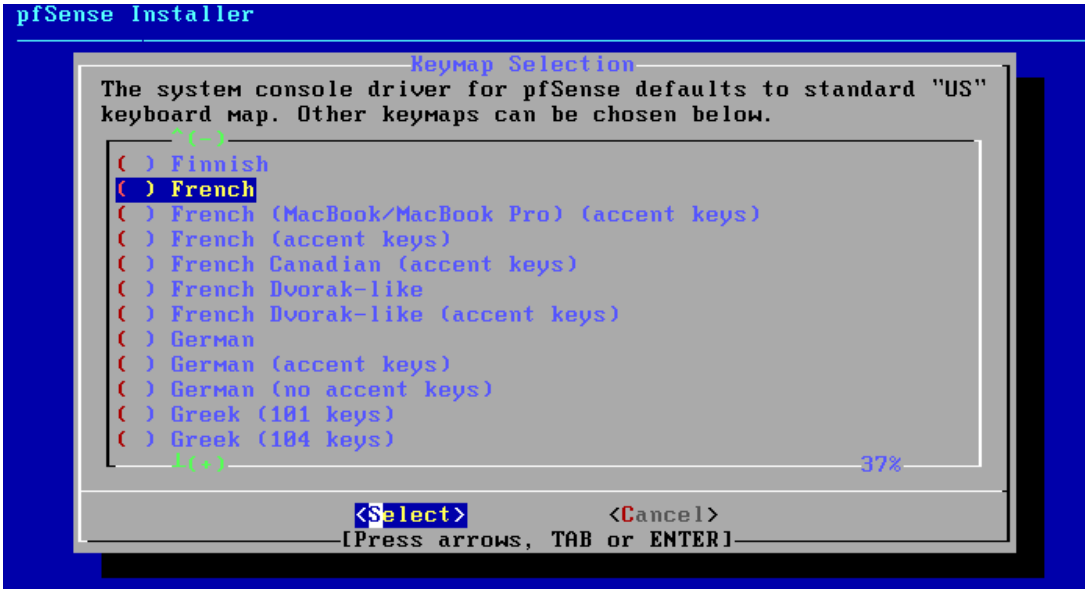
Début de l’installation :

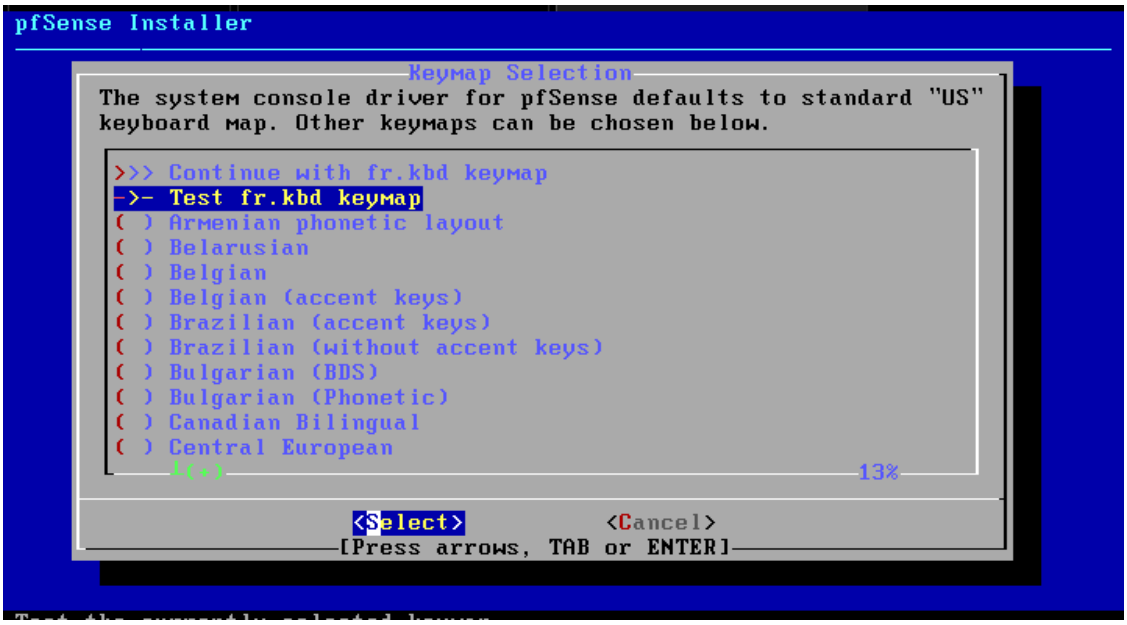


Nous allons cliquer sur Install pour commencer l'installation de Pfense :

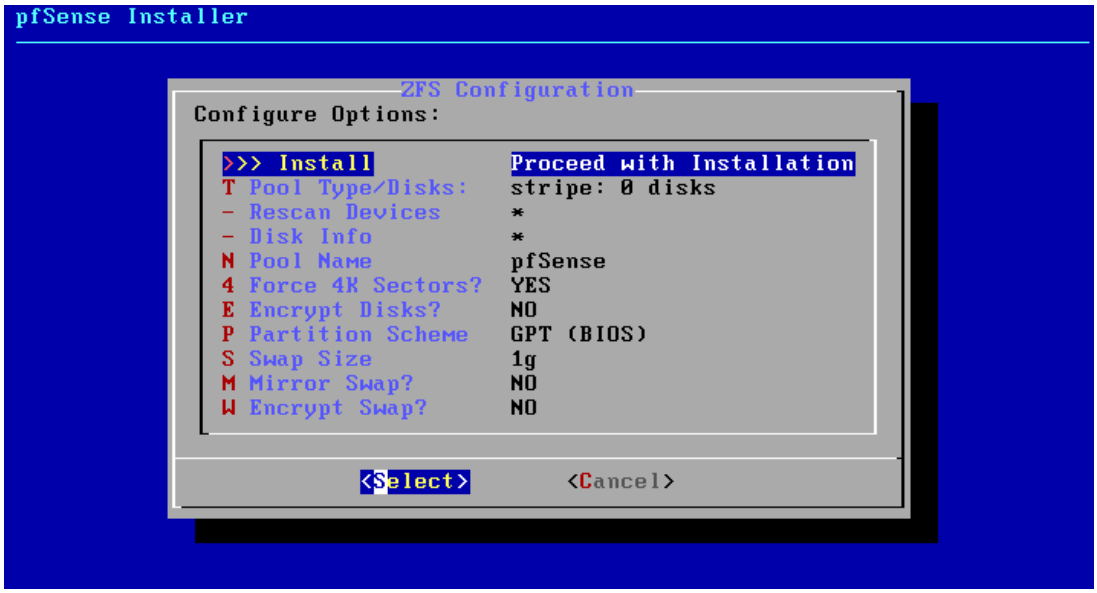


Nous allons sélectionner la langue :

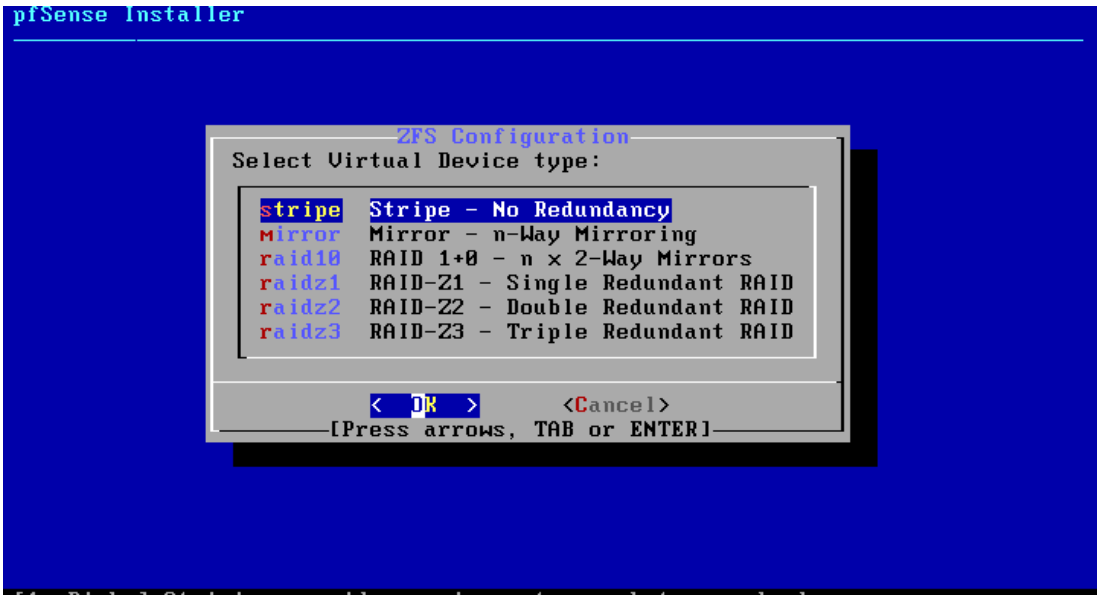




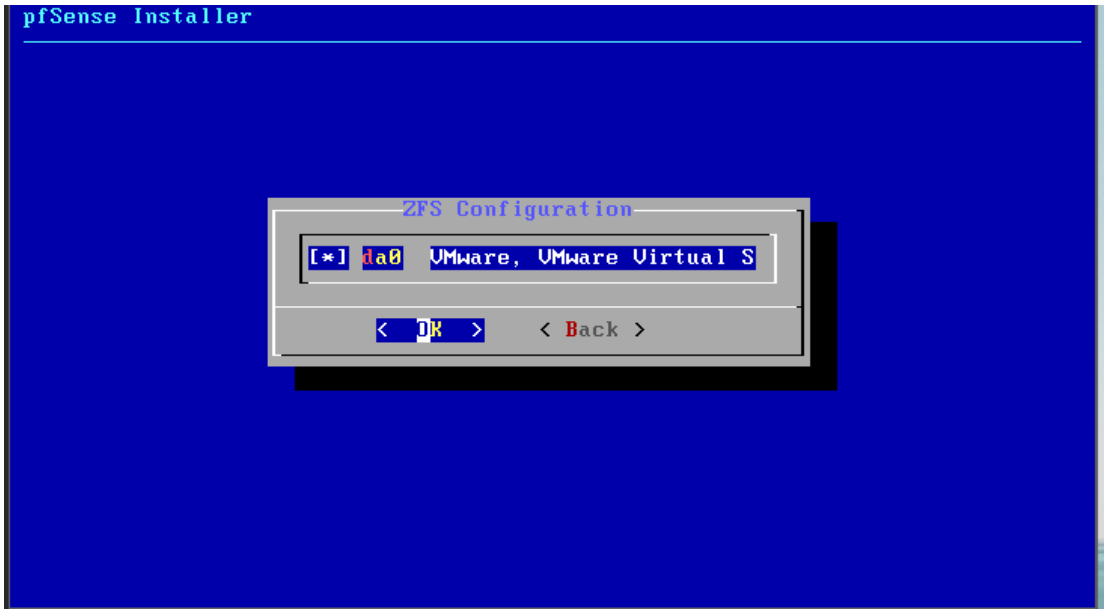
Nous allons sélectionner l'installation Auto (ZFS) :



