Holodeck

**SOMMAIRE**

1. Installation & configuration vm serveur
   1. Configuration carte réseau en Local
   2. Configuration OS debian 12
2. Installation & configuration serveur DHCP
3. Configuration vm client
4. Installation & config du serveur DNS sur carte LAN
5. Installation & config du serveur web Nginx

5.1. Mettre la dernière version 1.26.2 de nginx

5.2. Installation et conf pour certificat SSL

1. Installation & config php8 php7
2. Installation mariadb
3. Installation phpadmin
4. Configurer sftp
5. Installer et configurer cockppit

**OBJECTIF :**

À bord de l'USS Enterprise-D, la Fédération des Planètes Unies souhaite équiper ses ingénieurs du Holodeck de **machines virtuelles pour faciliter le développement de leurs sites web stellaires.**

**CONFIGURATION DEMANDER :**

* **Créer 1 VM SERVEUR** qui va servir de **serveur Web, PHP, SQL**

**CONFIG :** Sans interface graphique

2 go RAM

2 vpcu

disque 32 g et 2 cartes réseaux ( une WAN et une LAN)

va servir de Serveur DHCP / DNS (sur la carte «LAN») de serveur

FTP, serveur Web, serveur de base de données SQL et serveur LDAP.

* **Créer 1 autre VM Debian Client**

CONFIG VM CLIENT : Avec interface graphique ( GUI ) - 2go RAM - disque 16 g et connecté sur le LAN

Les contraintes :

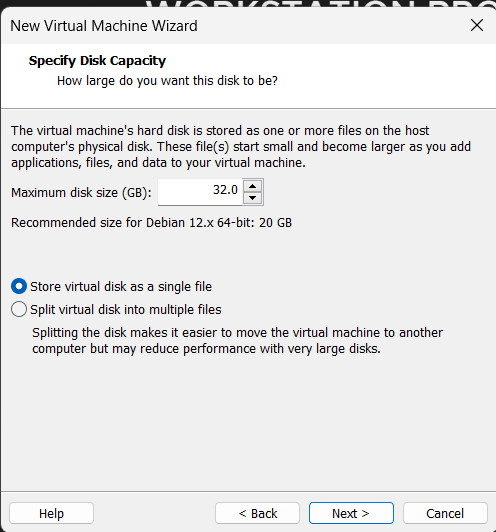
* Pas de compte sudo, conformément aux directives de Starfleet.
* Mise en place d'un pare-feu pour autoriser uniquement les ports requis.
* Le serveur Web doit être Nginx et en HTTPS.
* PHP , MariaDB , et Nginx doivent être la dernière version (pas celle des
* dépôts Debian).
* Pour PHP, la version 7.x et 8.x doivent cohabiter
* Mise en place du serveur DHCP/DNS, domaine : starfleet.lan
* Au Niveau Web :
* www8.starfleet.lan ⇒ site web en php8
* www7.starfleet.lan ⇒ site web en php7
* php.starfleet.lan ⇒ phpMyAdmin
* admin.starfleet.lan ⇒ administration de la VM
* Un serveur FTP (en SSL/TLS) pour copier les fichiers du serveur Web (chrooté sur le dossier web).
* Création d’un certificat SSL qui serviras pour le serveur Web et le Serveur FTP.
* Pour authentifier les Utilisateurs, le serveur Web utilisateur un annuaire LDAP.

1. **INSTALLATION & CONFIGURATION VM SERVEUR**

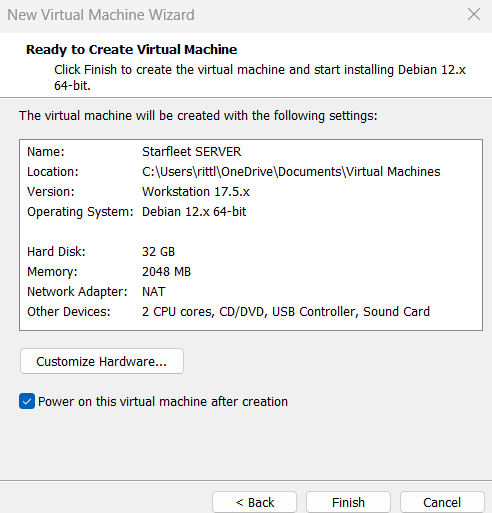
Pour commencer, nous choisissons un hyperviseur dans notre cas VM Ware Workstation

Puis créons notre nouvelle VM que l’on appellera **« STARFLEET** **SERVER »** pour que cela soit plus lisible.

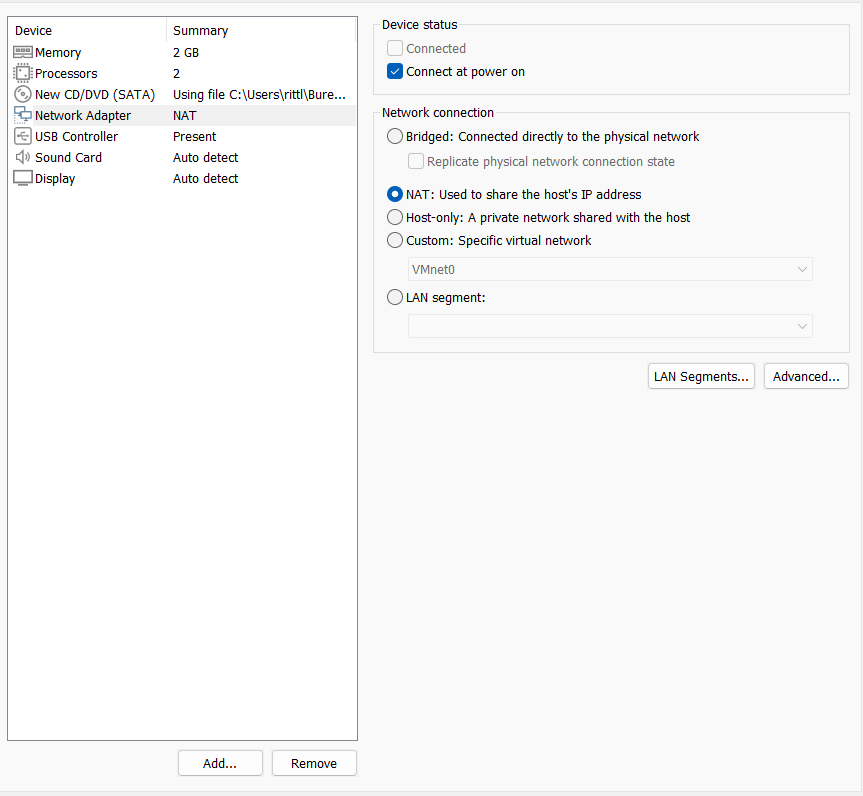
On définit la capacité du disk de notre machine comme precisé plus haut :



Puis on se rend dans « CUSTOMIZE HARDWARE » pour configurer notre vm :



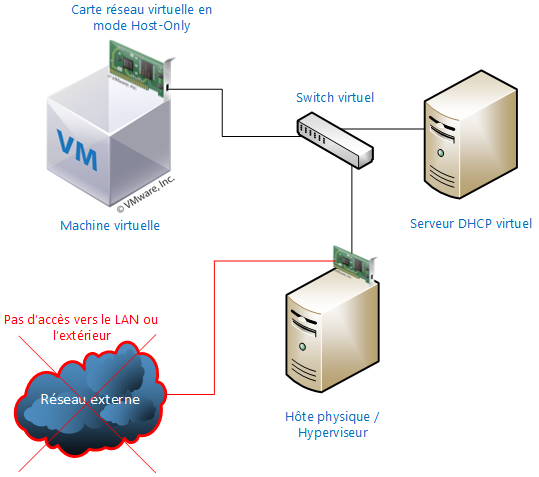
On vérifie la ram, les cœurs de base ils sont à 2 c’est ce que nous voulons



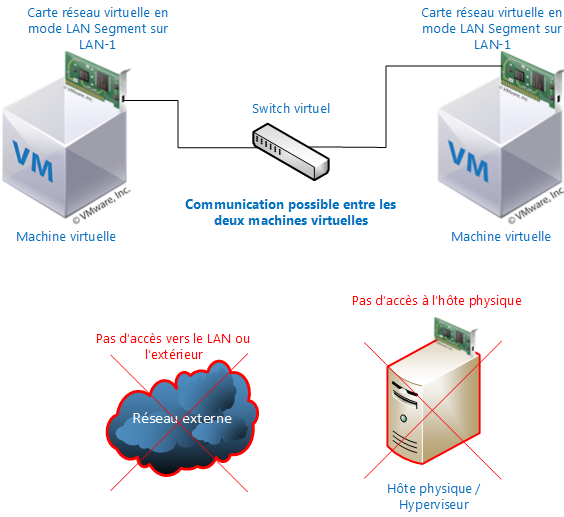
* 1. Configuration carte réseau en Local

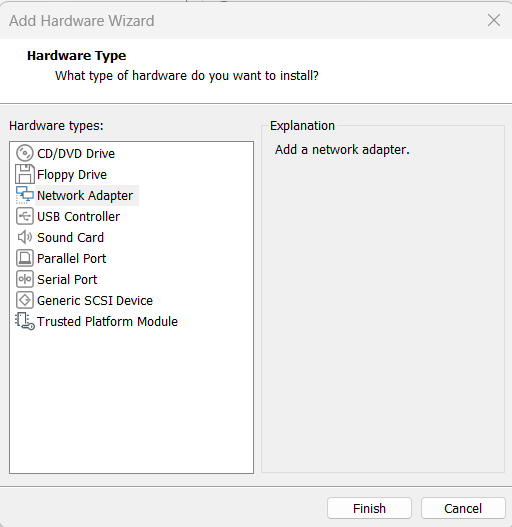
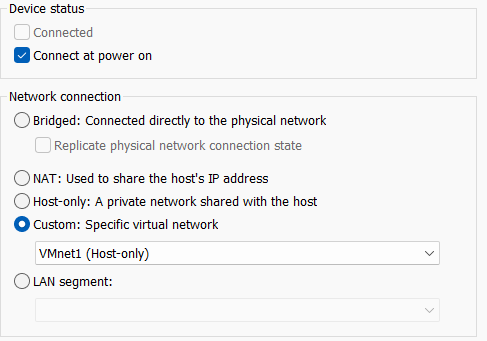
On ajoute notre seconde carte réseau que l’on configure en LAN pour cela il y a 2 solutions dont une qui est plus sécurisée et donc conseillé.

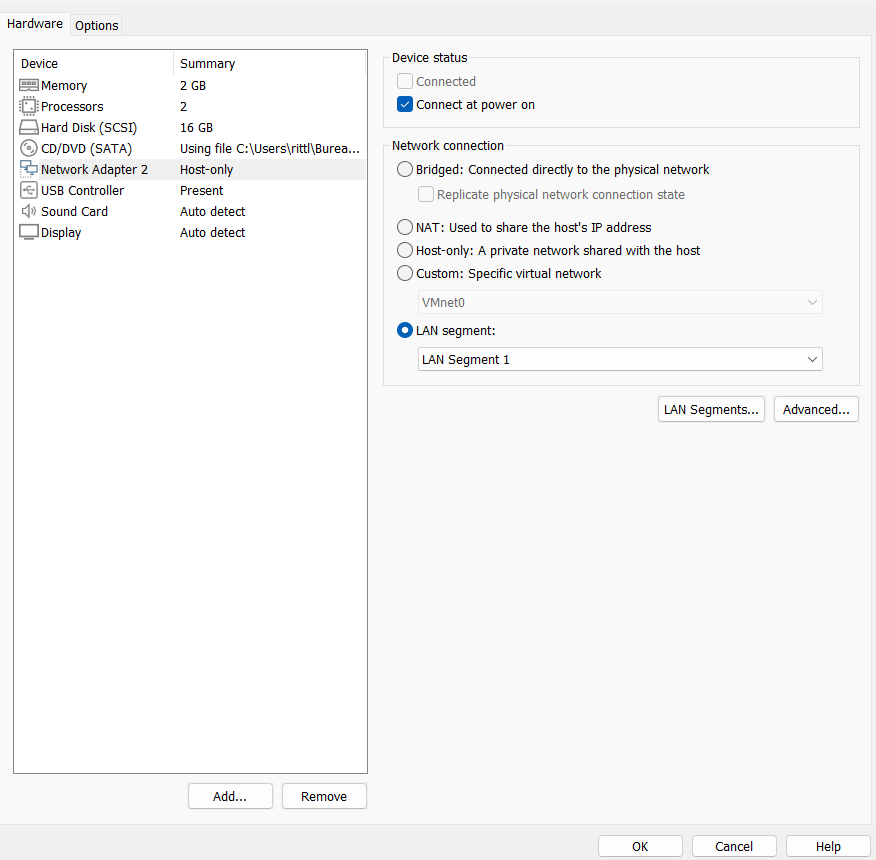
La première consiste a utilisé une carte réseau en host-only qui permet a nos 2 vm d’être dans le réseau local sans avoir une connexion internet (ce que nous voulons) mais notre machine hôte peut communiqué avec notre machine hôte (machine physique) ce qui n’est pas optimal en terme de sécurité.



