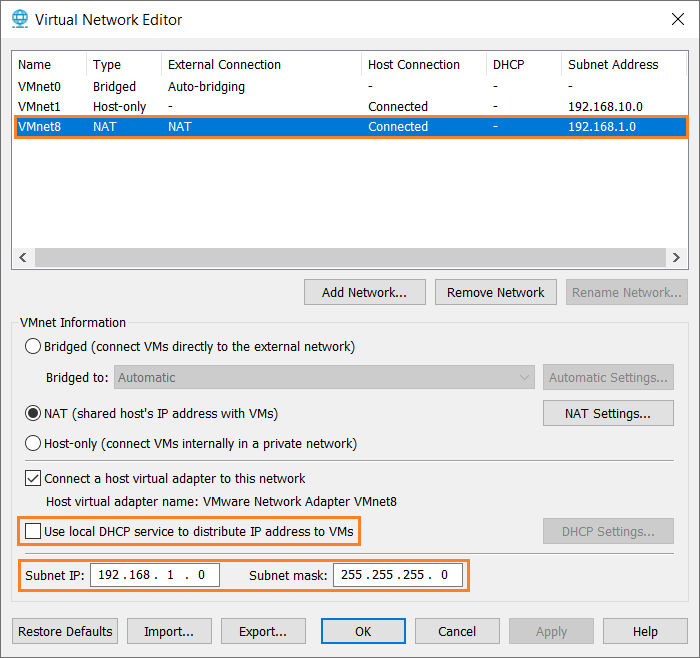
**HOLODECK PROJECT**

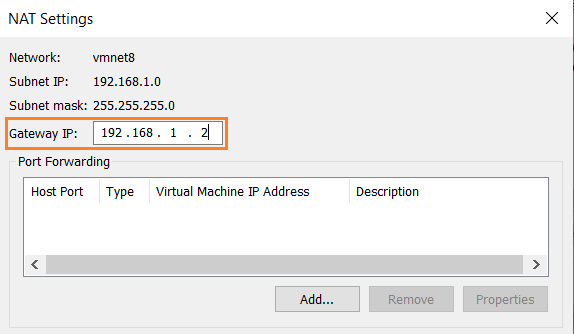
# CONFIGURATION DES VM’S

## 1 – Configuration du réseau

On se rends dans **Virtual Network Editor** =>

Ensuite, il faudra sélectionner notre **NAT (VMnet8),** désactiver le **DHCP local** (VMware), définir le réseau (**classe B : 192.168.1.0**) ainsi que le masque de sous-réseau (**255.255.255.0**)



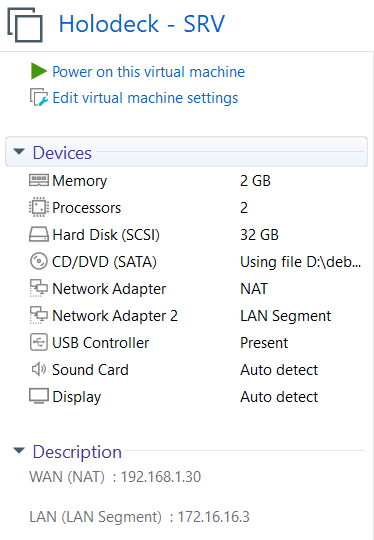


Ensuite, nous allons vérifier et configurer la **passerelle (Gateway).** Pour cela, rendez-vous dans les paramètres NAT (**NAT Setting**), puis configurez la passerelle ou laissez-la par défaut (par défaut, VMware utilise toujours l'adresse **.2** comme Gateway, ici **192.168.1.2**).