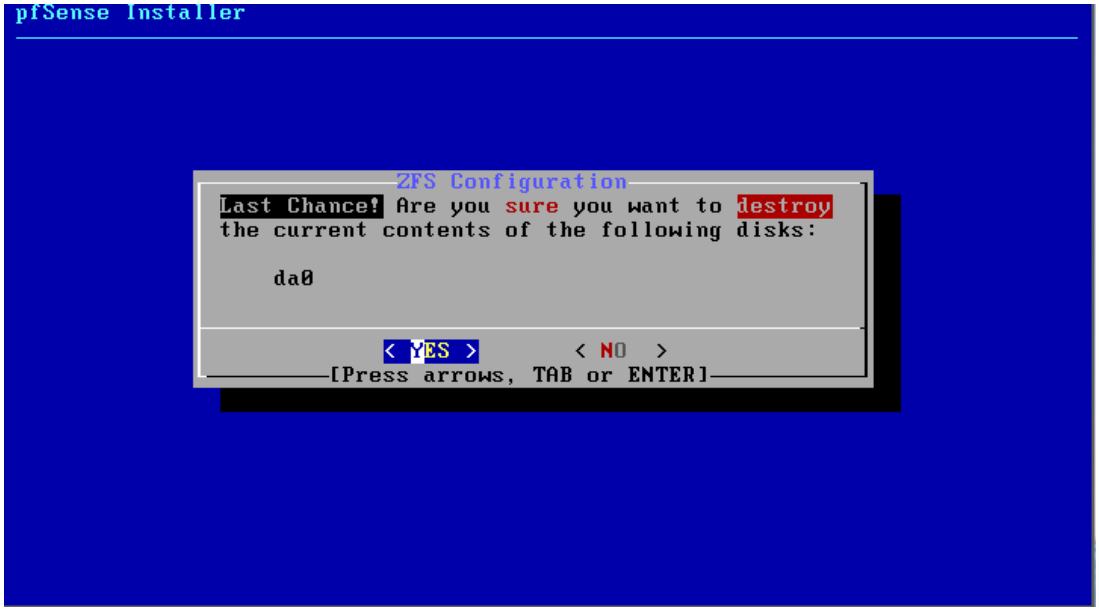
Nous allons sélectionner une configuration sans redondance :

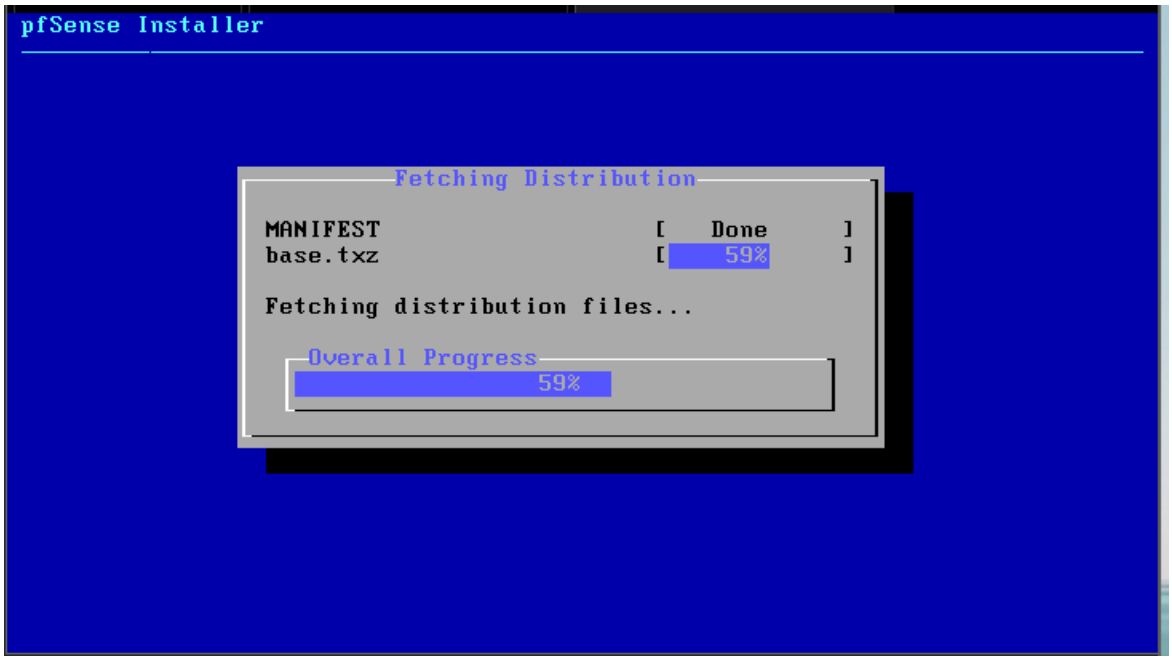


Choix du disque virtuel :



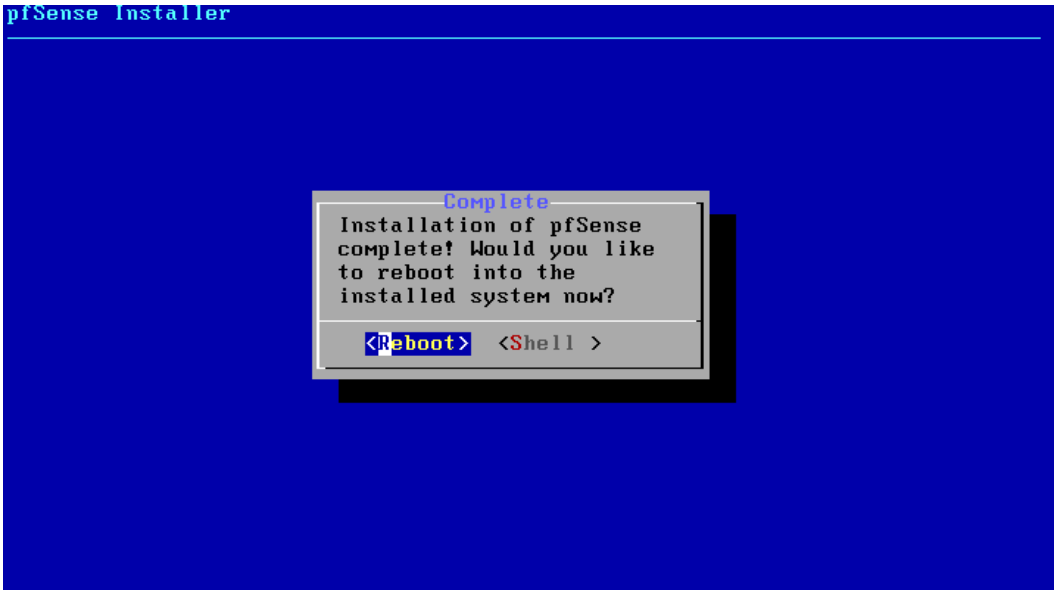
Nous autorisont la suppressions de nos données :





Nous allons finir l'installation sans apporter d'autre modifications :

Nous allons redémarrer notre VM afin de finir l'installation :



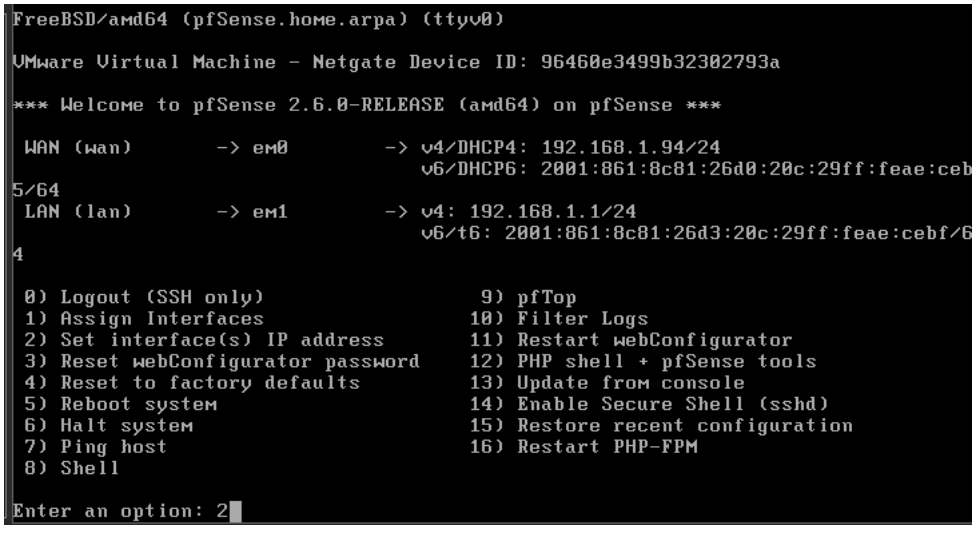
L'installation c'est donc dérouler avec succès.