Donc nous avons choisi de configuré notre carte réseau en LAN segment ce qui nous permet d’avoir une communication uniquement entre nos vm appartenant à ce réseau sans que la machine hôte puisse communiquer avec. Donc plus sécurisé

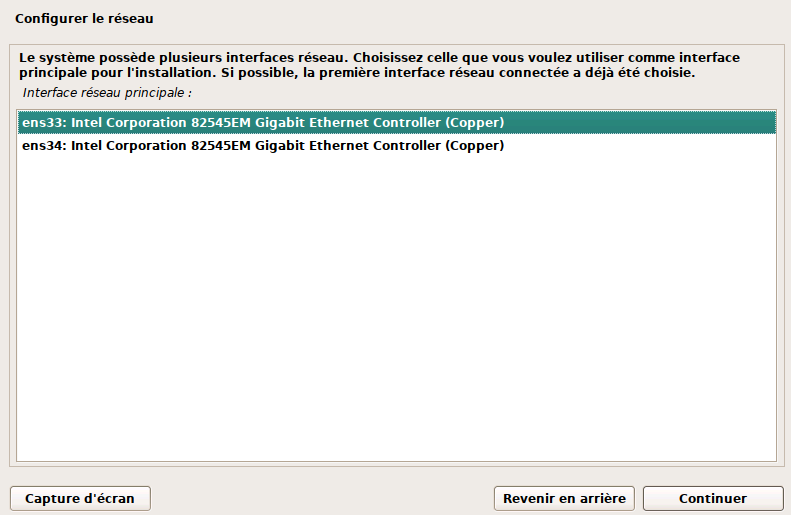


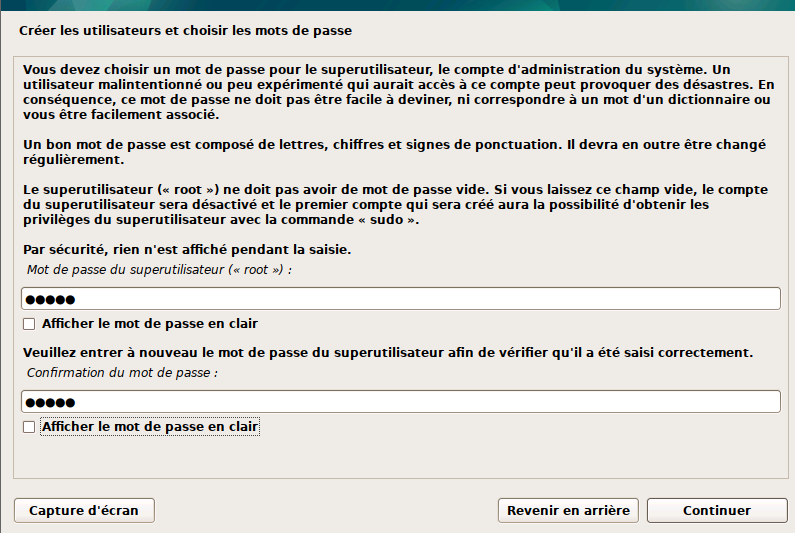


* 1. Configuration OS debian 12

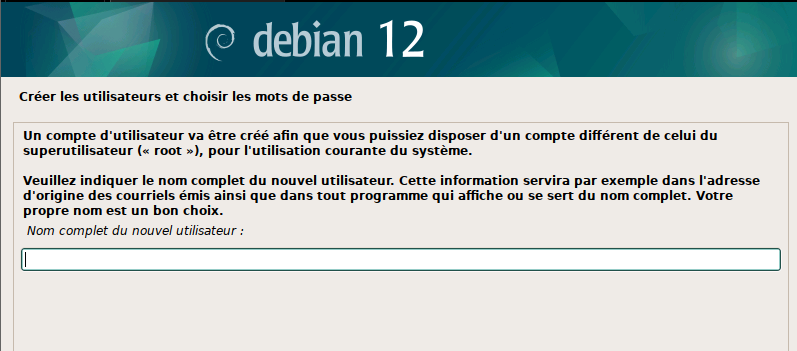
Maintenant on configure le système d’exploitation choisi, on choisit la carte réseau **« ENS33 »** car nous l’avons configuré en NAT pour récupérer une connexion internet.



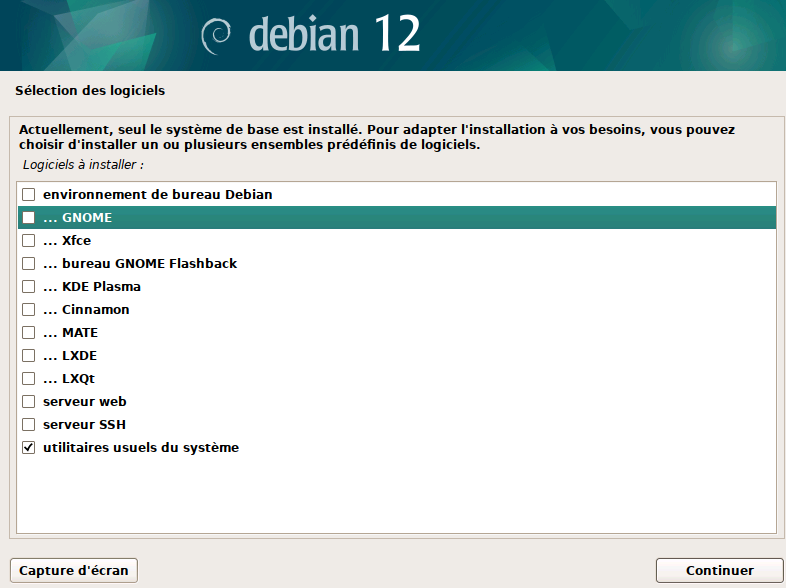
En suite on mais un mot de passe a notre root :

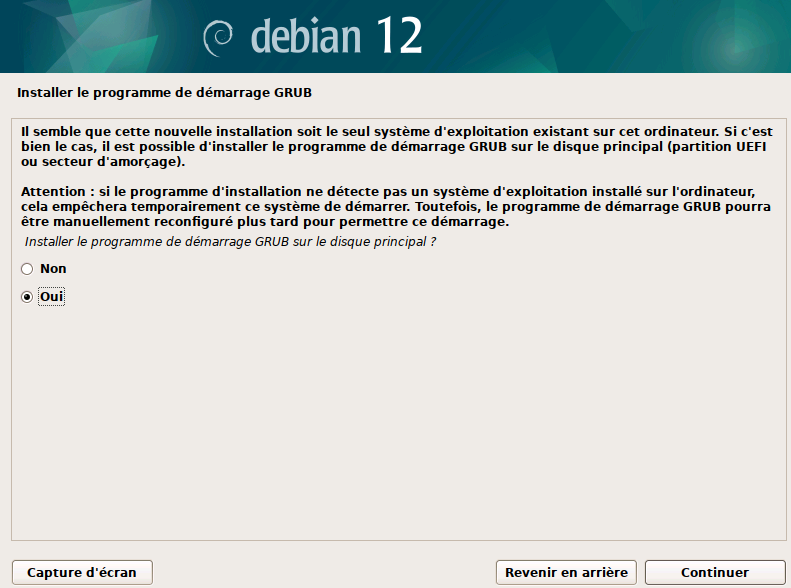


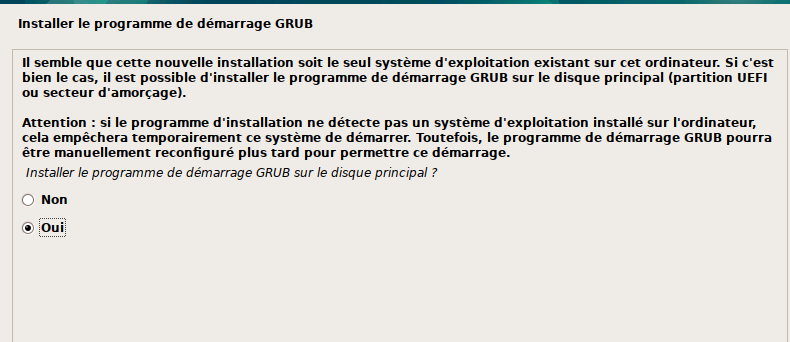
On crée un compte utilisateur

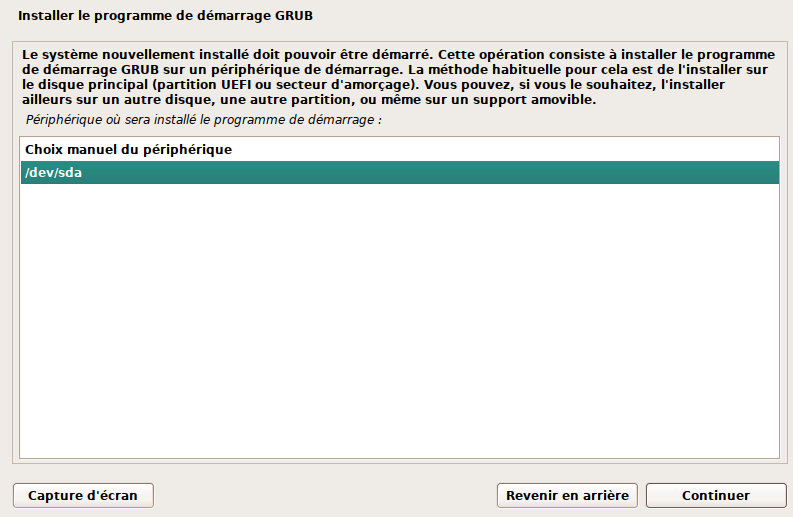


Nous ne voulons pas d’environnement de bureau sur cette vm donc nous le décochons











1. **INSTALLATION & CONFIG DU SERVEUR DHCP**

***RAPPEL :***

*Un serveur DHCP permet de fournir automatiquement une configuration IP a une autre machine. Cette configuration IP est composée :*

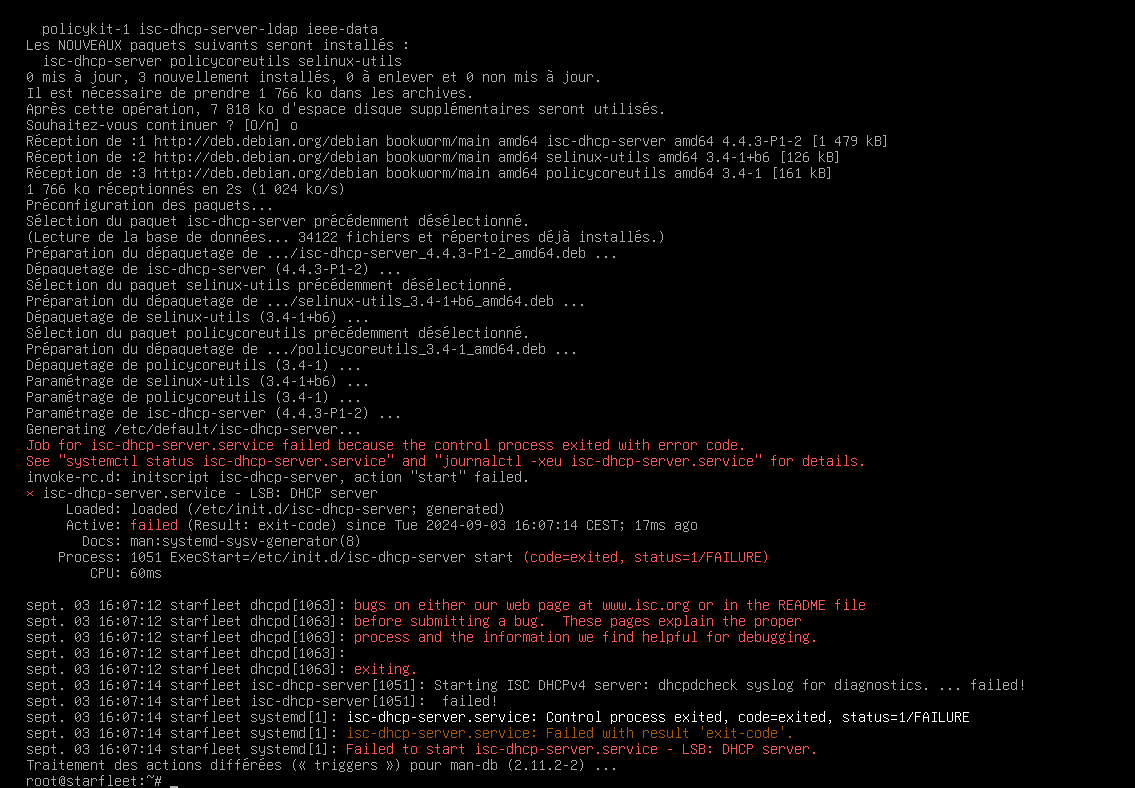
* *D’une adresse IP*
* *D’un masque de réseau*
* *D’une passerelle*
* *D’une adresse de DNS*

On doit installer un serveur DHCP sur notre vm server en utilisant la seconde carte réseau que nous avons configurer en LAN

Pour ce faire on commence par faire une mise a jour des paquets avec la commande suivante 

Puis on installe le serveur avec la commande suivante :  


Voici le résultat :



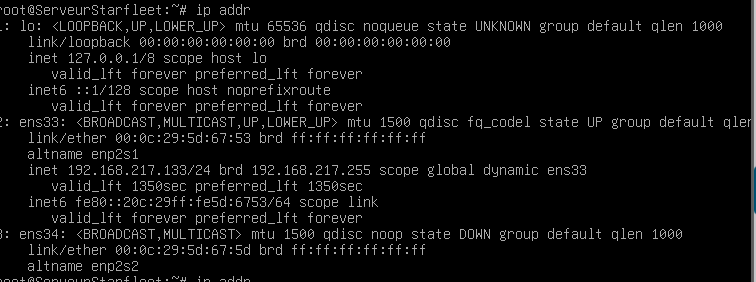
Nous constatons des erreurs ce qui est normal a cette étape.