Une fois les modifications enregistrées, nous pourrons passer à la création des VM

## 2 – Mise en place des VM

# Configuration des VM

**HOLODECK - SRV**

**Pour le serveur**, nous allons installer Debian en version minimale, sans interface graphique. Avec les spécifications suivantes :

- **RAM** : 2 Go

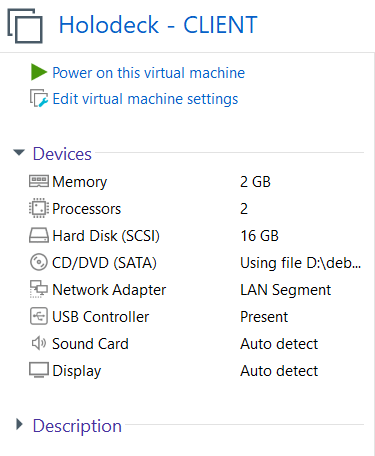
- **Processeurs** : 2 vCPU

- **Disque dur** : 32 Go

- **Cartes réseaux** : 2 interfaces – une pour le WAN et une pour le LAN

Cela permettra une configuration réseau flexible, avec une interface reliée à l'extérieur (**WAN = NAT**) et une autre pour le réseau interne (**LAN = LAN Segment**).

**HOLODECK - CLIENT**



**Pour la machine cliente**, nous allons installer Debian avec une interface graphique (GUI). Avec les spécifications suivantes :

- **RAM** : 2 Go

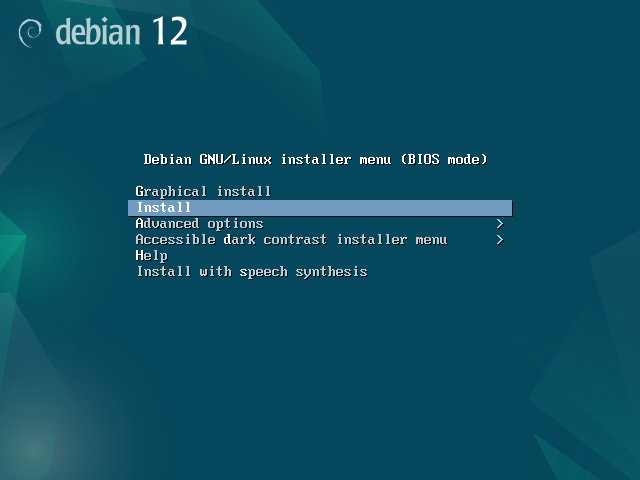
- **Processeurs** : 2 vCPU

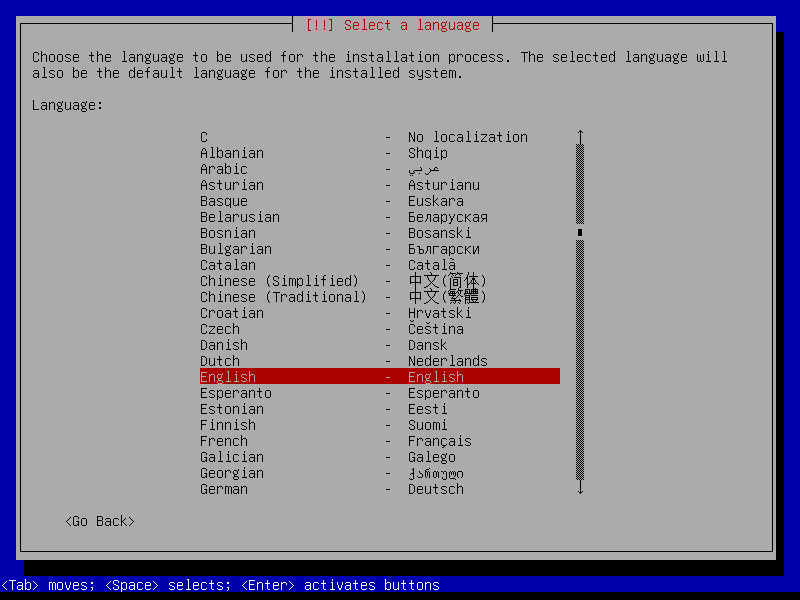
- **Disque dur** : 16 Go

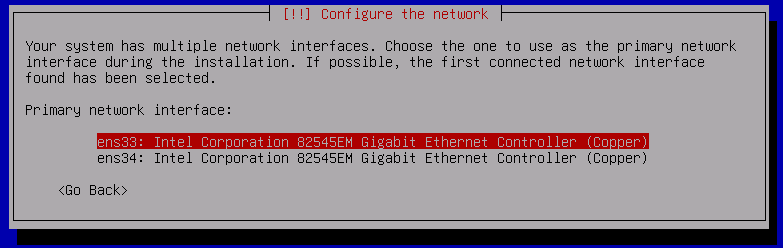
- **Connexion réseau** : reliée au réseau « LAN » de la VM serveur

Cette configuration permettra au client de communiquer avec le serveur via le réseau LAN (**LAN Segment**) et d'utiliser le navigateur pour les interactions web.