Configuration des interfaces réseaux :

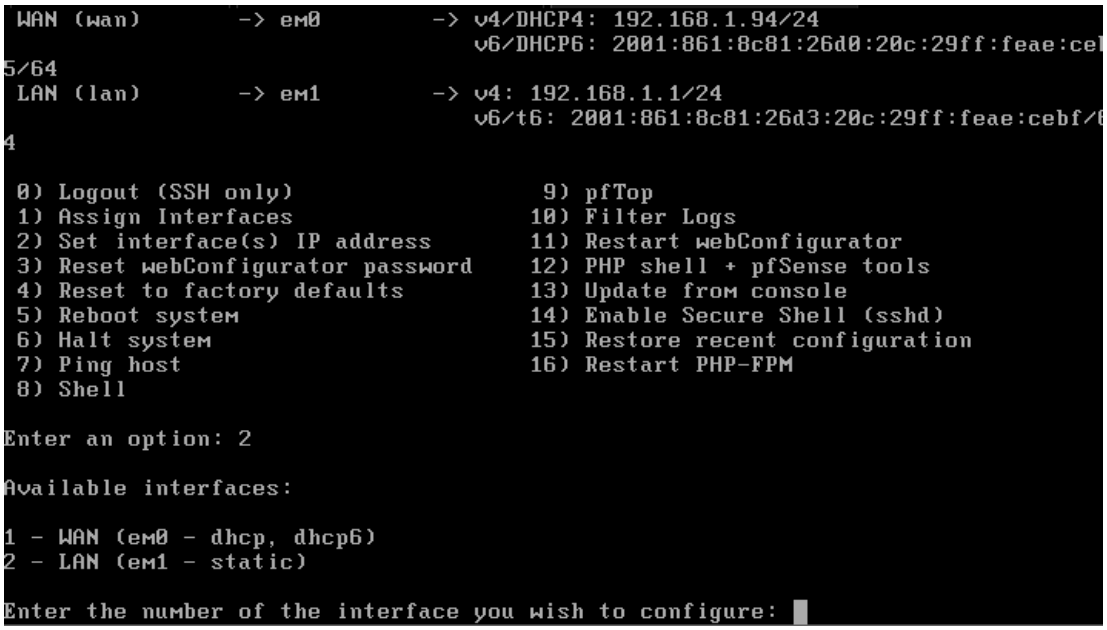
Nous allons maintenant configurer les interfaces réseaux de Pfense :

- Le WAN sera connecté en DHCP
- Le LAN aura une IP statique ici 192.168.10.254
- La DMZ aura une IP statique ici 192.168.30.254

L'interface coté WAN est connecté en DHCP nous n'avons donc pas besoin de la configurer :

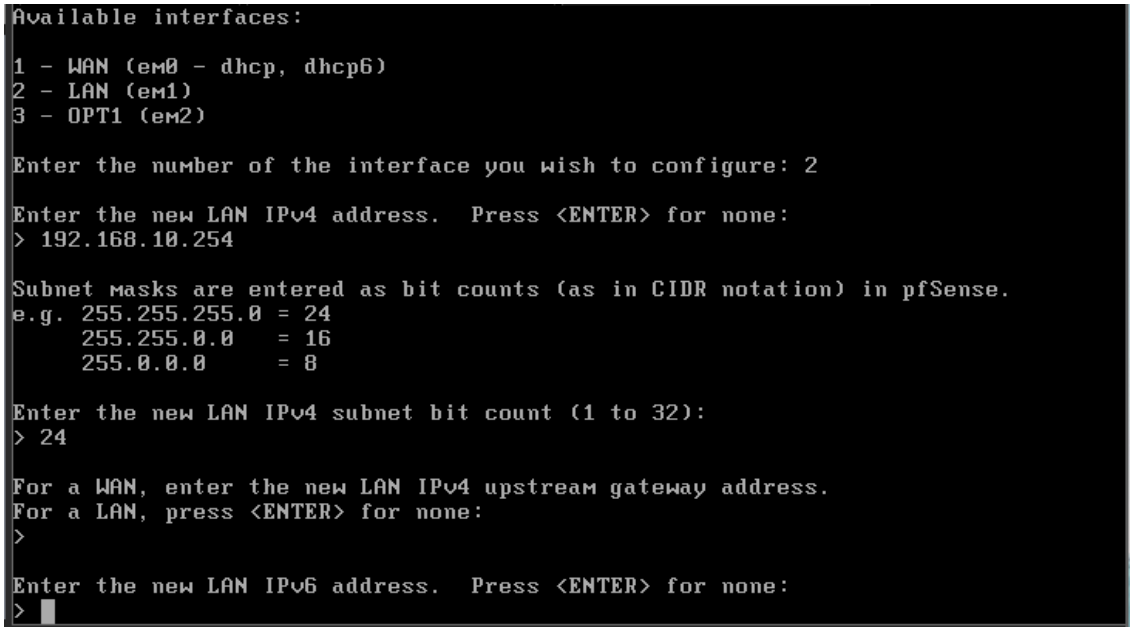


Nous allons sélectionner 2) dans le menu afin de configurer nos interfaces réseau :



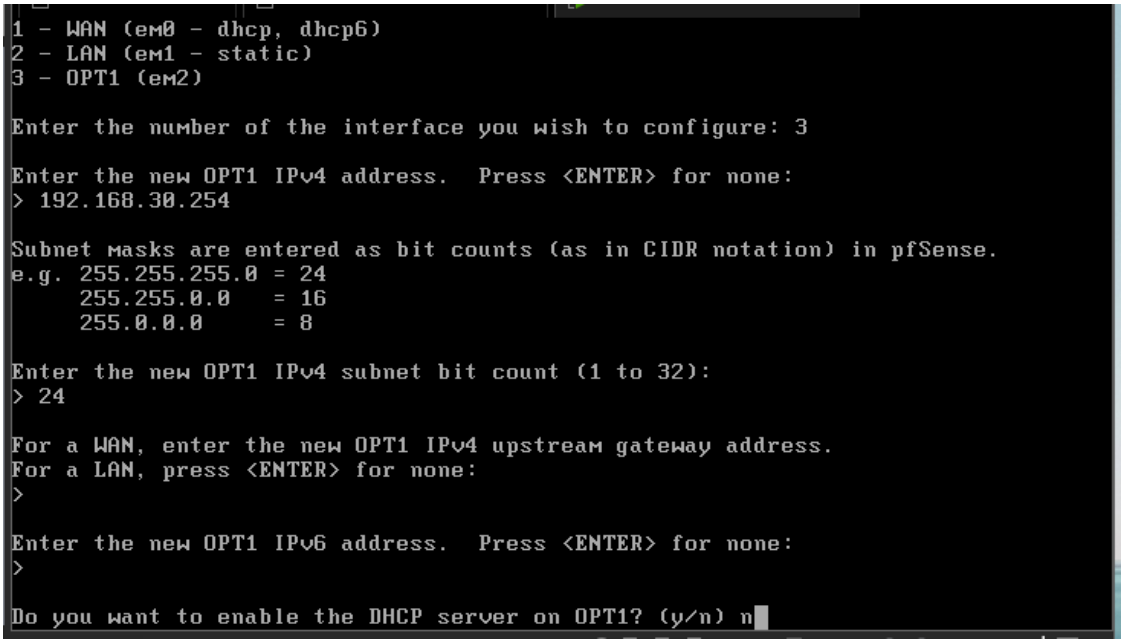
Nous allons sélectionner l'interface 2 – LAN (em1) :

- Choix de l'adresse IP statique : 192.168.10.254 /24 qui nous servira également de passerelle.

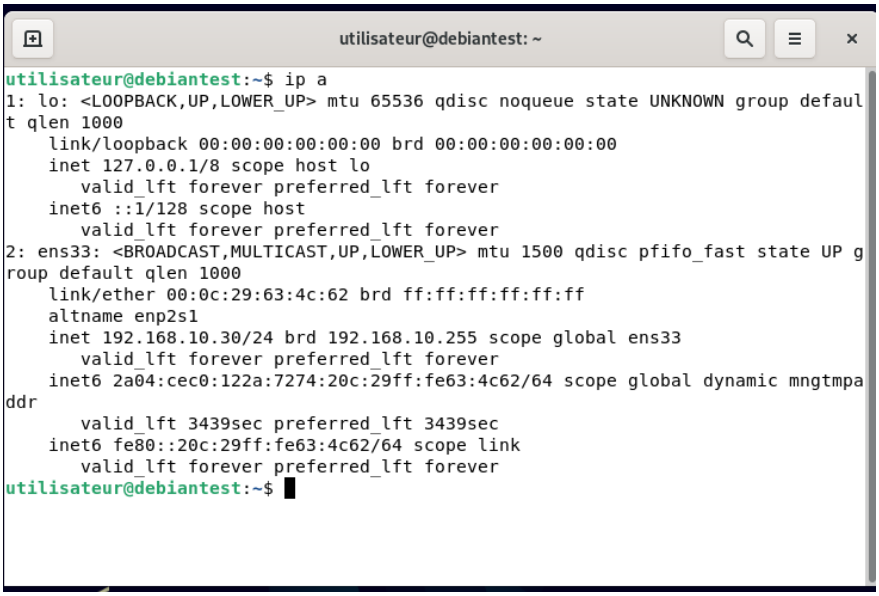


Ensuite nous allons sélectionner l'interface 3 – OPT1(em2) qui sera notre DMZ:

- Choix de l'adresse IP statique : 192.168.30.254 /24 qui nous servira également de passerelle.