*Nous allons donc maintenant le configurer :*

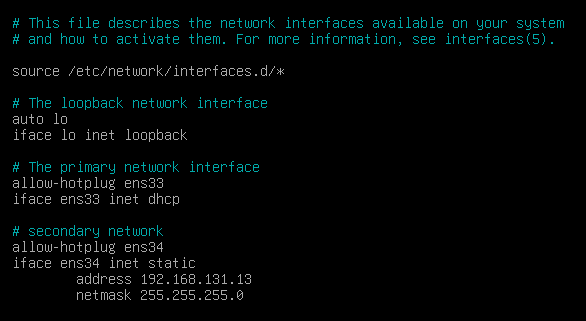
*On commence par aller dans le fichier et on va indiquer quelle interface réseau nous utiliserons :*

**

**

**

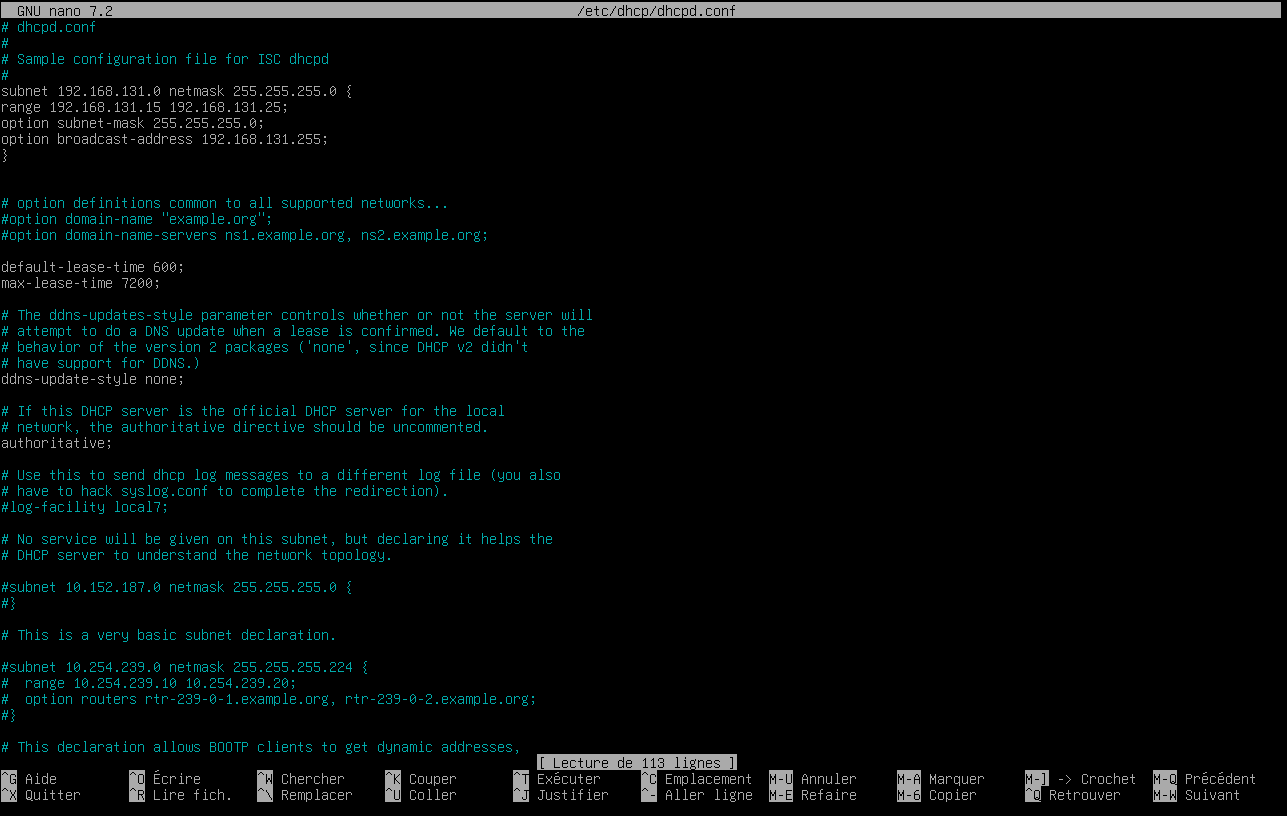
*On se rend dans le fichier qui gere les cartes réseaux avec la commande suivante :*

**

**

*Apres on se rend dans le fichier de configurer les plages pour attribuer les adresses ip dans ce range.*

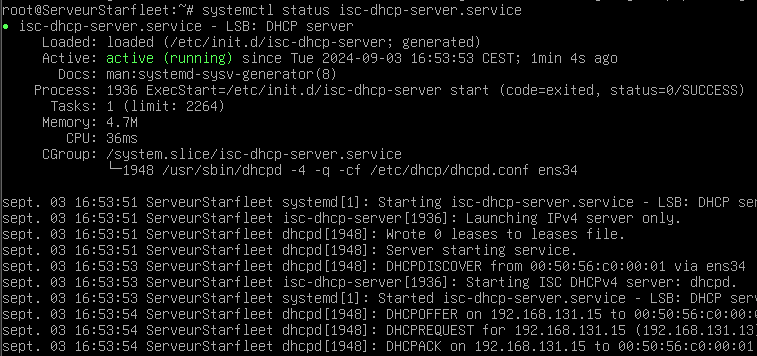
**

**

*On redémarre le service dhcp*

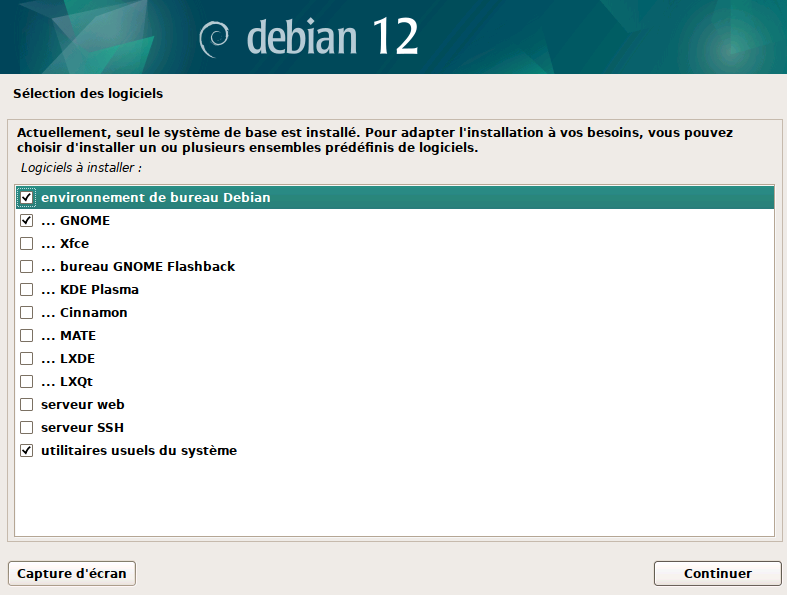
**

*Pour vérifier si notre serveur est bien actif, il suffit de taper cette commande.*

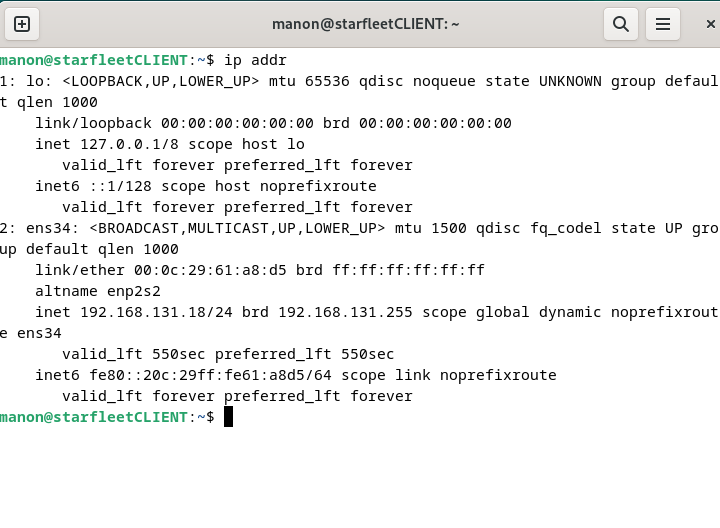
**

1. **CONFIGURATION VM CLIENT**

Pour installer notre seconde vm nous suivons les memes étapes que la premiere sauf au niveau du choix de l’environnement de bureau debian graphique, nous en choisirons une pour celle-ci.



Nous pouvons voir ici que notre serveur dhcp a bien attribué un adresse ip correspondant a notre réseau LAN

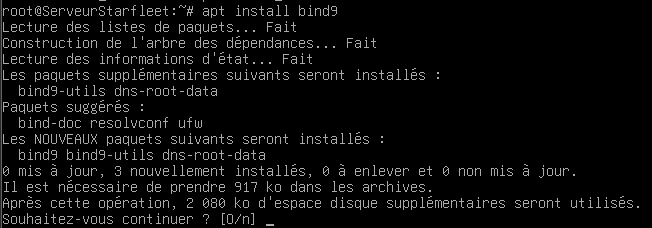


1. Installation & config serveur DNS sur LAN

Pour ce faire, nous allons sur notre vm serveur. Et nous allons intallé le logiciel bind9

***Rappel :* BIND9 (Berkeley Internet Name Domain, version 9)** est un logiciel open-source qui implémente **le protocole DNS**, permettant **la résolution des noms de domaine en adresses IP.** C'est l'un des serveurs DNS les plus utilisés au monde, tant pour les domaines publics sur Internet que pour les réseaux privés

*On commence donc par installé notre logiciel bind9avec la commande suivante :*



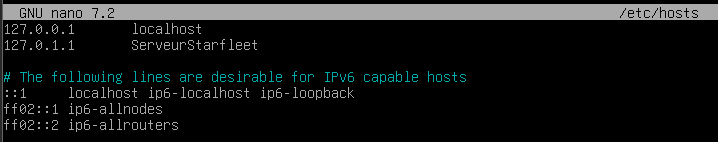
Apres on se rend dans le fichier **/etc/hostname** contient **le nom de la machine,** c'est-à-dire le **nom d'hôte**. Ce nom est utilisé par le système pour identifier **la machine sur le réseau**.

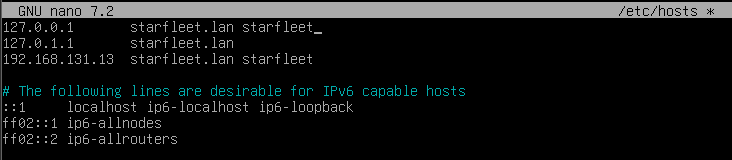
**Pourquoi renseigner le domaine en plus du nom d'hôte ?**

C’est pour associer un nom de domaine **pour spécifier pleinement le domaine auquel appartient cette machine**. Par exemple, si ton serveur s'appelle TRUCMUCH et fait partie du domaine .COM, on peut le spécifier cela dans **/etc/hostname**.

*C’est ce que l’on fait ici en renommant notre vm* ***starfleet.lan***  

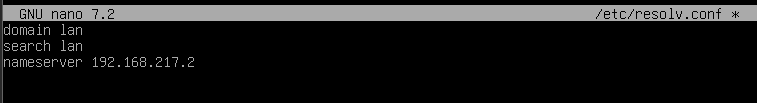

*Puis on se rend dans le fichier hosts pour ajouter l’adresse ipv4 de notre serveur ainsi que son* ***FQDN*** *qui signifie textuellement :* ***« Fully Qualified Domain Name »,*** *que l’on pourrait traduire par « Nom d’hôte pleinement nommé »*

*Voici le fichier d’origine :  
*

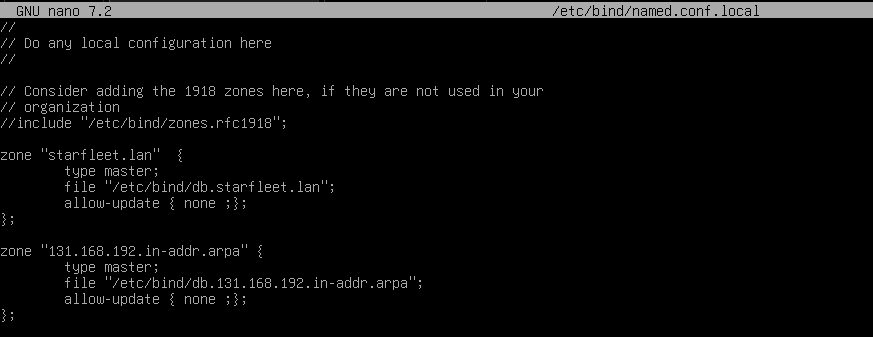
*Voici les modifications apportées :  
*

*Après on se rend dans le fichier* ***resolv.conf*** *avec la commande suivante :  
*

*Voici le fichier d’origine :  
*

*Voici le fichier modifier :   
*

*On fait une copie de notre fichier* ***named.conf.local*** *pour garder un fichier intact en cas de disfonctionnement :  
*

*Puis on se rend dans le fichier* ***named.conf.local*** *pour déclarer les zones  
*

*Explication :*

*Zone « starfleet.lan » : On définit le nom de la zone*

*Type master : On indique que ce serveur fait autorité sur la zone*

*File « /etc/bind/db.starfleet.lan » : On indique le lien vers le fichier contenant la base d’enregistrements pour la zone.*

*Allow-update { none ;} : On n’autorise pas les mises à jour du fichier d’enregistrements par un tiers, ce qui permet d’augmenter la sécurité et être sûr qu’il n’y ait que le serveur DNS qui s’occupe de la zone.*

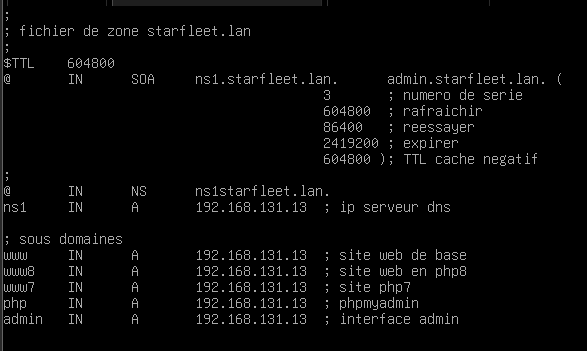
**Zone normales et/ou directe :**

Elles permettent de retrouver l'adresse IP à partir du nom.

**Zone inversée :**

Elles permettent de retrouver le nom à partir de l'adresse IP, c'est à dire de faire l'inverse des zones normales. Ce qui permet une résolution dans les deux sens.

*Maintenant nous allons construire notre 1er fichier que nous venons de déclarer pour la zone directe*



Explication construction fichier zone :

$TTL 604800 : correspond à la durée de vie des informations fournies et donc la durée pendant laquelle sont gardées en cache. 604800 secondes correspondent à une durée de 7 jours.

SOA : signifie « start of authority, ceci indique le serveur qui a autorité sur la zone, puis l’adresse mail du contact technique dont le caractère « @ » est remplacé par un «.»

@ : Ce symbole permet de représenter le nom de domaine de la zone

Les types d’enregistrement :

NS (Name Server) : Sert à définir le ou les serveurs DNS qui répondent à cette zone.

A : Enregistrement d'un hôte ayant une adresse IPv4 (32 bits).

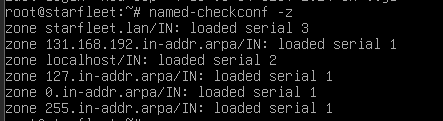
AAAA : Enregistrement d'un hôte ayant une adresse IPv6 (128 bits).

CNAME (Canonical Name) : Enregistrement qui sert d'alias à un enregistrement "A" déclaré auparavant.