# Installation de Debian

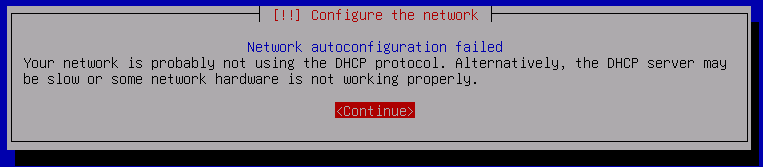




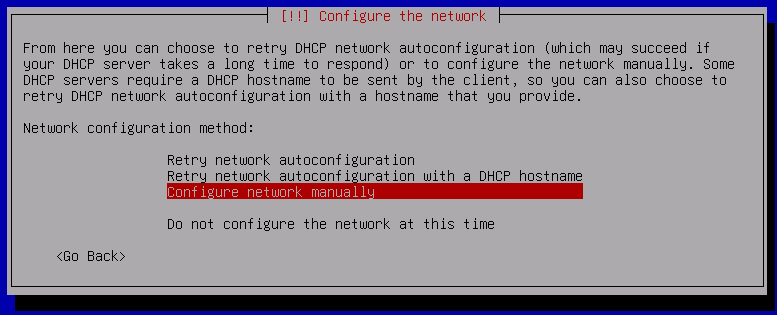


L'interface NAT est toujours celle avec l'adresse la plus petite.

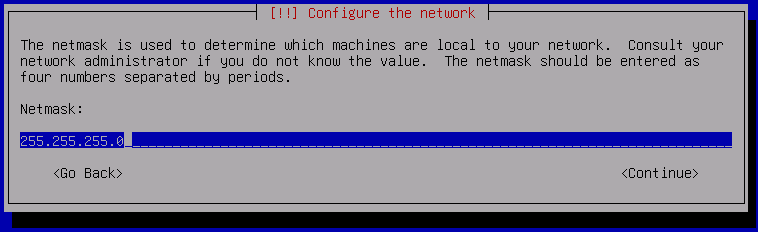
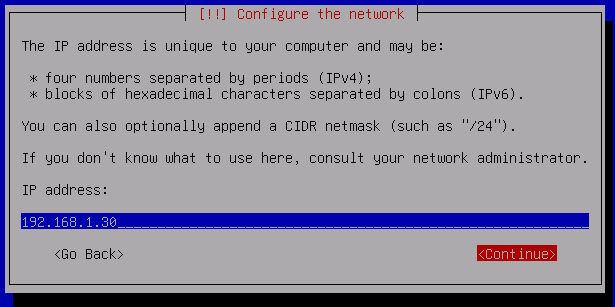
Donc, entre **ens33** et **ens34**, l'interface **ens33** sera utilisée comme NAT.

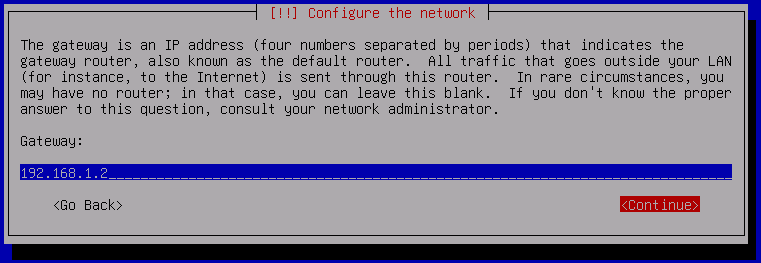
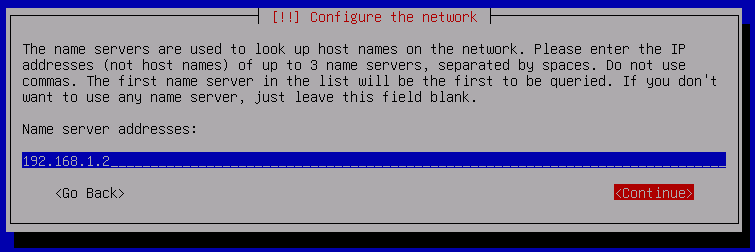