Connection depuis l'interface WEB depuis un poste du LAN :



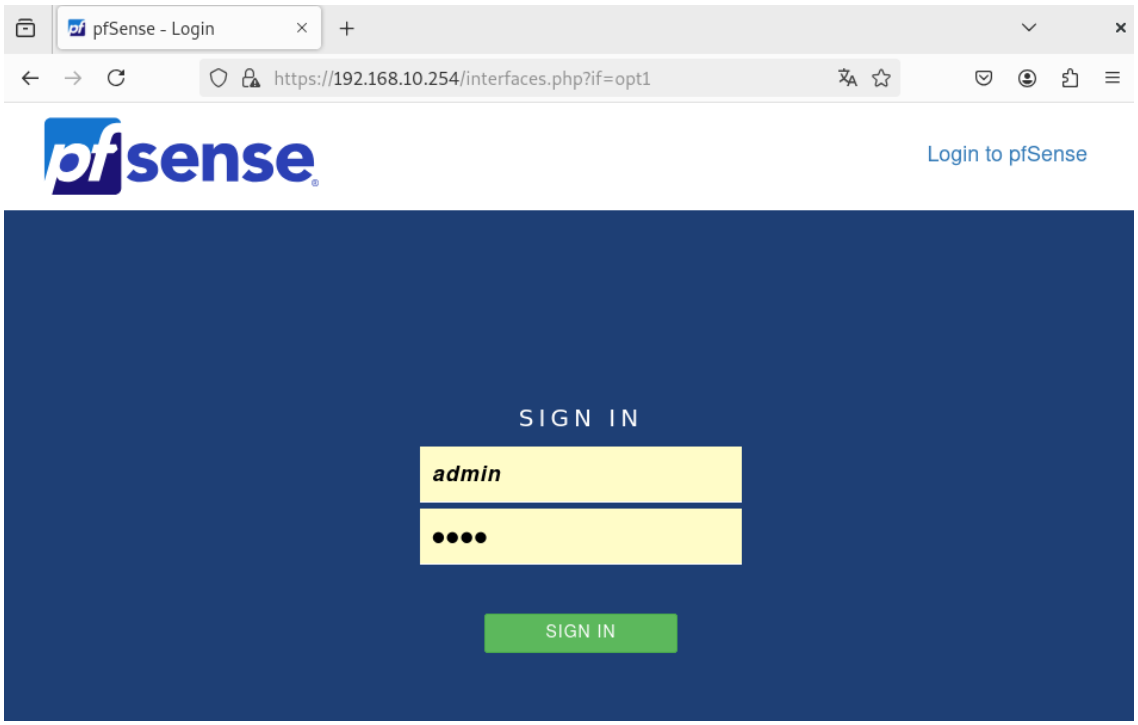
Nous allons nous connecter à l'interface WEB depuis un poste qui se trouve dans le LAN ici :

- 192.168.10.30

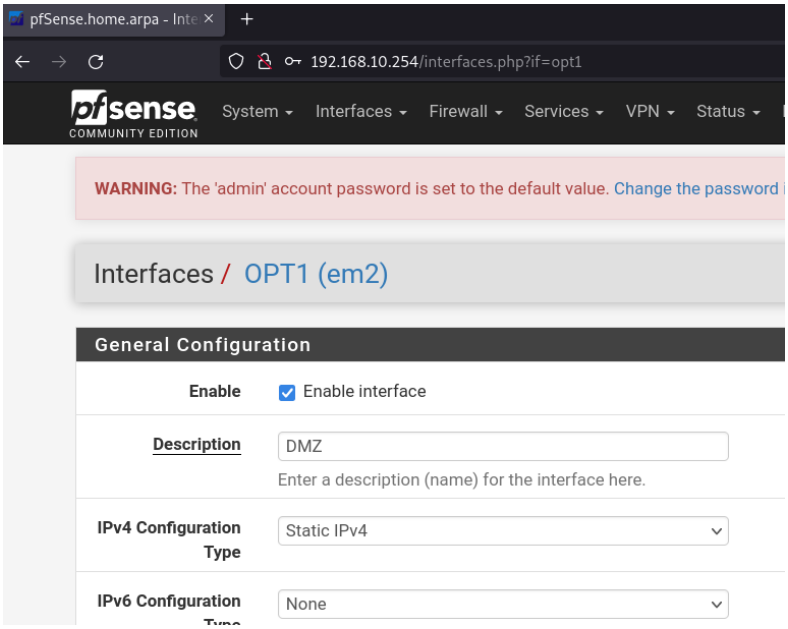
Et qui aura comme passerelle :

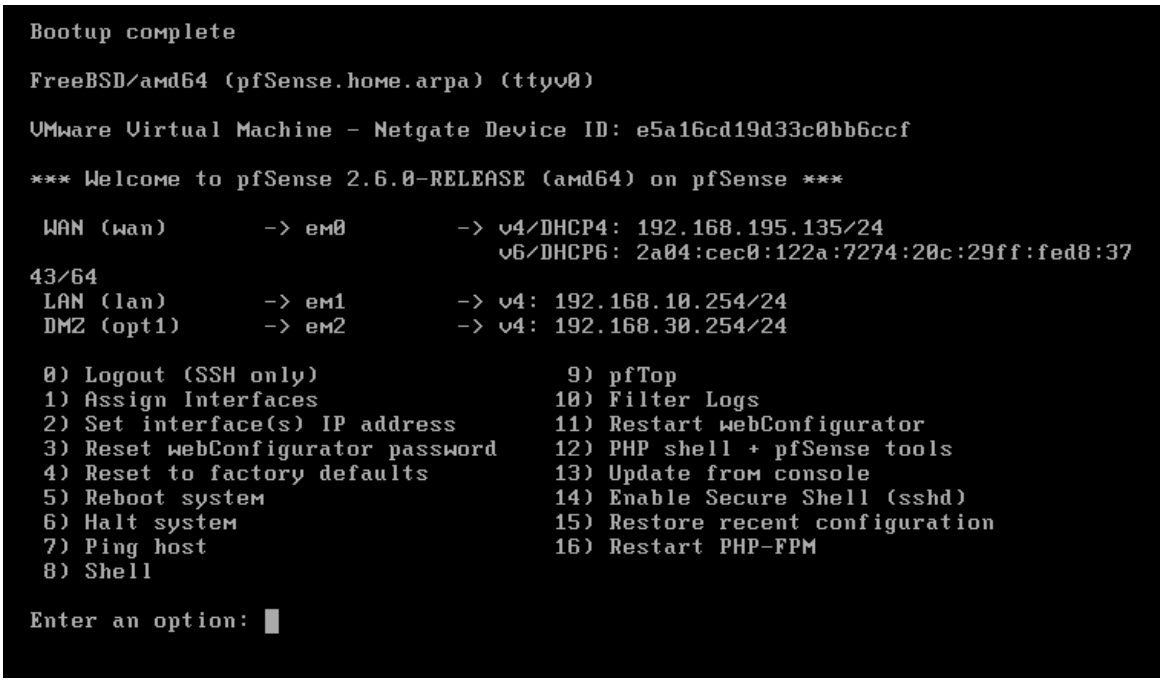
- 192.168.10.254

Connexion avec comme identifiant admin et mot de passe Pfense :

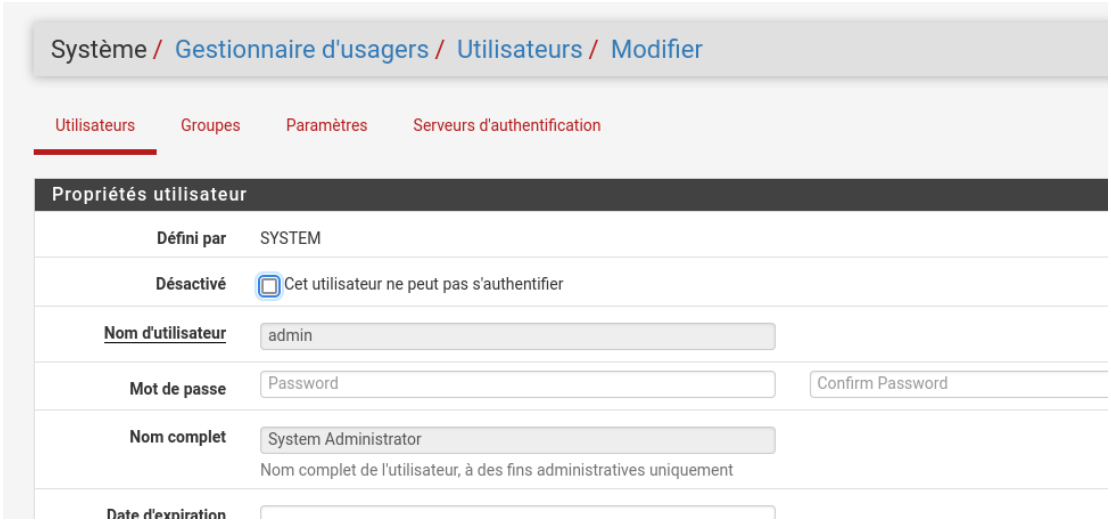


Nous allons commencer par renommer OPT1 (em2) par DMZ :

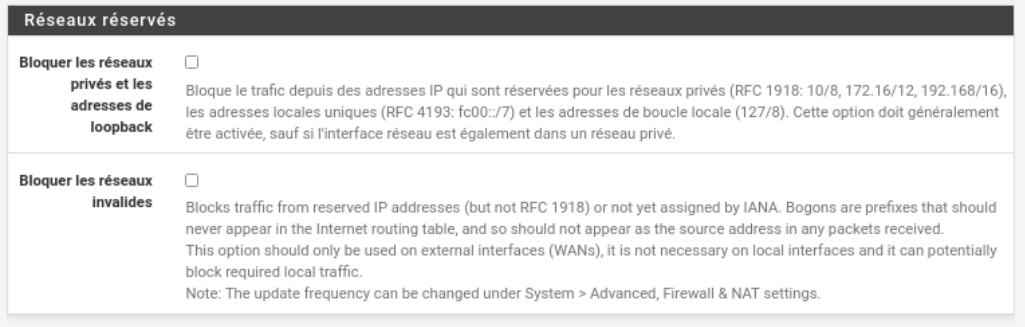




Puis nous allons changer le mot de passe administrateur de base :



Nous allons ensuite nous rendre sur l'interface WAN afin de désactiver ces deux options car nous allons travailler dans un lab :