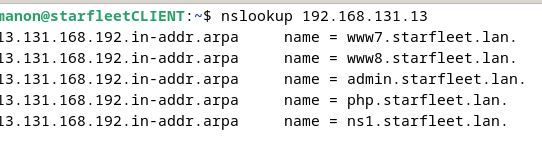
Nous avons besoin de créer des enregistrements de type "AAAA" et "CNAME", l'enregistrement de type "NS" étant déjà définit par défaut.

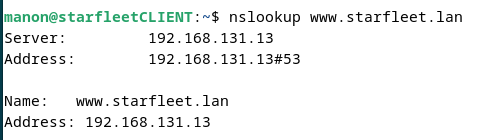
Après nous créons le second fichier correspondant à la zone inverse   




Vérification de la configuration par la commande suivante :

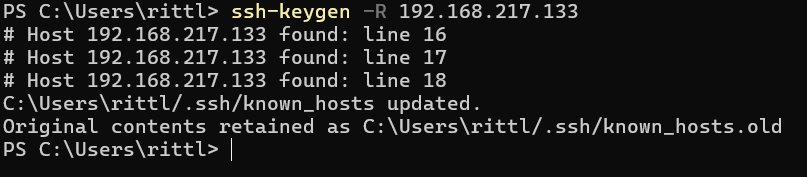


*Vérification sur vmClient* zone inversé  


*Vérification sur vmCLient zone normales*

1. Installation & config du serveur web Nginx

Probleme clé ssh sur terminal



*On commence par faire* ***une mise à jour de la liste des paquets*** *déjà installés puis on fait la commande pour* ***installer Nginx***  


Dès l’installation fini on peut se rendre sur notre vm cliente qui dispose d’une interface graphique et tester en ouvrant un navigateur en tapant 

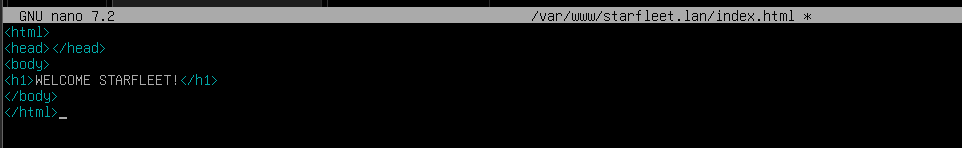


Nous allons maintenant déclarer un nouveau site sur notre serveur web Nginx

*Pour se faire, on commence à créer un dossier qui va accueillir notre site  
*

*Ensuite, on va déclarer l’utilisateur « www.data comme propriétaire de ce dossier. Il s’agit de l’utilisateur par défaut de Nginx* *(correspondant à la propriété "user www-data" du fichier nginx.conf)   
*

*Puis les droits du dossier  
*

*On créer notre fichier « index.html » qui correspondra à la page d’accueil du site web.  
*

*Maintenant on crée le fichier de configuration de notre site. Dans le dossier ‘sites-available’, on créer le fichier ‘starfleet.lan’   
*

*Explication :*

La**directive "listen"***:* la première ligne permet d'indiquer que le serveur Nginx écoute sur toutes ses adresses IPv4, sur le port 80, ce qui correspond au protocole HTTP. La seconde ligne est similaire, mais pour toutes les adresses IPv6 du serveur, toujours sur le port 80.

**root /var/www/starfleet.fr** ; : permet de déclarer la racine du site Internet : en toute logique, on précise la racine que l'on a créée précédemment et où se situe la page index.html

**server\_name :** sert à déclarer le nom de domaine, ou les noms de domaine, concerné par ce bloc "*Server*". On peut également utiliser une adresse IP.

***Location* :** permet d'indiquer un chemin relatif dans l'URL. En indiquant "/", on cible toutes les requêtes puisqu'une requête commence toujours par "/" après le nom de domaine pour spécifier le chemin vers une page.

"***try\_files*" suivie de $uri et $uri/**: nous allons chercher à vérifier l'existence du fichier ou du dossier (d'où le "/") passé en paramètre dans l'URL. La variable $uri reprend automatiquement l'URL saisie par le client qui accède au site. En fait, la règle "**try\_files $uri $uri/ =404;**" permet de retourner une erreur 404 (page introuvable)

***Pour que notre site soit actif et la configuration chargée par Nginx, nous devons créer un lien symbolique****: rappelez-vous de l'intérêt du dossier "sites-enabled". Pour créer un lien symbolique et renvoyer "/etc/nginx/sites-enabled/starfleet.lan" vers "/etc/nginx/sites-available/startfleet.lan", voici la commande pour activer la config du site starfleet  
*

*Intérêt du dossier "sites-enabled" :*

Séparation des configurations actives et disponibles :

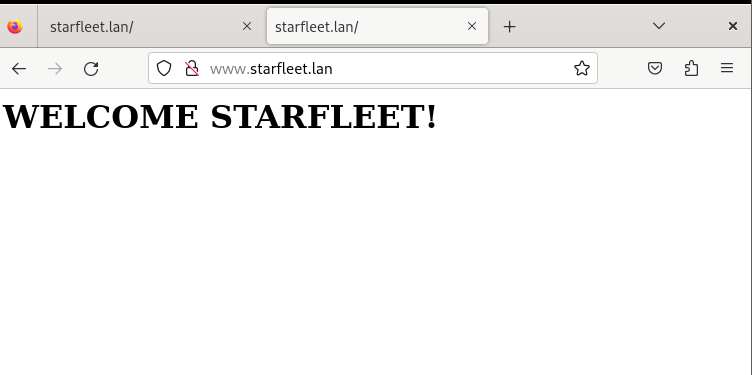
**"sites-available"** contient toutes les configurations possibles de serveurs (sites) que vous pourriez vouloir utiliser, mais ces configurations ne sont pas activées par défaut.

**"sites-enabled"** contient uniquement les configurations actives, c'est-à-dire les sites web que Nginx doit effectivement servir.

Activation des sites par lien symbolique *:*

Créer un lien symbolique (ln -s) entre un fichier de configuration dans "sites-available" et le dossier "sites-enabled" active ce site.  
Cela permet d'activer ou désactiver un site facilement sans supprimer son fichier de configuration. Si vous supprimez le lien dans "sites-enabled", le site est désactivé, mais la configuration reste intacte dans "sites-available".  
Il devient facile d’ajouter ou de désactiver rapidement des sites web, ce qui simplifie la gestion des serveurs Nginx.  
Vous pouvez préconfigurer plusieurs sites dans "sites-available" et simplement les activer au besoin en créant le lien symbolique.

*Vérification de la config  
*

*On redémarre le nginx et on se rend sur la vmClient pour tester sur le navigateur  
*

* 1. **Mettre la dernière version 1.26.2 de Nginx**
* J’ai choisi de faire un upgrade Nginx sur la version déjà installé 1.22 pour des raisons de compabilité. Quand j’installais nginx 1.26.2 directement, il manquait des fichiers de configuration et ça fonctionnait une fois sur 2.

Télécharger et ajouter la clé GPG correcte pour Nginx :

curl -fsSL https://nginx.org/keys/nginx\_signing.key | gpg --dearmor -o /usr/share/keyrings/nginx-archive-keyring.gpg

Vérifier que le dépôt Nginx est bien ajouté à la liste des sources : Ouvrez le fichier de liste des dépôts :

nano /etc/apt/sources.list.d/nginx.list

deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] http://nginx.org/packages/debian/ bookworm nginx

apt update

apt upgrade nginx

nginx -v

Modification du fichier de configuration nginx.conf :

On ouvre et modifie le fichier de configuration Nginx pour mettre à jour le bon utilisateur (www-data) et les chemins :

* 1. **CERTIFICAT SSL POUR HTTPS**

*On commence par une clé privée et le certificat ssl, on commence par créer le dossier dans lequel nous souhaitons mettre notre certificat puis on le créer avec la commande suivante :*

*Explication des options :*

openssl req *:* Utilise OpenSSL pour générer une requête de certificat (CSR) ou créer un certificat auto-signé.

*-new :* Crée une nouvelle demande de certificat (CSR).

*-newkey rsa:4096 :* Génère une nouvelle clé privée RSA de 4096 bits en même temps que la CSR.

*-x509 : Crée un certificat auto-signé au lieu d'une CSR. Le certificat X.509 est un format de certificat standard.*

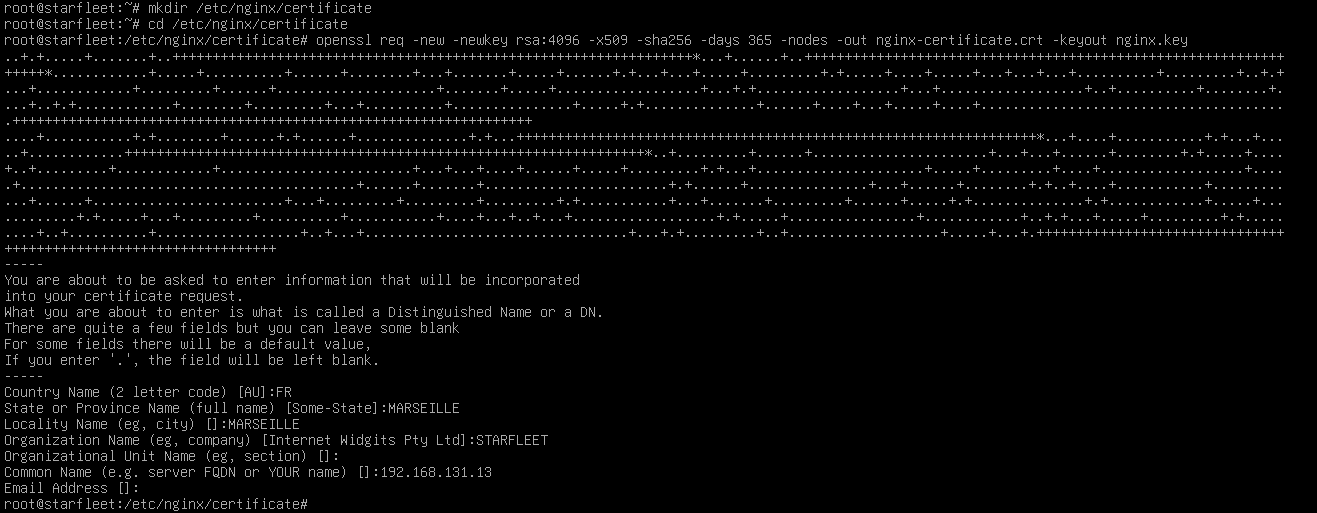
*-sha256 : Utilise l'algorithme de hachage SHA-256 pour signer le certificat.*

*-days 365 : Le certificat est valide pendant 365 jours.*

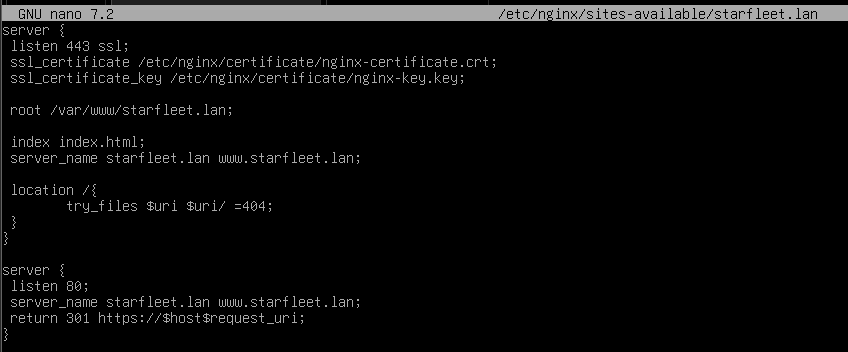
*-nodes : N'utilise pas de chiffrement pour la clé privée, ce qui signifie que vous n'aurez pas à entrer de mot de passe pour utiliser la clé.*

*-out nginx-certificate.crt : Spécifie le nom de fichier de sortie pour le certificat (dans ce cas, nginx-certificate.crt).*

*-keyout nginx-key.key : Spécifie le nom de fichier de sortie pour la clé privée (dans ce cas, nginx-key.key).*

**

*Maintenant on se rend dans sites- available dans notre fichier de configuration est on modifie de la manière suivante :*

*Cd *

Explication détaillée du fichier de configuration nginx pour un site*:*

*BLOC SSL (port 443) :*

*listen 443 ssl; :* Le serveur écoute sur le port 443 (le port par défaut pour HTTPS) et utilise SSL/TLS.

*ssl\_certificate et ssl\_certificate\_key* : Ces lignes définissent le chemin vers les fichiers de certificat SSL (le certificat public et la clé privée).

*root /var/www/starfleet.lan; :* Définit le répertoire racine des fichiers du site.

*index index.html;* : Spécifie que le fichier par défaut pour une requête à la racine est index.html.

*server\_name starfleet.lan www.starfleet.lan; :* Définit les noms de domaine pour lesquels ce bloc de serveur répondra (dans ce cas, starfleet.lan et www.starfleet.lan).

*location / :* Gère les requêtes à la racine (/) et les fichiers correspondants *:*

*try\_files $uri $uri/ =404;* : Essaie de trouver un fichier correspondant à l'URI demandé. Si le fichier ou le répertoire n'existe pas, renvoie une erreur 404.