Ensuite, vous verrez un message indiquant qu'aucun serveur DHCP n'est actif sur le réseau, ce qui est normal puisque nous avons désactivé le **DHCP** dans les paramètres de VMware (voir précédemment).



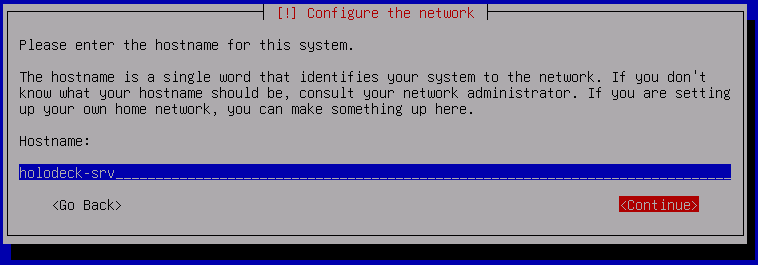
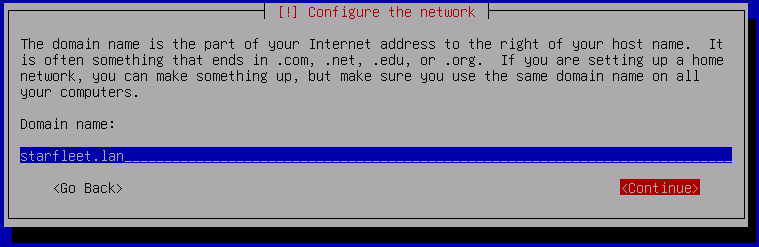
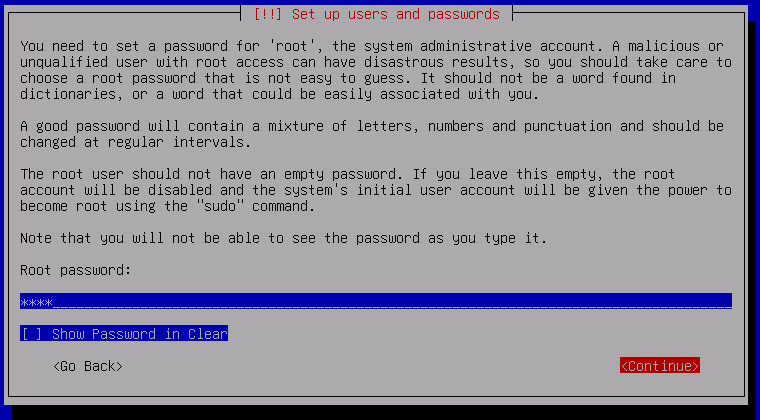
Pour résoudre cela, nous allons configurer manuellement le serveur avec les bonnes **adresses IP**, **passerelle** et **DNS** afin de lui permettre d'accéder à Internet.

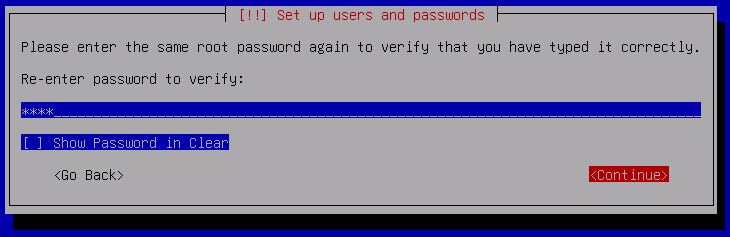
** WAN IP (192.168.1.30) NETMASK : 255.255.255.0**