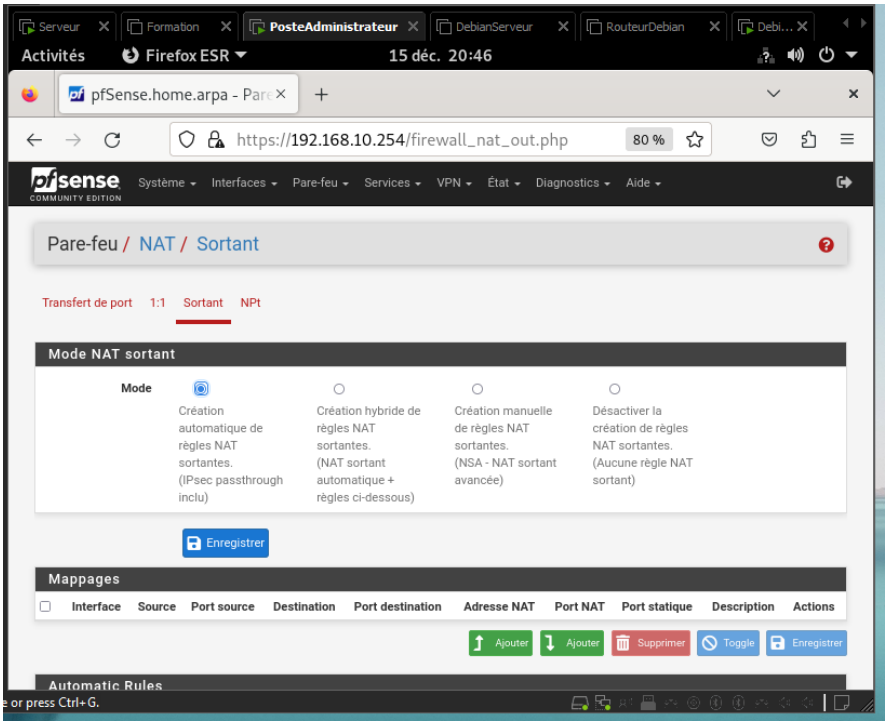


Nous allons maintenant pouvoir mettre en place nos règles de filtrages afin de contrôler le trafic.

Mise en place des règles de filtrages :

Mise en place du NAT :

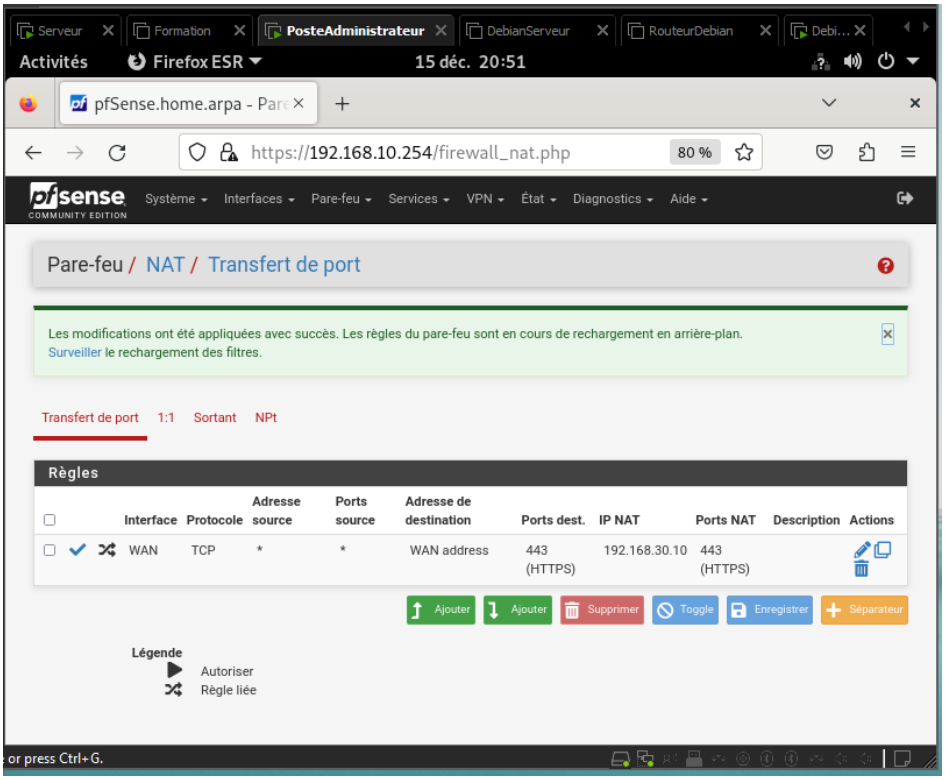
Pour la configuration du NAT Outbound (ou NAT sortant) nous allons laisser les règles automatiques de Pfense par défaut (il fonctionnera comme un PAT ici) :



- Toutes les machines de mon réseau pourront donc pouvoir accéder à internet à partir d'une seule adresse IP public.

Nous allons ensuite devoir configurer une règle de redirection de ports afin que notre serveur WEB (192.168.30.10) soit accessible depuis le WAN.

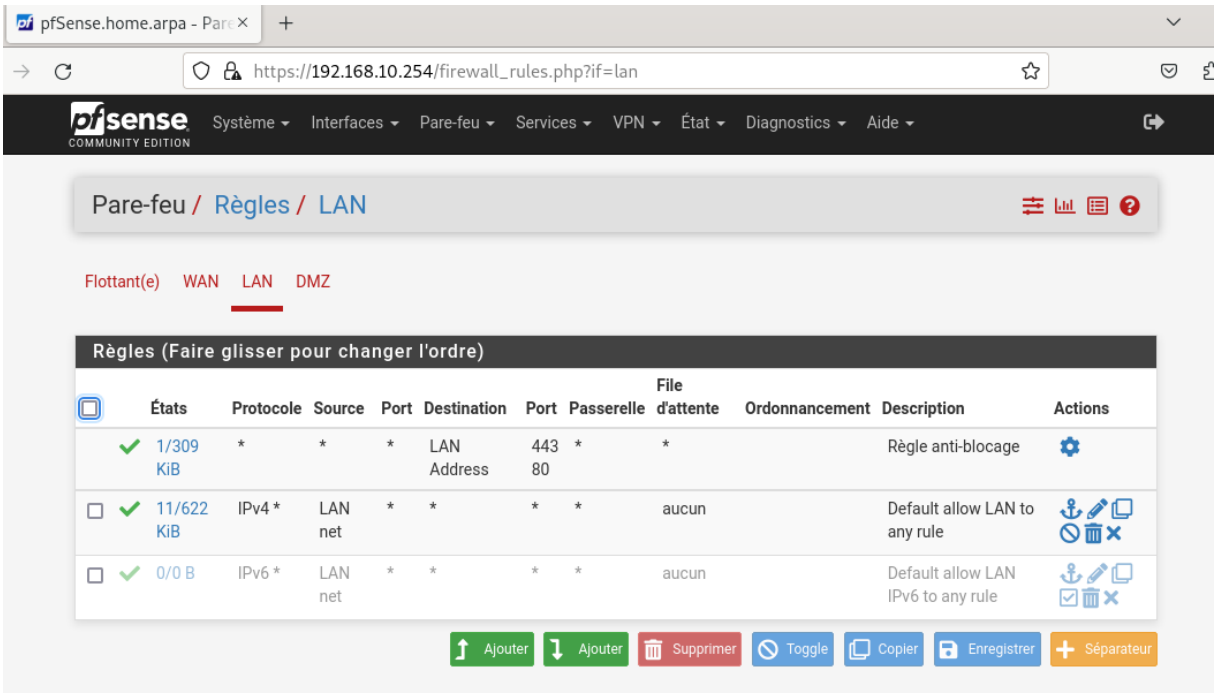
Nous devrons aller dans la partie Port Forward (ou Transfert de port en français) pour créer une nouvelle règle :



- Dans interface sélectionner : WAN
- Dans Famille d'adresse sélectionner : IPv4
- Dans Protocole sélectionner : TCP
- Dans Destination sélectionner : WAN address
- Dans Plage de port de destination sélectionner : HTTPS
- Dans IP de redirection cible choisir Hôte unitaire puis rentré son adresse ici celle du serveur web : 192.168.30.10
- Dans Port de redirection cible sélectionner : HTTPS

Nous allons pouvoir maintenant passer à la configuration des règles de filtrage de nos différents réseaux (LAN/DMZ/WAN).

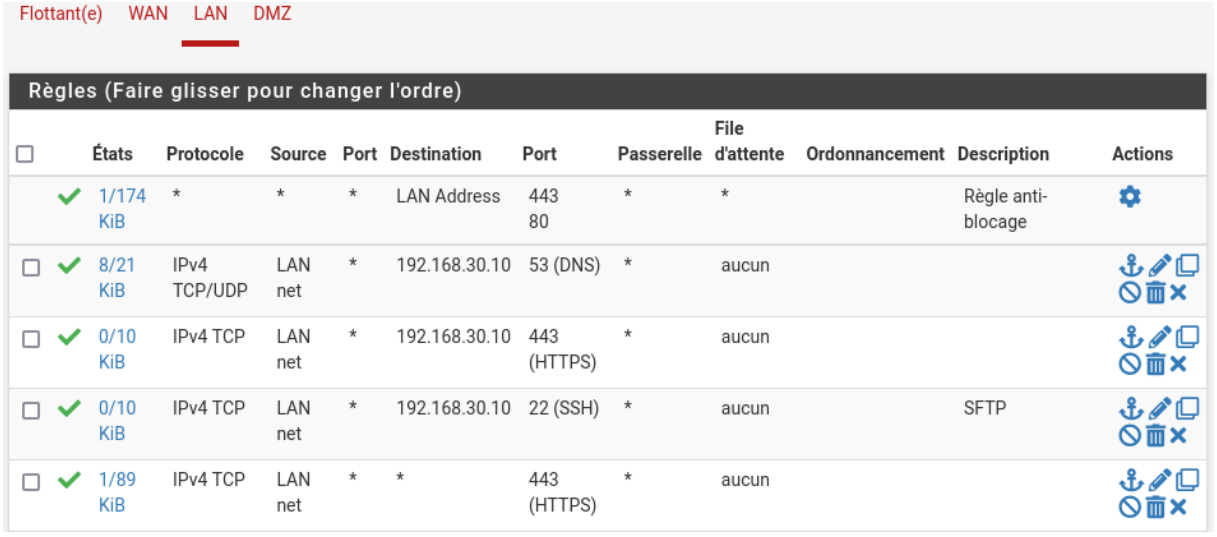
Règles de filtrage pour le LAN :