*BLOC HTTP (port 80) :*

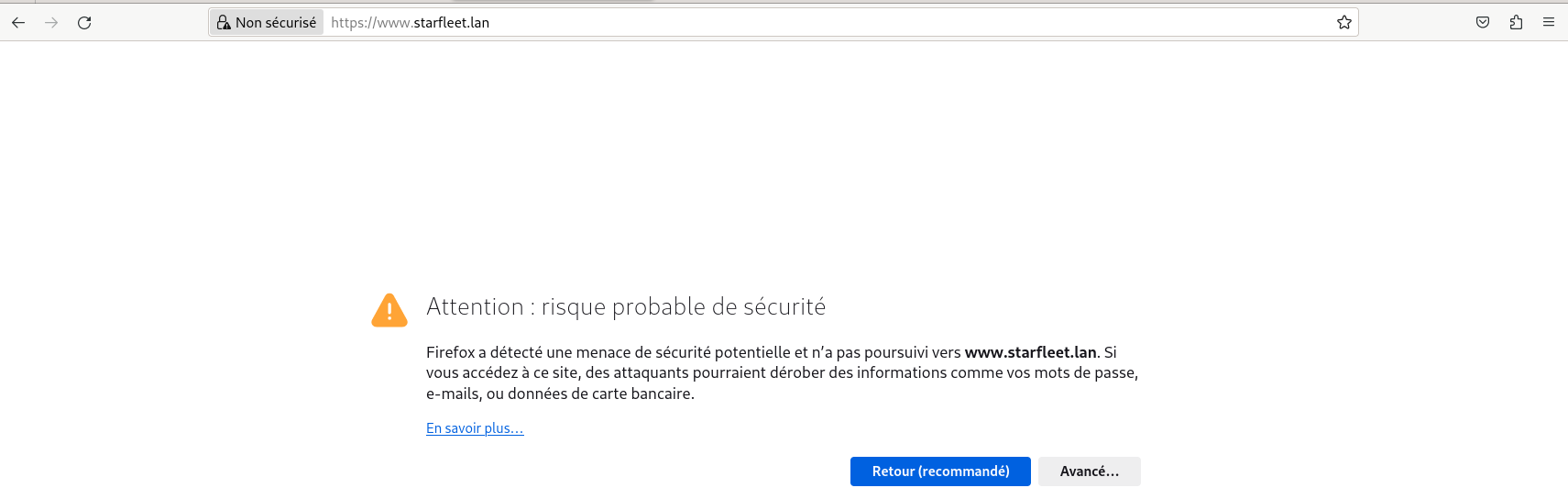
*listen 80; :* Le serveur écoute sur le port 80 (le port par défaut pour HTTP).

*server\_name starfleet.lan www.starfleet.lan; :* Comme pour le bloc SSL, définit les noms de domaine.

*return 301 https://$host$request\_uri; :* Redirige toutes les requêtes HTTP vers HTTPS (redirection permanente 301).

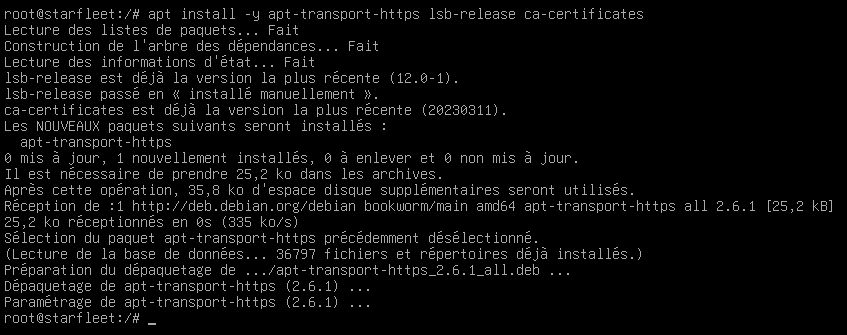
*Commande pour vérifier la configuration :*

Vérification sur vm client de http, https et redirection http en https ok

**

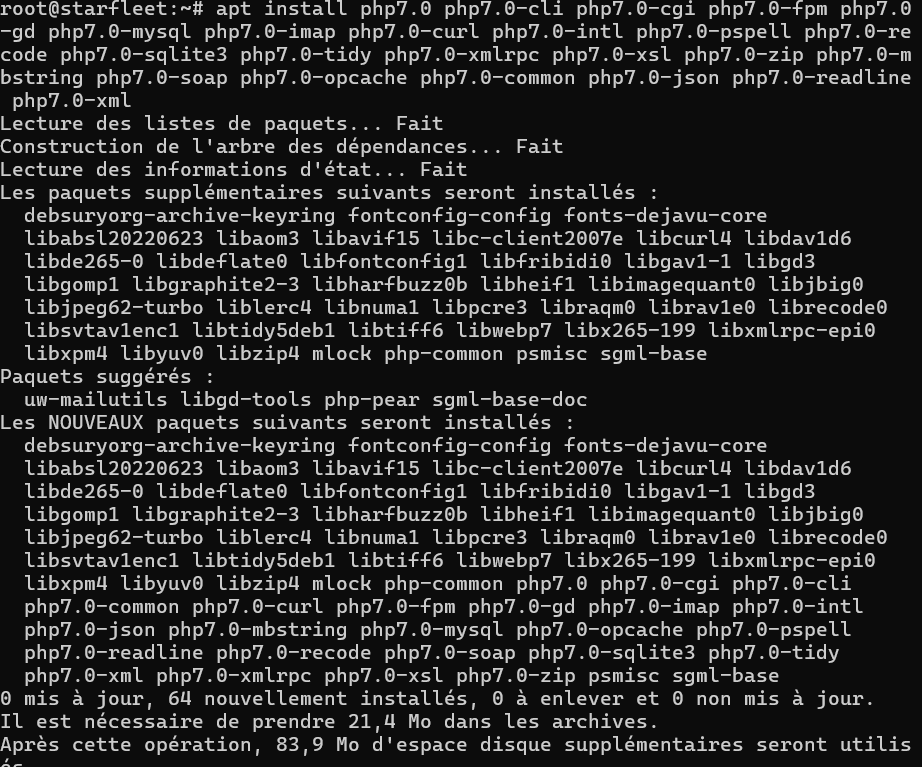
1. **Installer plusieurs versions de php sur debian**

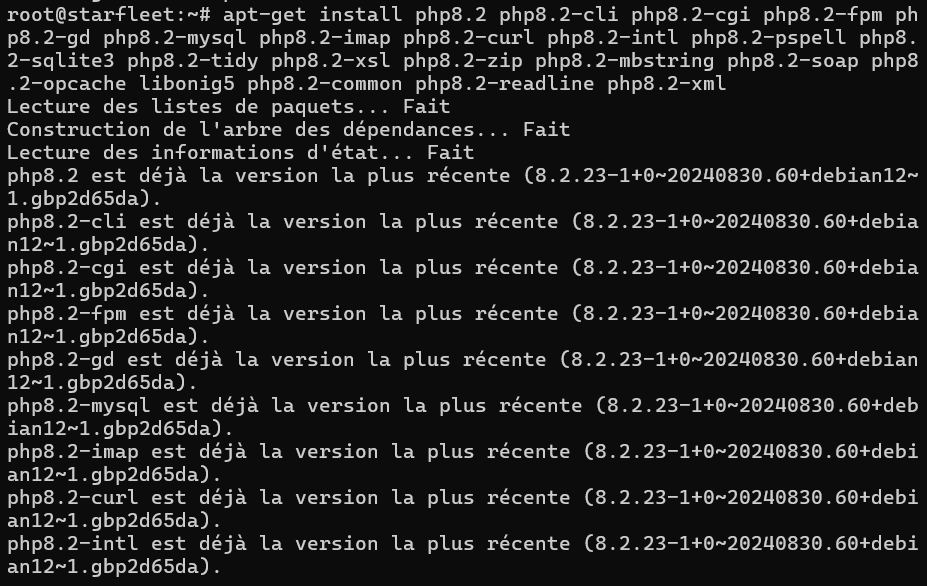
Nous utiliserons le dépôt SURY, c’est un dépôt supplémentaire qui fournit les paquets PHP pour l’installation avec APT. PHP-FPM peut être utilisé avec les serveurs apache et nginx tandis que FASTCGI n’est dispo que pour apache.

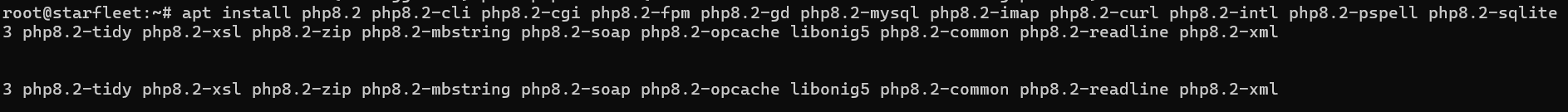
*Pour commencer on ajoute le dépôt SURY avec la commande qui suit  
*

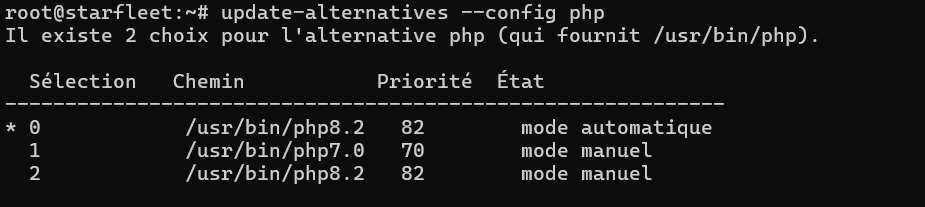
*Maintenant on ajoute la clé du référentiel SURY :  
*

*On ajoute le référentiel SURY lui-même :  
*

*Après on install php 7.0  
*

*Pour la version php8.2 on peut voir que des paquets été déjà installé de base sur debian 12  
*

*On installe également php8  
*

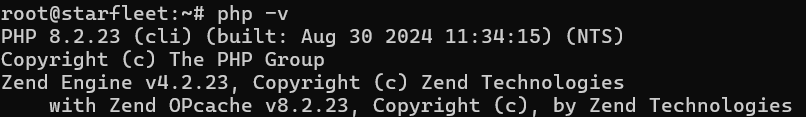
*Maintenant on vérifie les versions et on laisse la sélection par défault   
*

*A quoi sert la commande « update-alternatives --config php » ?*

Elle permet de gérer les différentes versions de php installées sur notre système celle à utiliser par défaut.

Mode automatique : Le système choisit la version avec la priorité la plus élevée.

Mode manuel : Vous sélectionnez manuellement la version de PHP que vous souhaitez utiliser.

*Maintenant on vérifie la version sélectionnée :  
*

*Maintenant on va créer un nouveau répertoire dans /var/www que l’on nommera pour une question de lisibilité www7-starfleet dans lequel se trouvera notre index.html ainsi que le phpinfo.php*

**

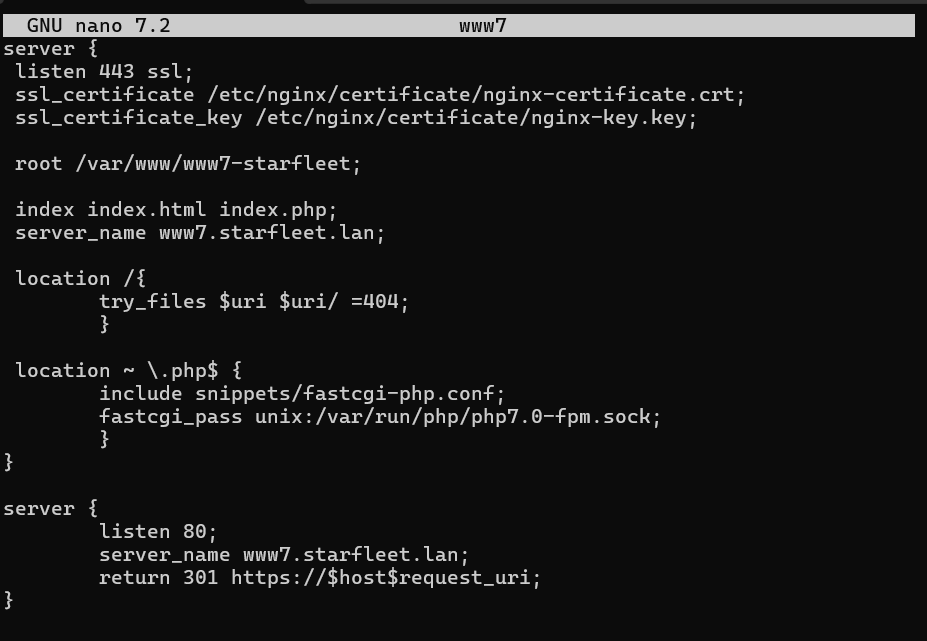
*A quoi sert le fichier phpinfo ?*

Phpinfo.php sert généralement à vérifier la configuration ainsi que les variables prédéfinies. La fonction phpinfo() est un bon outil de débogage, car il affiche le contenu de toutes les variables EGPCS (Environnement, GET, POST, Cookie, Serveur).

**



*On va déclarer notre site www7 dans nginx via le chemin* ***/etc/nginx/sites-available comme on la fait pour notre fichier starfleet.lan correspondant à www.***