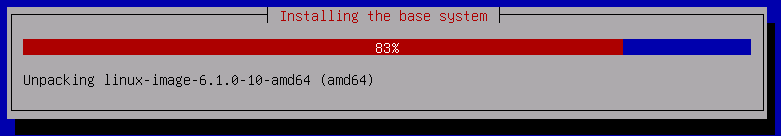
**** **Passerelle (Gateway) : 192.168.1.2 Name Server : 192.168.1.2**

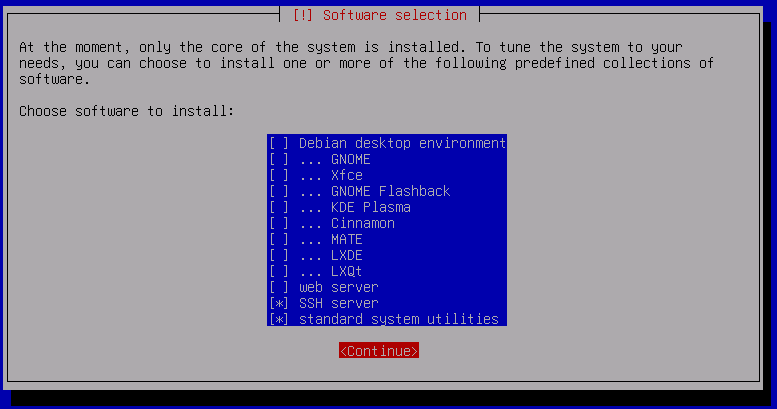
Normalement, si tout est correctement configuré, le serveur devrait avoir accès à Internet. Nous pourrons alors poursuivre l'installation sans problème.

**Hostname: holodesk-srv Domaine: starfleet.lan**

** MDP ROOT**

 **CONFIMER LE MDP ROOT**



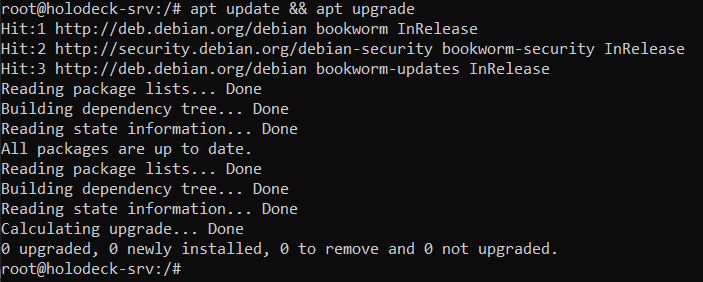


Ici, nous désélectionnons **Debian desktop environment** (Environnement de bureau) ainsi que **GNOME** afin de désactiver l'interface graphique (GUI) et d'installer une version minimale de Debian sans interface graphique



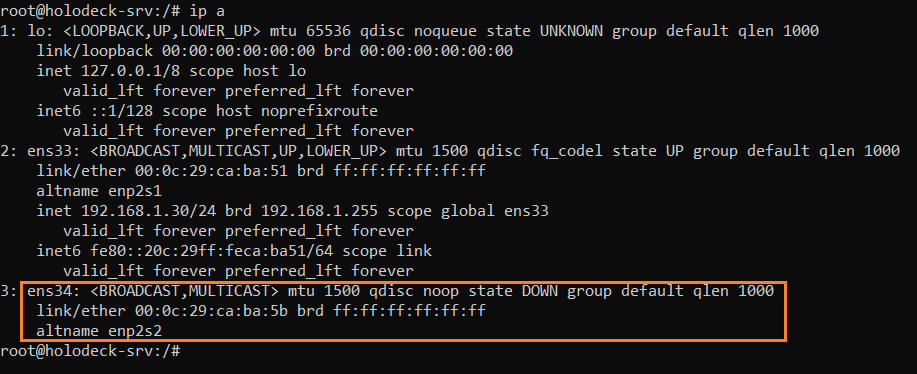
L'installation terminée, nous pouvons désormais procéder à l'installation et au déploiement des différents services sur notre serveur.