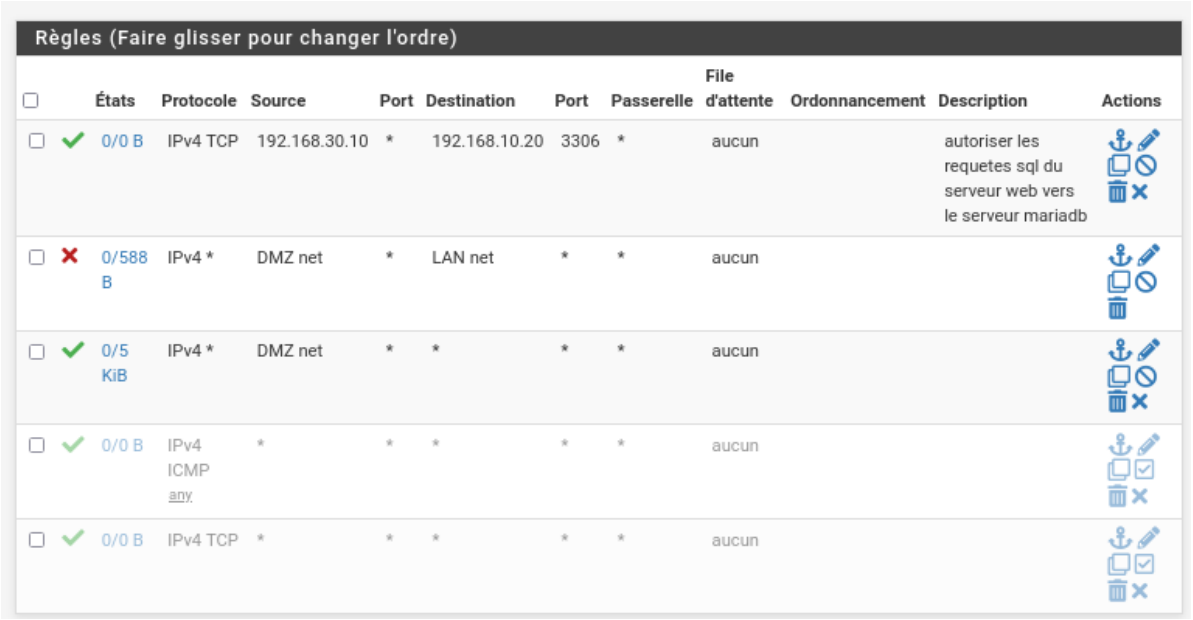
Par défaut, Pfsense autorise toutes les connexions sortantes. Ne souhaitant pas bloquer une connexion en particulier, nous pouvons laisser les règles par défaut.

Nous pouvons aussi autoriser uniquement les connexions voulues :

- Accès au serveur WEB.
- Accès au serveur FTP.
- Accès à Internet.



Règles de filtrage pour la DMZ :



Règles (Faire glisser pour changer l'ordre)

	États	Protocole	Source	Port	Destination	Port	Passerelle	File d'attente	Ordonnancement	Description	Actions
☐	✓	0/0 B	IPv4 TCP	192.168.30.10	*	192.168.10.20	3306	*	aucun		autoriser les requetes sql du serveur web vers le serveur mariadb	
☐	✗	0/588 B	IPv4 *	DMZ net	*	LAN net	*	*	aucun			
☐	✓	0/5 KiB	IPv4 *	DMZ net	*	*	*	*	aucun			
☐	✓	0/0 B	IPv4 ICMP any	*	*	*	*	*	aucun			
☐	✓	0/0 B	IPv4 TCP	*	*	*	*	*	aucun			

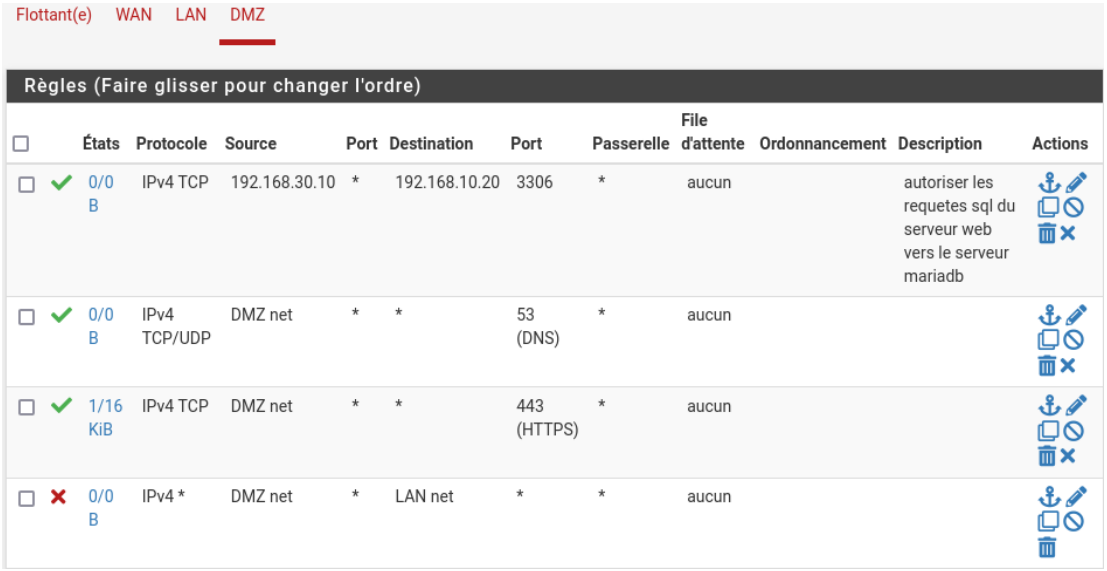
Nous souhaitons limiter les communications de la DMZ vers le LAN :

- On autorise le serveur web (192.168.30.10) à communiquer avec le serveur MariaDB (192.168.10.20) sur le port 3306.
- On bloque toutes les autres connexions vers le LAN

Puis :

- On autorise toutes les connexions vers les autres réseaux

Nous pouvons aussi autoriser uniquement les connexions voulues :

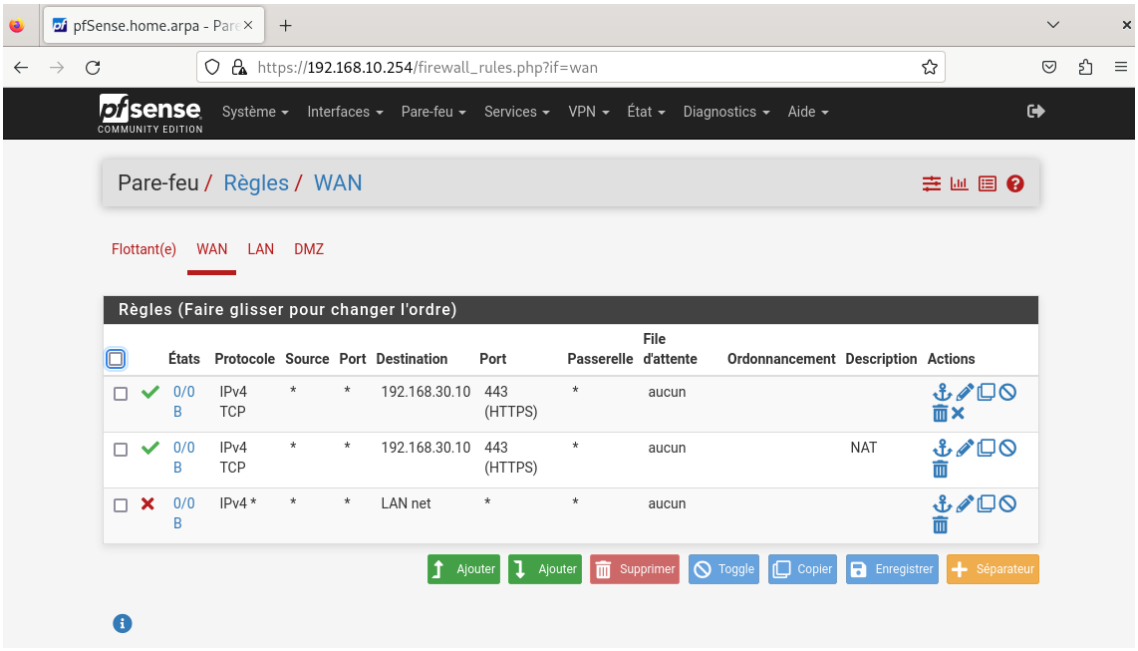


Flottant(e) WAN LAN DMZ

Règles (Faire glisser pour changer l'ordre)

	États	Protocole	Source	Port	Destination	Port	Passerelle	File d'attente	Ordonnancement	Description	Actions
☐	✓	0/0 B	IPv4 TCP	192.168.30.10	*	192.168.10.20	3306	*	aucun		autoriser les requetes sql du serveur web vers le serveur mariadb	
☐	✓	0/0 B	IPv4 TCP/UDP	DMZ net	*	*	53 (DNS)	*	aucun			
☐	✓	1/16 KiB	IPv4 TCP	DMZ net	*	*	443 (HTTPS)	*	aucun			
☐	✗	0/0 B	IPv4 *	DMZ net	*	LAN net	*	*	aucun			

Règles de filtrage pour le WAN :



- Nous allons autoriser les connexions HTTPS au serveur web (192.168.30.10) pour tous les postes du réseau WAN.
- On peut également voir la régle de redirection de port créé via le NAT.
- Nous bloquons l'accès au LAN.

Nous allons maintenant pouvoir tester nos règles de filtrage !