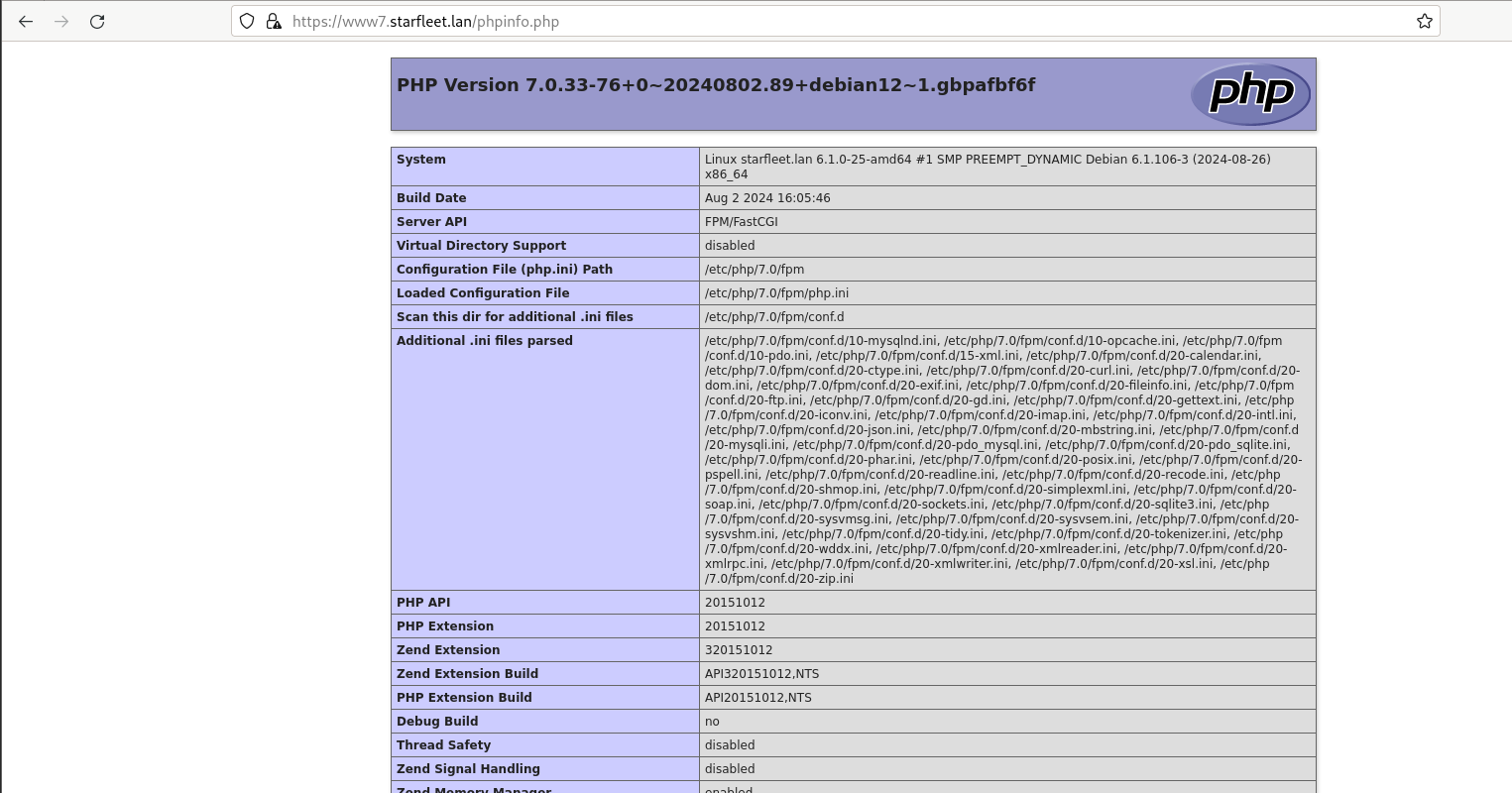
**

*Faire le lien symbolique*

**

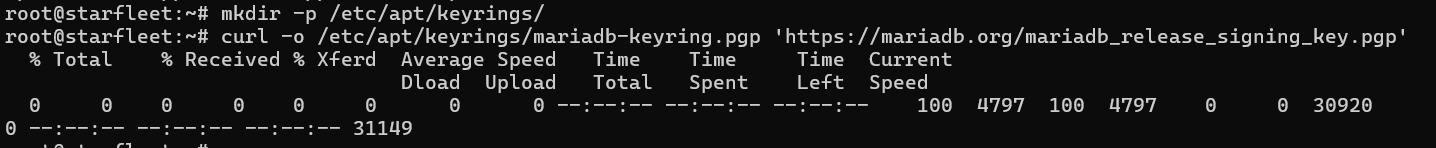
* On pense bien à redémarrer nos services, et sur la VM client on vide la cache du navigateur.*

*Vérification sur vmCLient*

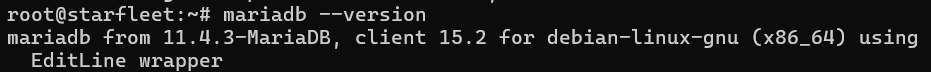
**

1. **INSTALLATION DE LA DERNIERE VERSION DE MARIA DB**

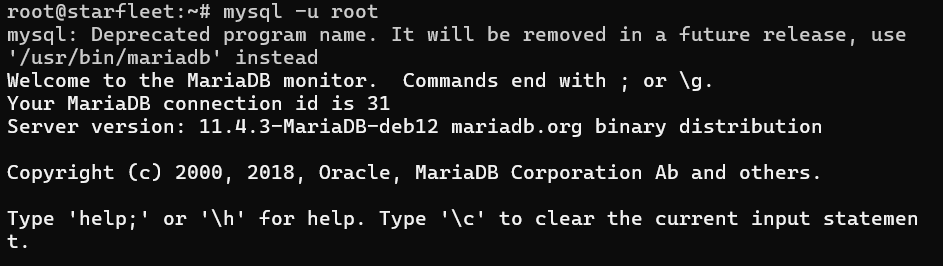
On commence par se rendre sur le site de mariaDB pour récupérer le fichier dépôt de la dernière version. J’ai choisi la version 11.4

****

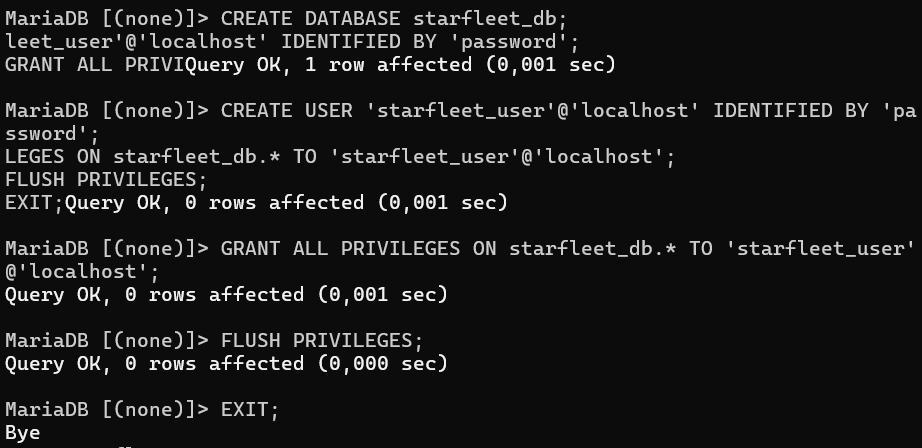
****

****

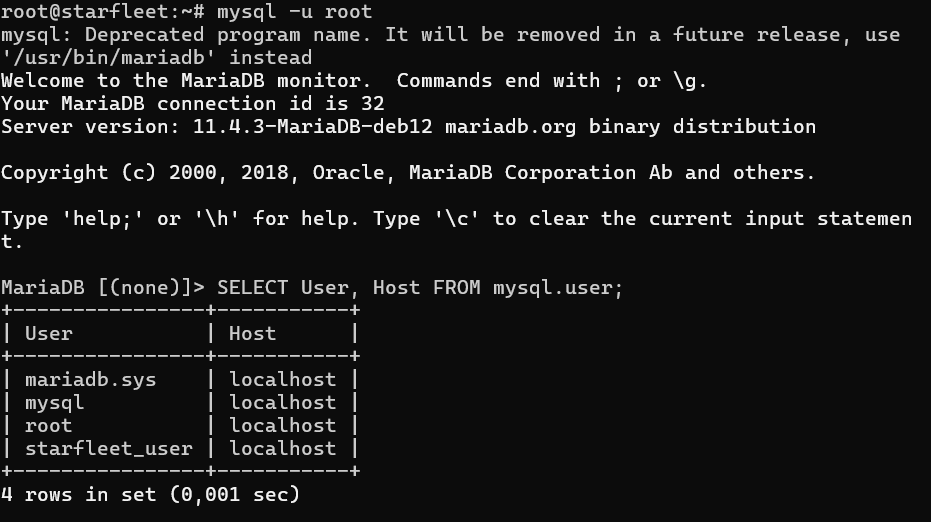
Apres on rentre dans mariadb avec la commande suivante

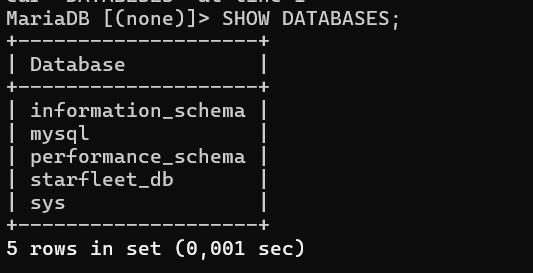
****

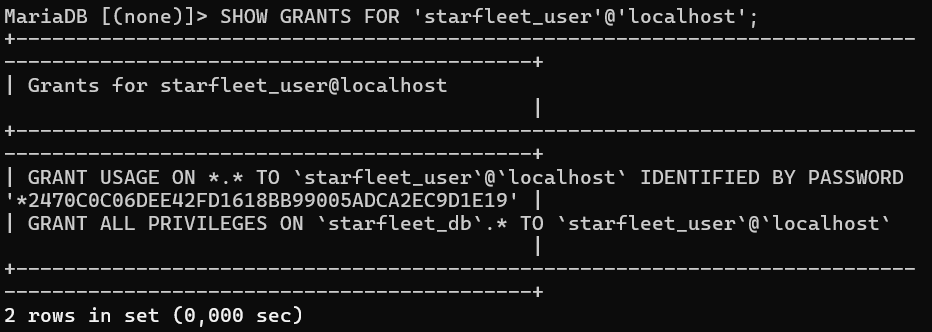
*On crée la base de donné strarfleet\_db*

****

**Starfleet\_user = user ; mdp =password**

****

****

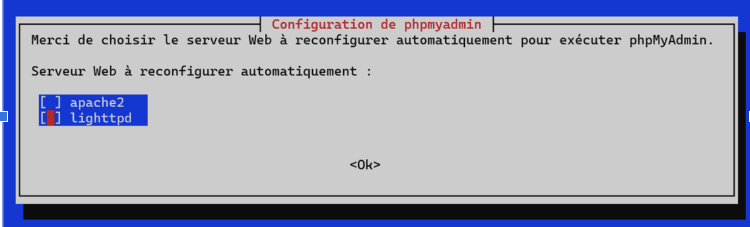
****

1. **INSTALLATION PhpMYAdmin**

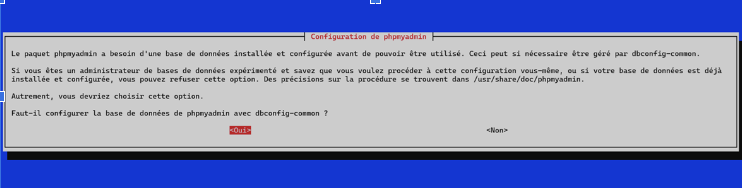
On commence par mettre à jour nos paquets et on télécharge le paquet phpadmin

apt install phpmyadmin

*Là on ne sélectionne rien, on va direct sur ok*



la on met oui pour reconfigurer la base de données de php

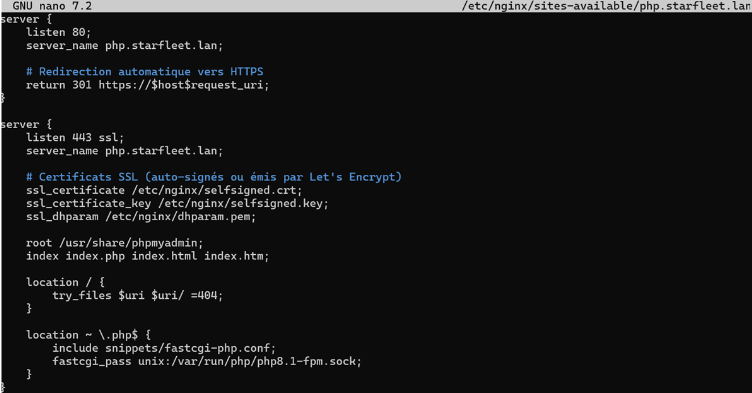


Ensuite, nous avons dû créer des liens symboliques pour que Nginx puisse

accéder aux fichiers de phpMyAdmin, car phpMyAdmin est installé par défaut dans /usr/share/phpmyadmin, et Nginx sert les fichiers à partir de /var/www :

**ln -s /usr/share/phpmyadmin /var/www/phpmyadmin**

On édite le fichier /etc/nginx/sites-available/php.starfleet.lan



Une fois le fichier créé, nous avons activé ce site en créant un lien symbolique dans le dossier sites-enabled :

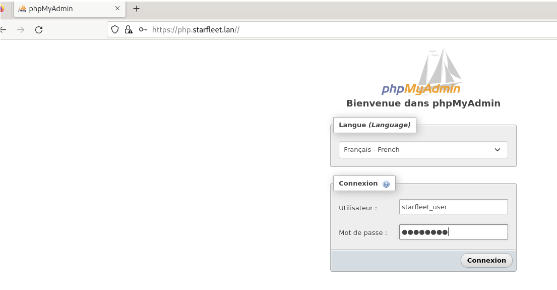
ln -s /etc/nginx/sites-available/php.starfleet.lan /etc/nginx/sites-enabled/

apt install php8.1 php8.1-fpm

apt install php8.1-mbstring php8.1-xml php8.1-zip php8.1-mysql

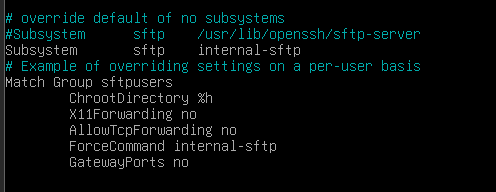
nginx -t

restart nginx et tester sur la vm client



1. **configuration sftp**

On commence par crée notre groupe d’utilisateur qui sera dédié a sftp  


*Puis on se rend dans le fichier de configuration de ssh pour ajouter les lignes suivantes que je vais vous détailler en dessous.*   




On crée les règles pour notre groupe utilisateurs spécifique (groupe que l’on a créé en 1er)

Match Group sftpusers : Cette ligne spécifie que les directives qui suivent s'appliquent uniquement aux utilisateurs faisant partie du groupe sftpusers. Si on veut restreindre certains utilisateurs à SFTP uniquement, tu dois les ajouter à ce groupe.

**ChrootDirectory %h** : Cette directive restreint l'utilisateur à son propre répertoire personnel (%h représente le chemin du répertoire home de l'utilisateur). En activant cela, l'utilisateur ne pourra pas accéder aux fichiers en dehors de son propre dossier.

ForceCommand internal-sftp : Cela force l'utilisateur à n'utiliser que la commande SFTP, empêchant tout accès au shell. L'utilisateur ne pourra que transférer des fichiers via SFTP et ne pourra pas exécuter d'autres commandes sur le serveur.

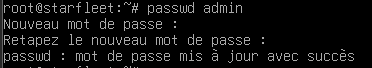
X11Forwarding no : Désactive le transfert de l'interface graphique (X11) pour ces utilisateurs, ce qui est souvent inutile pour SFTP.

AllowTcpForwarding no et GatewayPorts no : Empêchent l'utilisateur de faire du "port forwarding" via SSH, ce qui peut être un risque de sécurité

 : Par défaut, OpenSSH utilise /usr/libexec/openssh/sftp-server pour gérer les connexions SFTP. Cette ligne est maintenant commentée.