# Configuration de l’interface pour le réseau LAN

Tout d'abord, avant de commencer les installations, nous allons exécuter les commandes suivantes pour vérifier et installer les mises à jour disponibles :

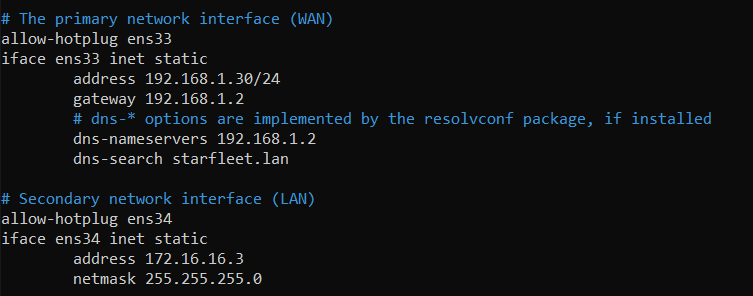
**apt update && apt upgrade**

Cela permettra de s'assurer que le système est à jour avec les derniers correctifs de sécurité et les mises à jour de paquets avant de poursuivre les configurations et installations.

Comme on peut le voir ici, en exécutant la commande : **ip a**, notre deuxième interface réseau **ens34 (LAN)** n'est pas encore configurée et ne possède pas d'adresse IP assignée.

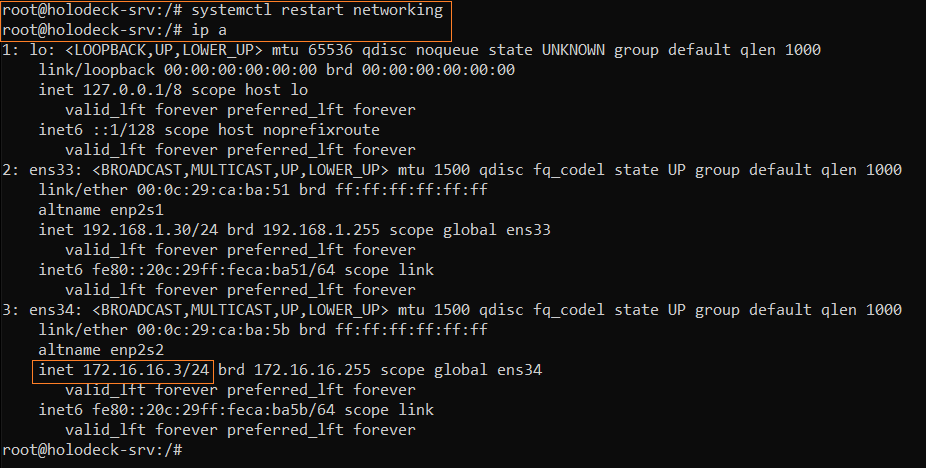
Nous allons donc procéder à la configuration de cette interface en lui attribuant une adresse IP statique.

Ensuite, nous pourrons déployer un serveur DHCP sur cette interface

Pour cela, on va éditer le fichier de configuration réseau :

**nano /etc/network/interfaces**

et ajouter notre configuration



Maintenant, nous allons redémarrer le service réseau pour appliquer les changements avec la commande : **systemctl restart networking**

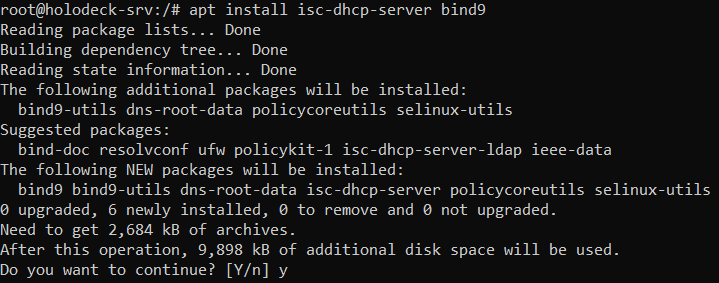
**ip a**

Et comme on peut le constater, la configuration a bien été appliquée, et notre interface possède désormais l'adresse IP **172.16.16.3**