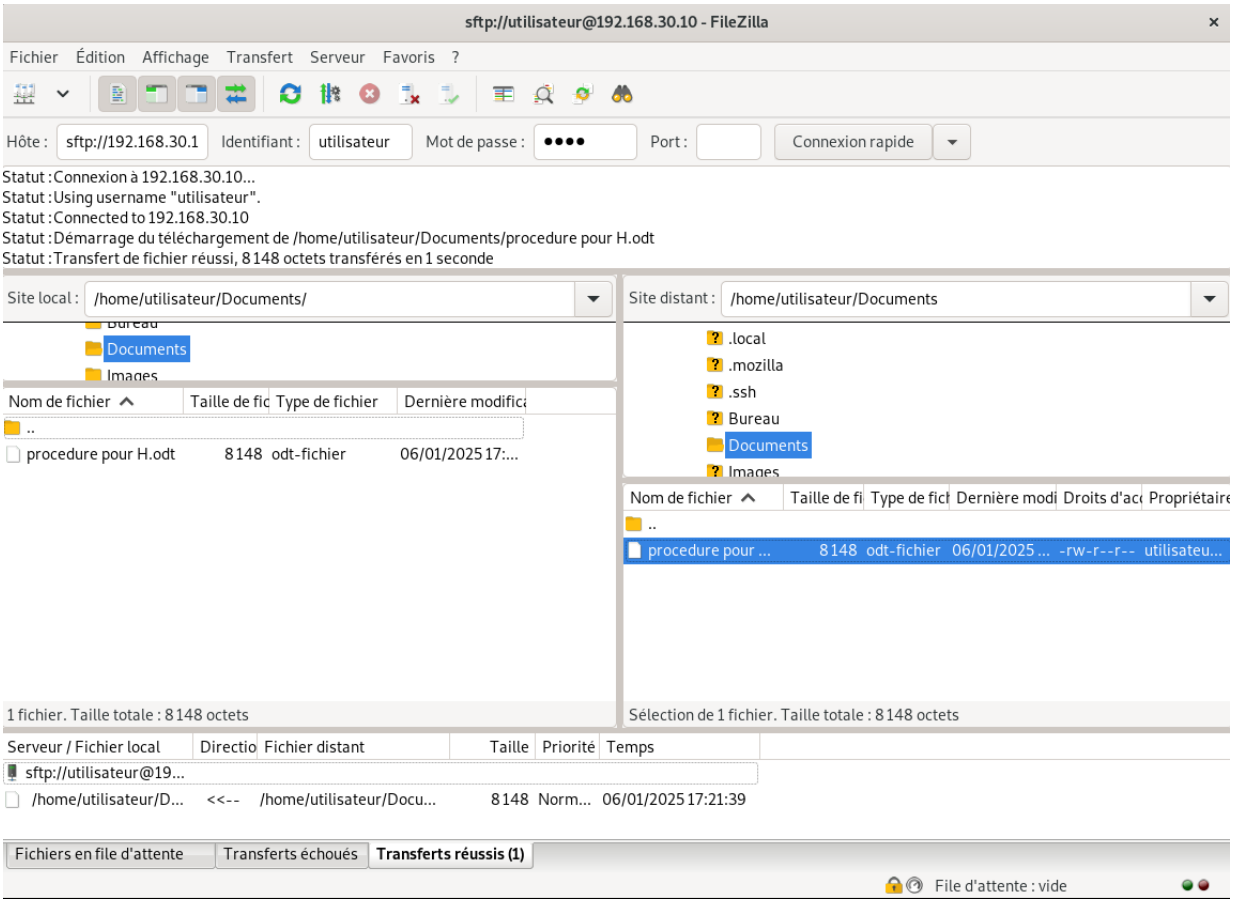
Test de Connexion :

- *Autoriser les accès au serveur Web depuis Internet (WAN) et depuis le LAN :*

Depuis le WAN :

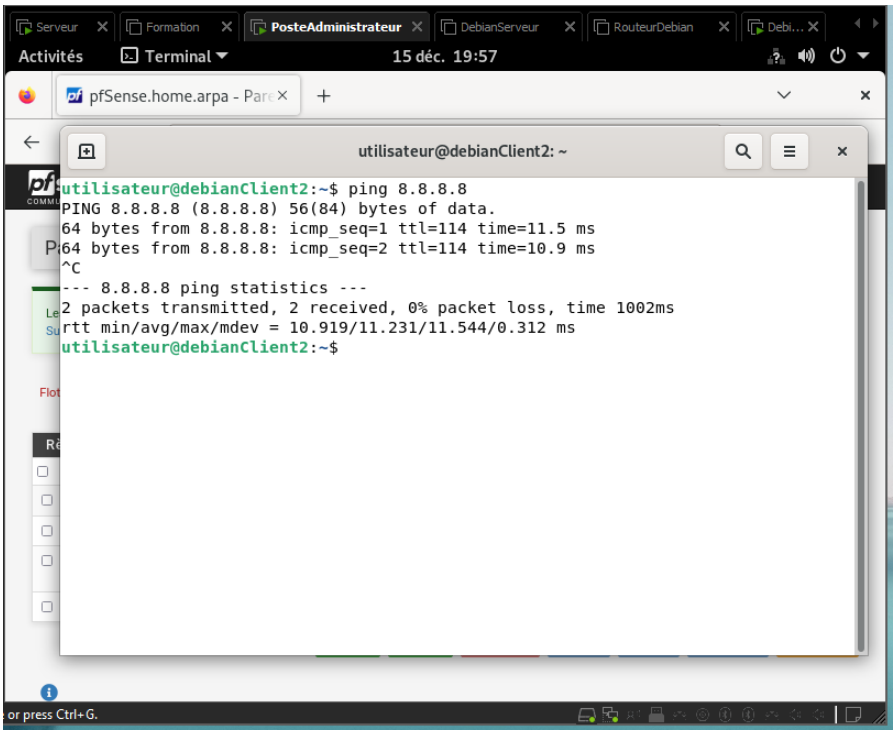
Depuis le LAN :

- *Autoriser les accès FTP sur le serveur de la DMZ depuis le LAN :*

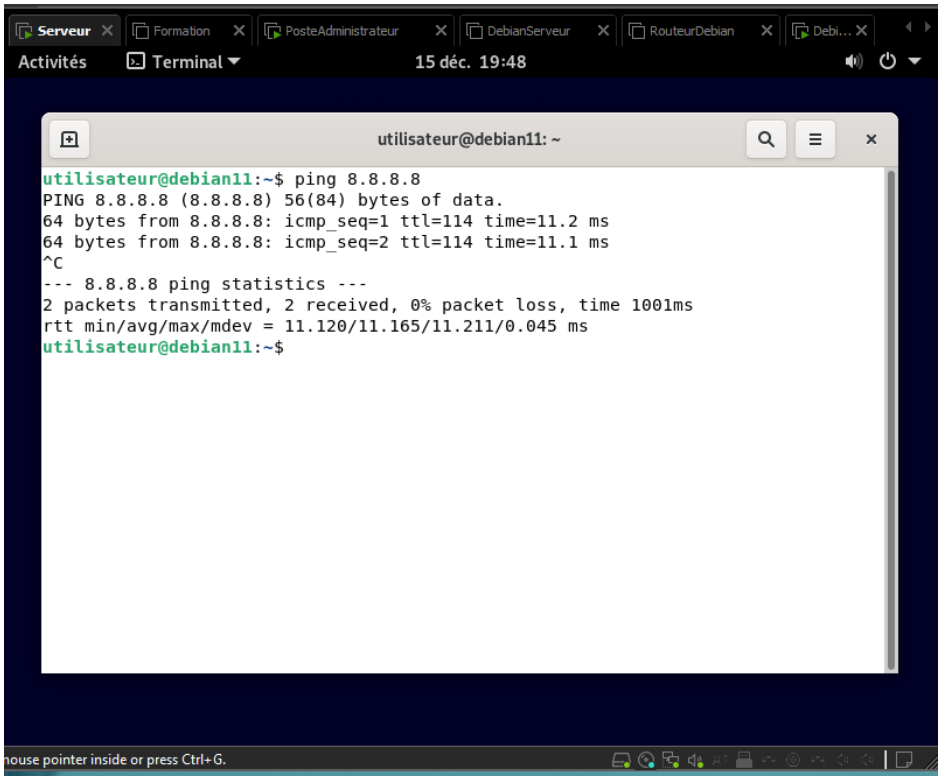


- *Autoriser les accès Internet depuis le LAN et la DMZ en passant par le Firewall :*

Depuis le LAN :

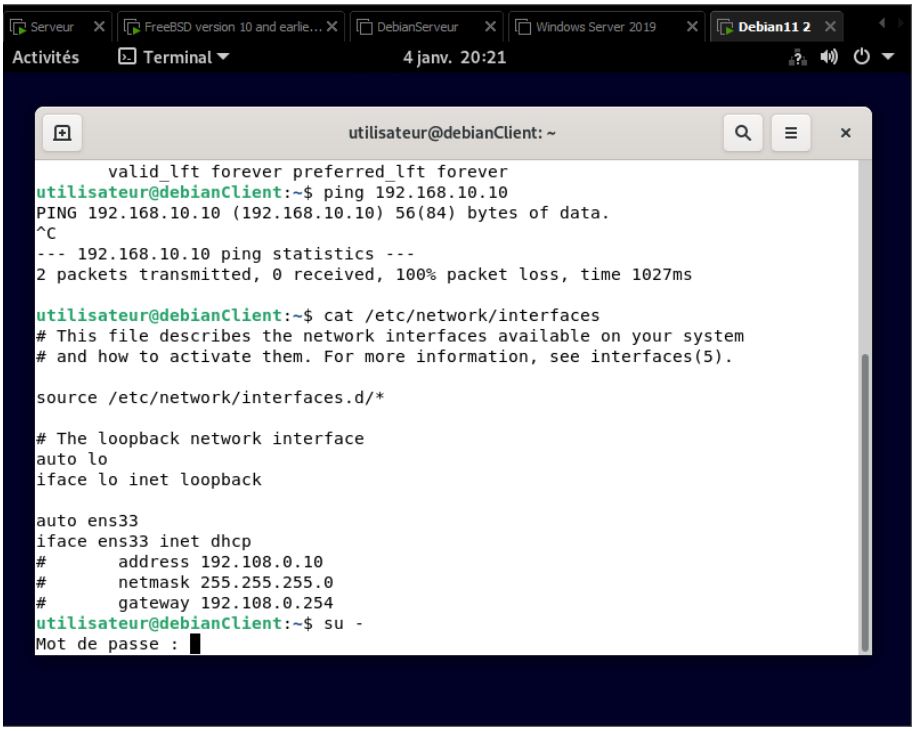


Depuis la DMZ :