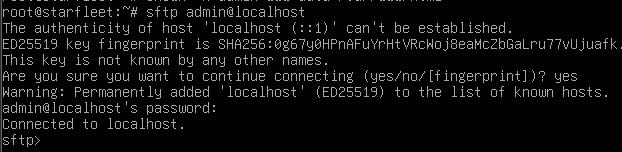
Nouvelle ligne  : En utilisant internal-sftp, on configure OpenSSH pour utiliser **une version interne de SFTP**, ce qui évite de lancer un processus externe. Cela permet d'améliorer la sécurité et les performances.

*Apres on ajoute notre utilisateur a notre groupe spécifique a sftp*





On redemarre ssh et on fait la commande suivante pour se connecter au service sftp



**Pourquoi choisir sftp plutôt que ftps ?**

**SFTP** est généralement considéré comme plus récent et plus sûr que **FTPS**. Il s'agit d'une technologie moderne, bien intégrée dans les systèmes actuels, qui offre un haut niveau de sécurité tout en étant plus simple à configurer. Grâce à son utilisation de **SSH**, la sécurité est assurée sans qu'il soit nécessaire de mettre en place des configurations complexes.

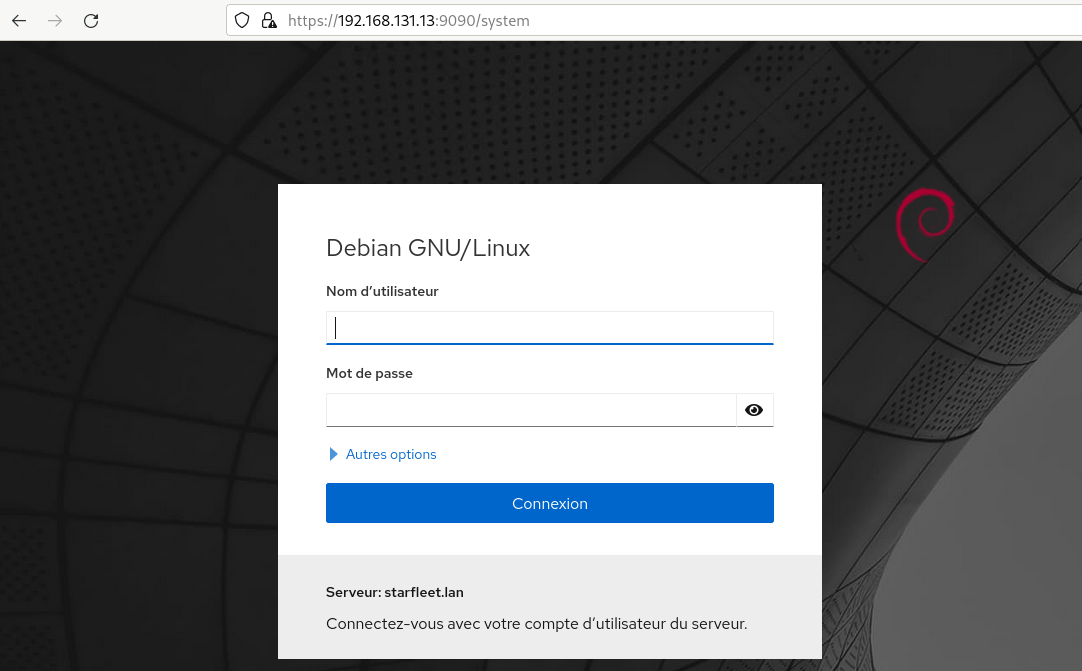
De son côté, bien que **FTPS** soit également sécurisé, il est souvent perçu comme plus compliqué à gérer et moins flexible, notamment dans des environnements où la configuration des pare-feux est stricte. Il reste néanmoins utilisé pour des raisons de compatibilité avec certains systèmes plus anciens.

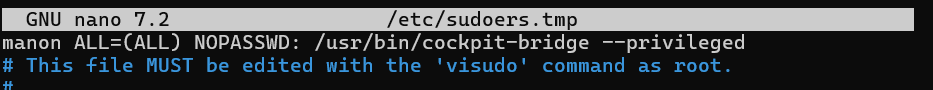
1. **Installer & configurer cockpit**

On commence par mettre à jour les paquets et on télécharge le paquet pour cockpit  

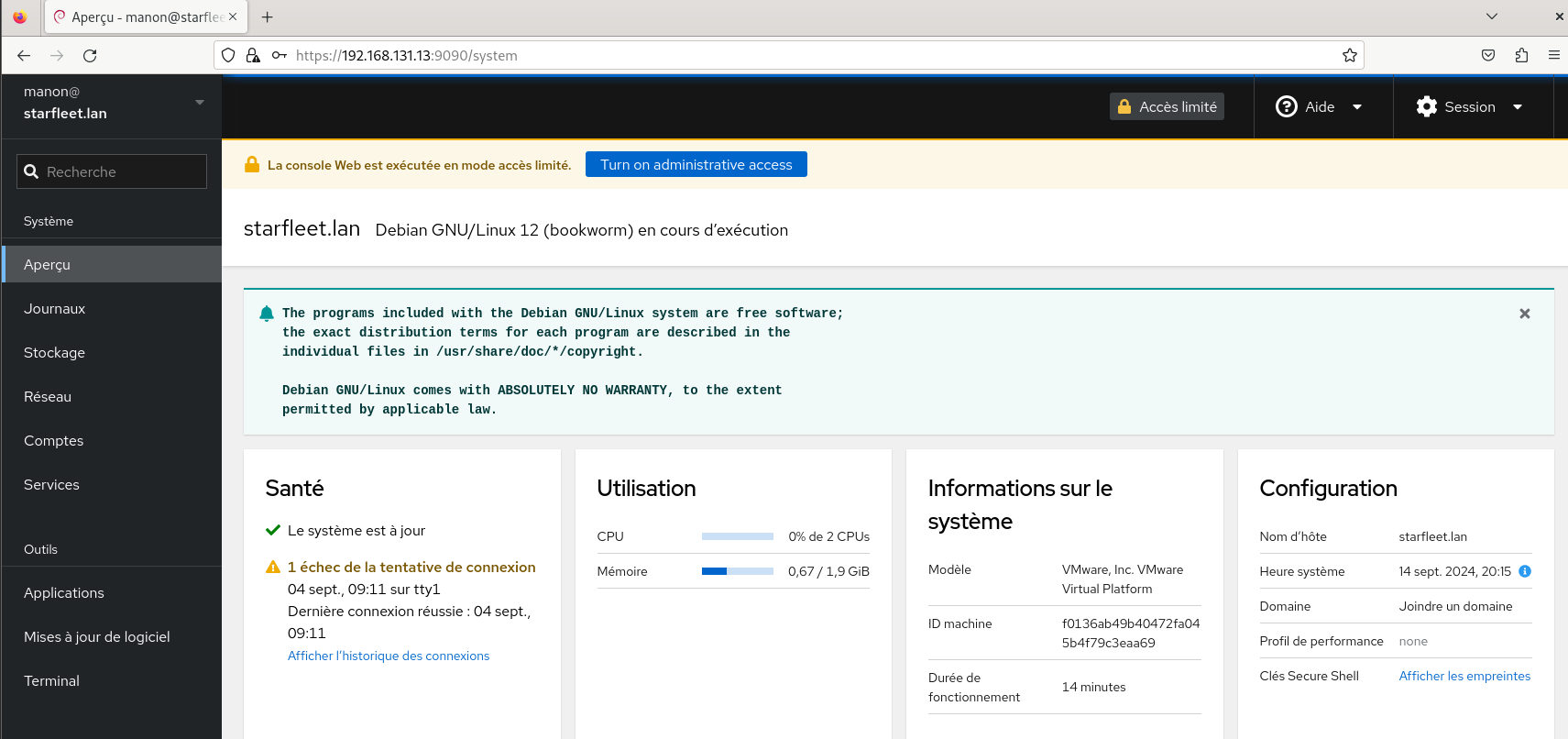

Ensuite, on active le démarrage automatique de cockpit  


On se rend sur notre vmCLient on tape https:/192.168.131.13 :9090

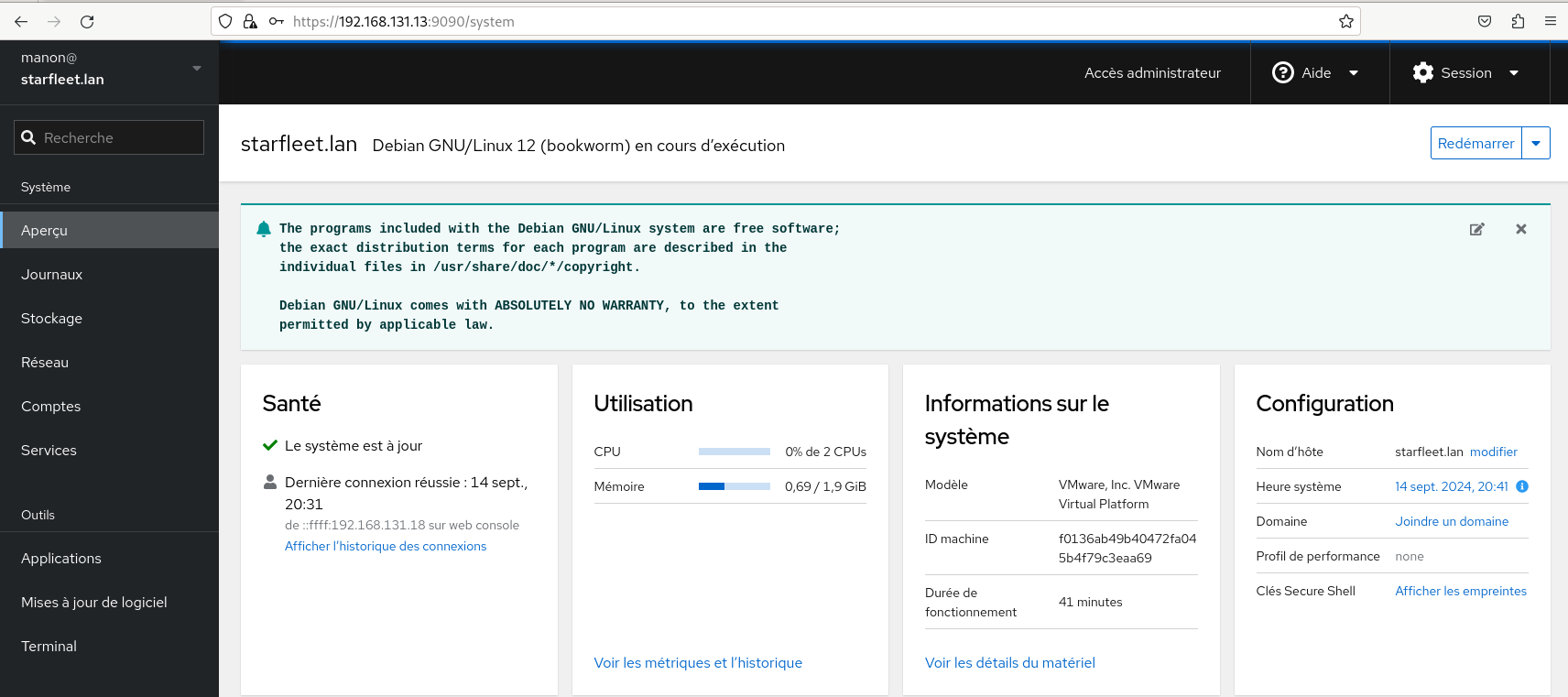


Pour avoir l’accès en tant qu’administrateur il suffit de se rendre sur visudo est ajouter cette ligne 

On redémarre le system cockpit et on se reconnecte sur la vmclient







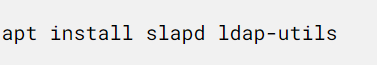
Pour ouvrir cockpit avec l’adresse suivante, il suffit de se rendre notre fichier etc/hosts

Et on ajoute tout simplement notre adresse ip de notre serveur et le nom que l’on souhaite



1. Installation ldap

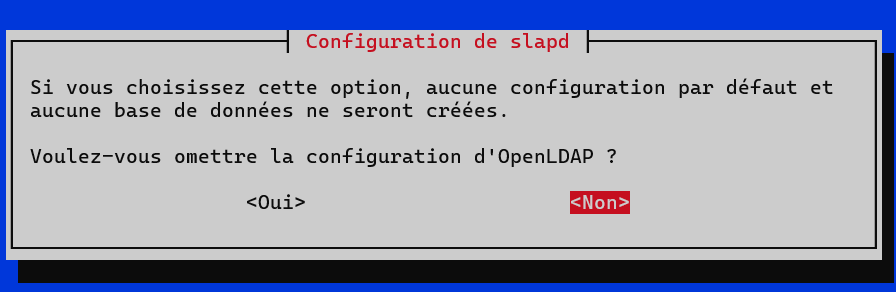
On commence par installer les paquets suivants avec les commandes suivante