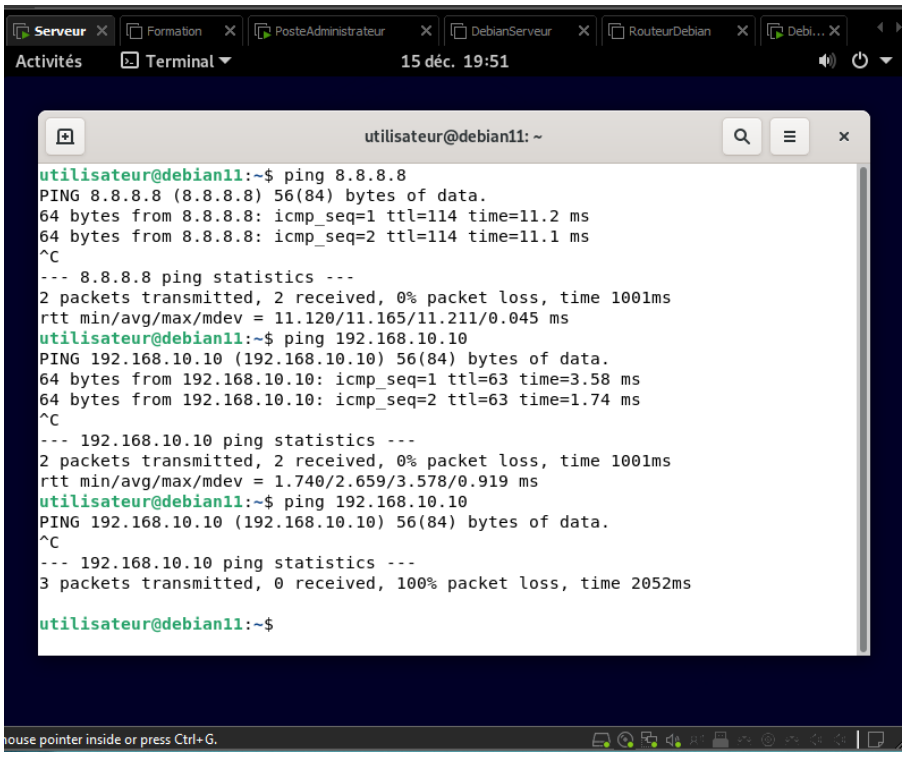


- *Interdire tout accès au LAN depuis l'Internet (WAN) ou la DMZ :*

Depuis le WAN :

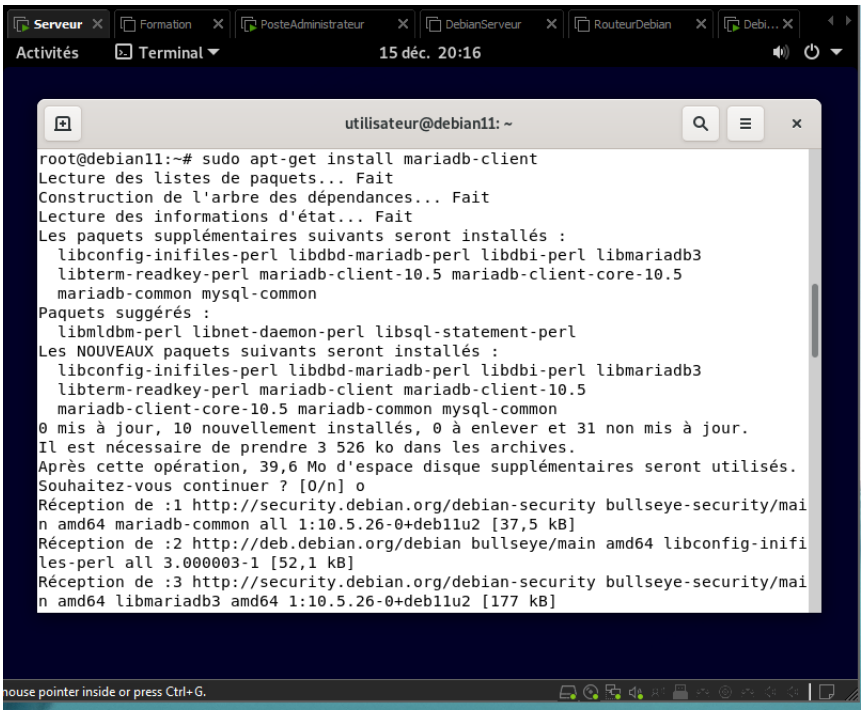


Depuis la DMZ :

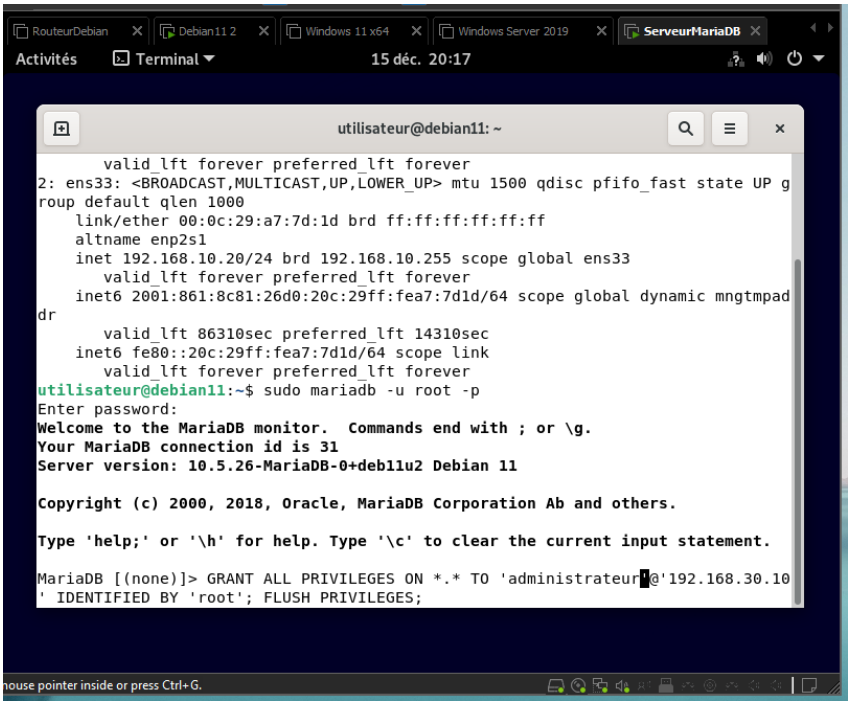


- *autoriser les requêtes SQL du serveur WEB vers le serveur MariaDB :*

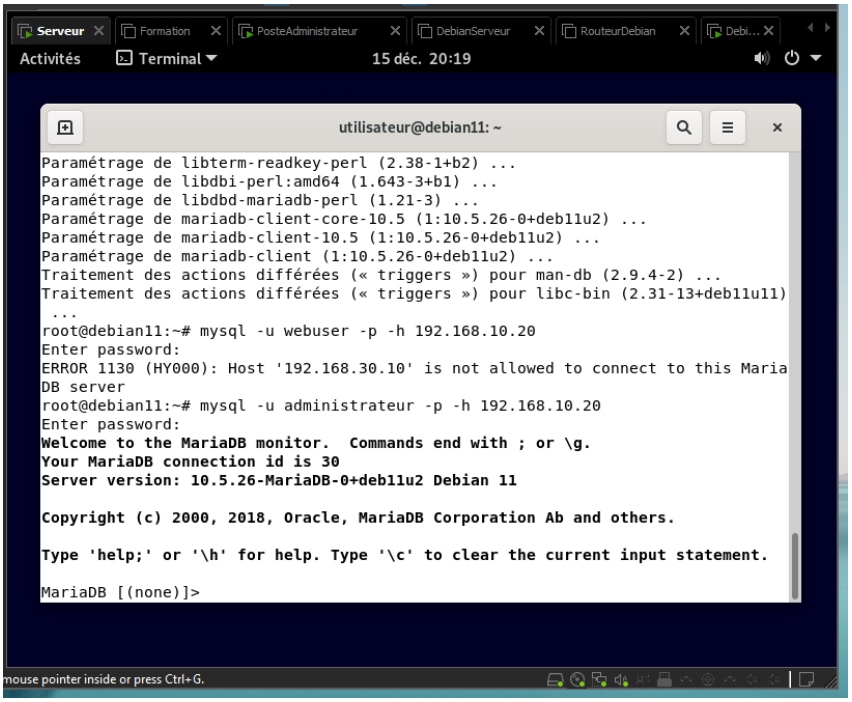
Afin de pouvoir nous connecter au serveur MariaDB nous devons au préalable installer mariadb-client sur notre serveur web :



Puis connexion au serveur MariaDB :



Connexion réussie !



Conclusion :

Tous les points du cahier des charges ont été respectés. Le serveur WEB est déployé dans un réseau DMZ, et le serveur de bases de données MariaDB est déployé dans le LAN. Les règles de filtrage ont été configurées via PFSENSE.

- Le serveur web est accessible depuis Internet et le LAN.
- Le serveur MariaDB est installé dans le LAN avec autorisation des requêtes SQL du serveur web.
- Les règles de sécurité permettent les accès FTP et Internet selon les spécifications, tout en interdisant l'accès au LAN depuis Internet et la DMZ.

Ce projet a permis de sécuriser les accès aux serveurs et d'améliorer la gestion des services accessibles depuis l'extérieur. Tous les objectifs ont été atteints avec succès.