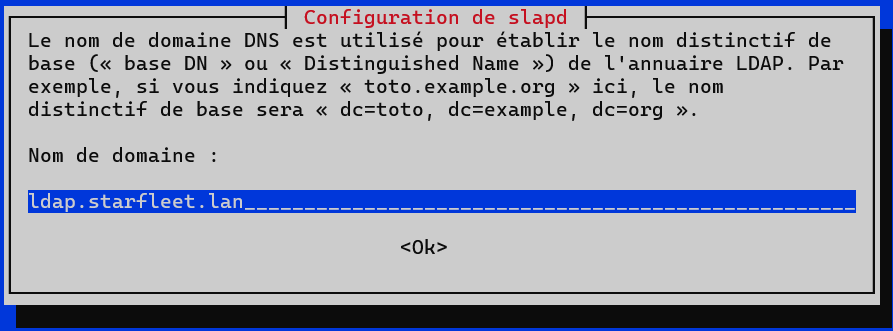


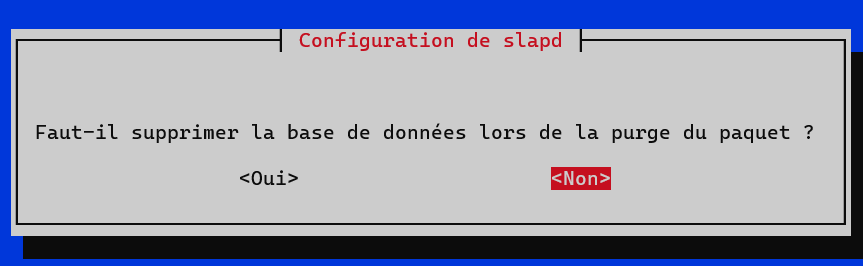


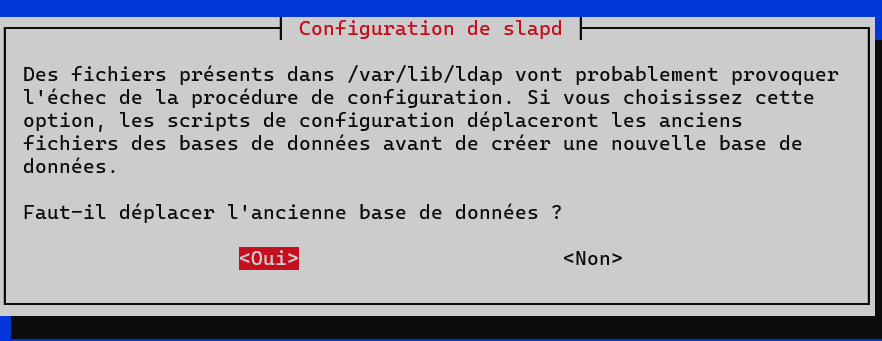
Apres on se rend dans la configuration de slapd avec la commande suivante

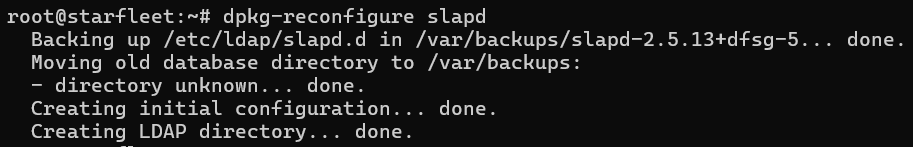
dpkg-reconfigure slapd



****

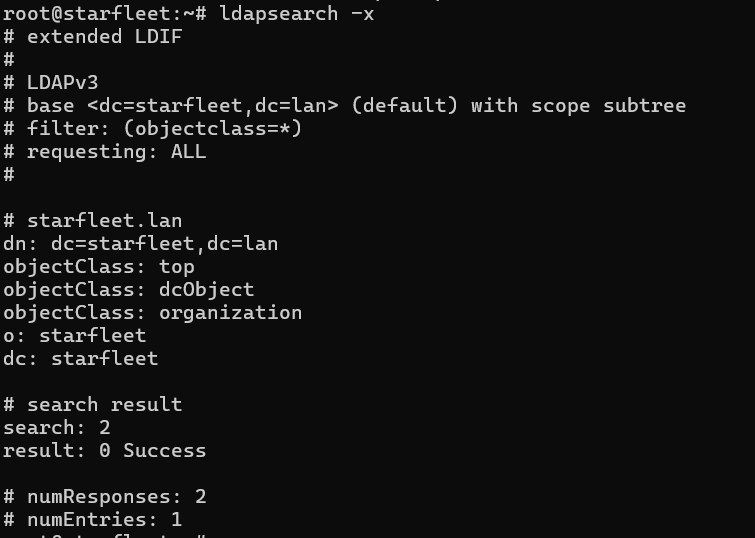
****

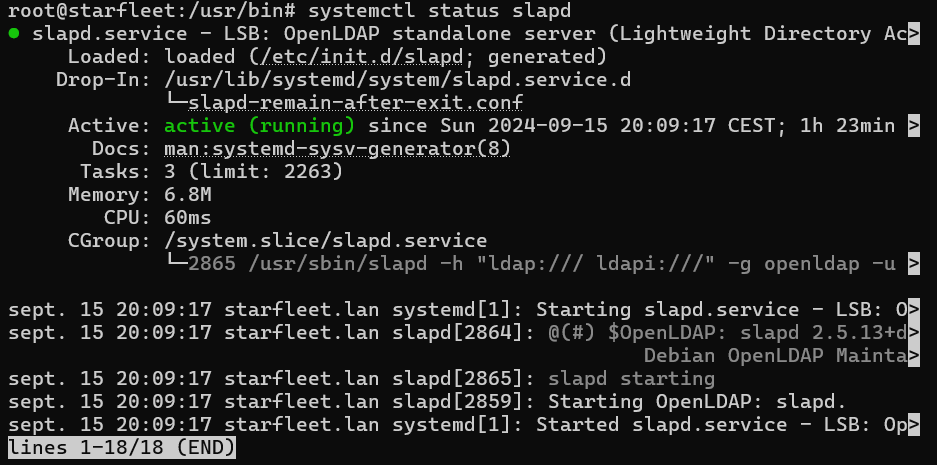
****

****

Apres on se rend dans le fichier de conf du ldap



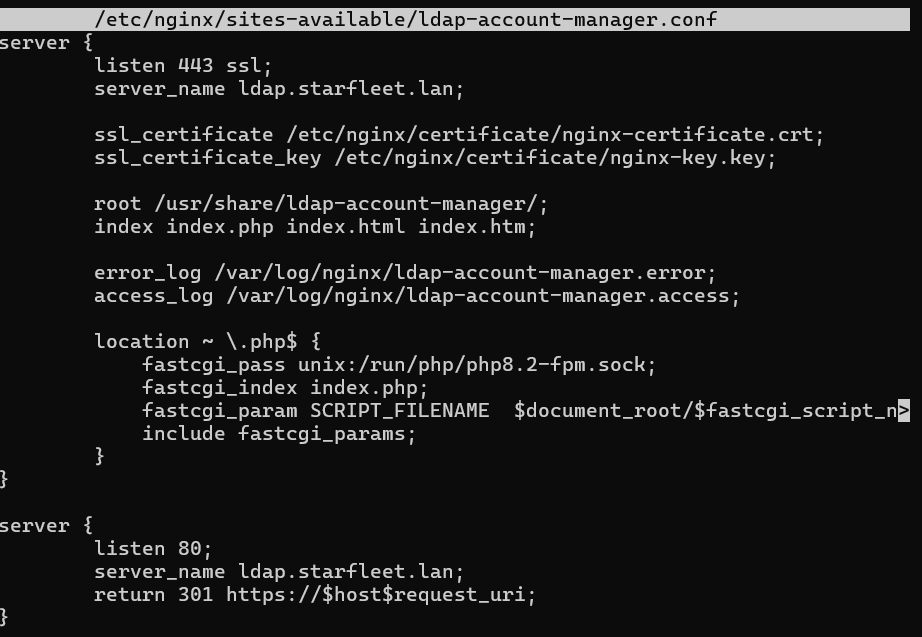
****

****

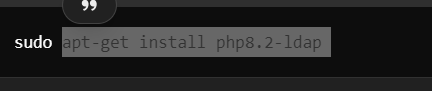
On installe ldap-account-manager qui permet d’avoir le graphique sur notre page web ldap.starfleet.lan

****

Maintenant on déclare notre site ldap sur nginx :

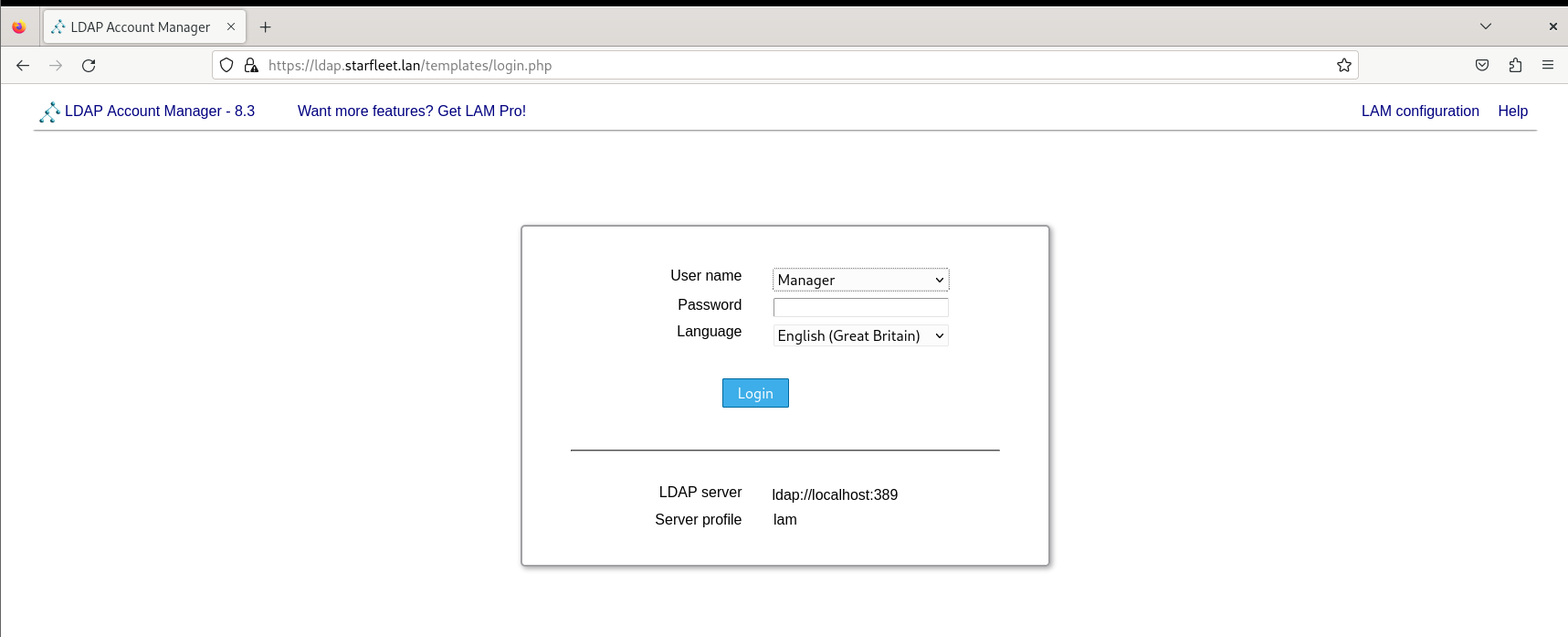
****

On pense à faire le lien symbolique pour notre ldap

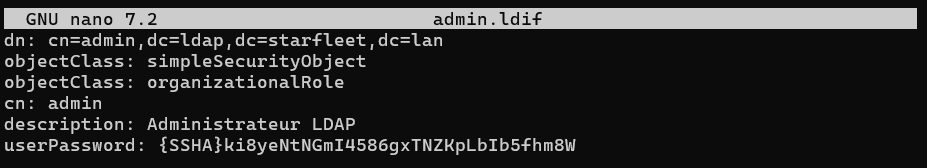
****

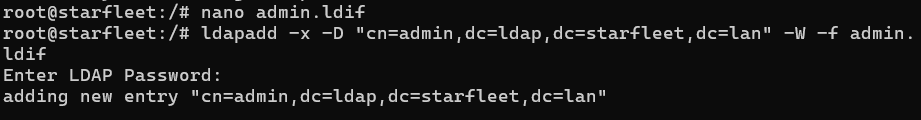
****

*On redémarre les services nginx, ldap,bind9 histoire d’etrer sur et on se rend sur notre vm client pour tester*

****

Maintenant on va créer notre utilisateur « admin » pour cela on crée notre fichier admin.ldif que l’on met n’importe où dans notre système pour des raisons de lisibilité j’ai choisi de le mettre simplement a la racine

*On configure le fichier de cette manière*

*Puis on ajoute l’utilisateur à ldap*

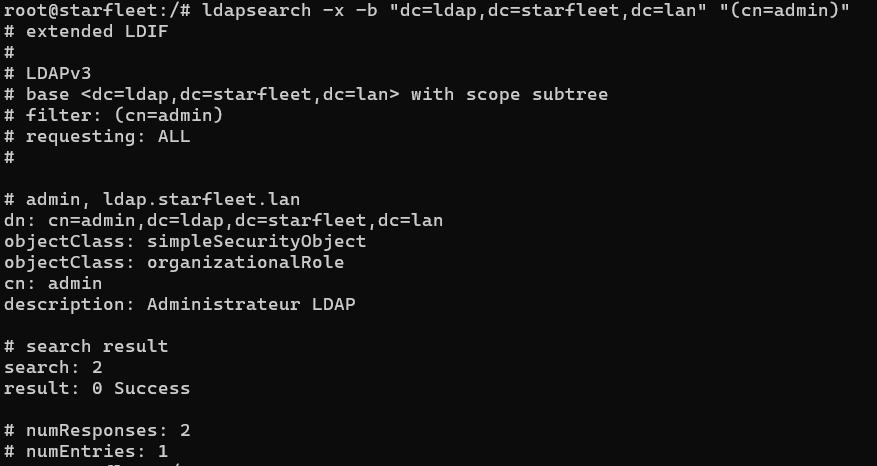
*Explication de la commande pour ajouter l’utilisateur a ldap :*

-x : Utilise l'authentification simple (pas SASL).

-D : Indique le DN (Distinguished Name) de l'utilisateur qui fait l'ajout (ici, cn=admin,dc=ldap,dc=starfleet,dc=lan).

-W : Demande le mot de passe pour cet utilisateur.

-f : Spécifie le fichier LDIF à utiliser (ici, admin.ldif).

Ensuite, on vérifie que l’utilisateur soit bien ajouter a notre ldap avec la commande suivante :  


*apt install build-essential libpcre3-dev zlib1g-dev libldap2-dev libssl-dev*

*git clone* [*https://github.com/kvspb/nginx-auth-ldap.git*](https://github.com/kvspb/nginx-auth-ldap.git)

***wget https://nginx.org/download/nginx-1.26.2.tar.gz***

***tar zxvf nginx-1.26.2.tar.gz***

***cd nginx-1.26.2***

***Configurez et compilez Nginx avec le module LDAP :***

***./configure --with-http\_ssl\_module --addmodule=/usr/local/src/nginx-auth-ldap***

***Make***

***make install***