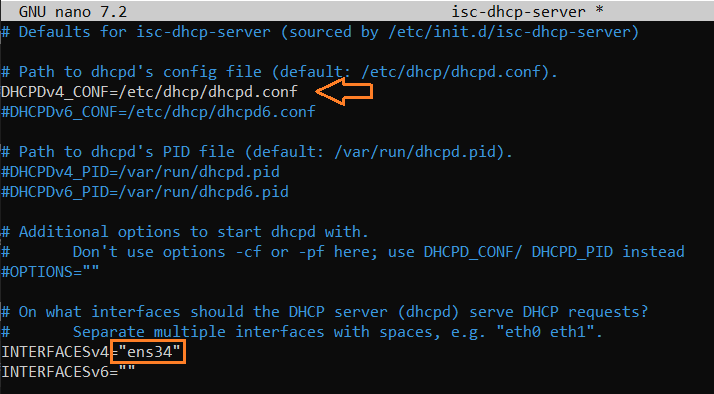
# **Installation du serveur DHCP et DNS**

On va maintenant notre installer le DHCP (**isc-dhcp-server**) et le DNS (**bind9**) avec la commande :

**apt install isc-dhcp-server bind9**



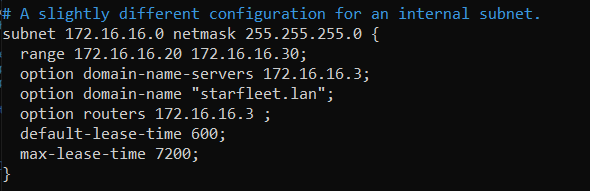
Editer le fichier **/etc/default/isc-dhcp-server**

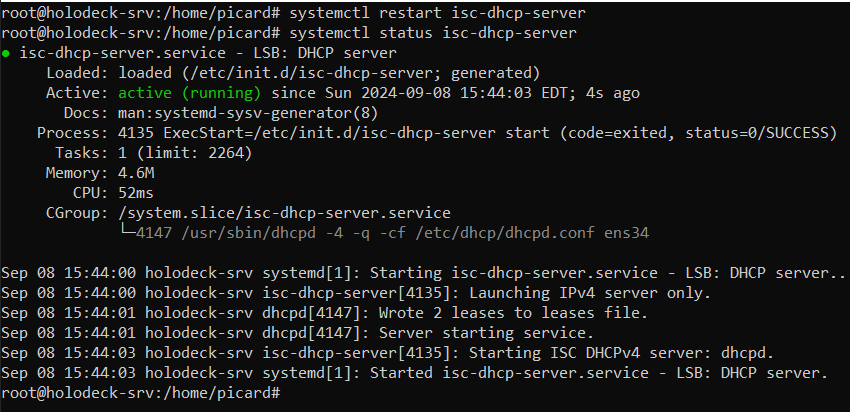


Décommenter la ligne **DHCPDv4\_CONF** et indiquez l'interface réseau sur laquelle écoute le serveur **INTERFACESv4 (ens34)**.

Enregistrez les modifications.

Editer le fichier **etc/dhcp/dhcpd.conf** , puis configurez votre réseau. Enregistrez et quittez.

****

Ensuite, nous redémarrons le service DHCP pour appliquer les modifications :

**systemctl restart isc-dhcp-server**

Puis, vérifions le statut du service pour nous assurer qu'il fonctionne correctement :

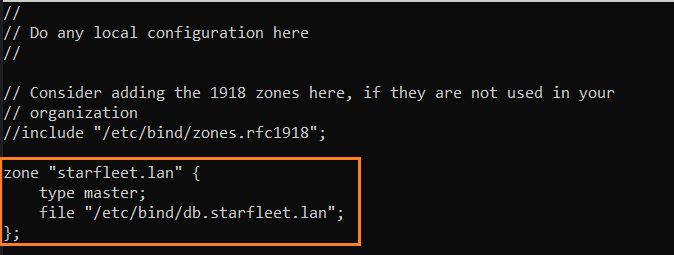
**systemctl status isc-dhcp-server**

Si aucune erreur n'apparaît, cela signifie que tout est bien configuré et fonctionne correctement.

## **Configuration DNS Bind9 : Créer une zone DNS**

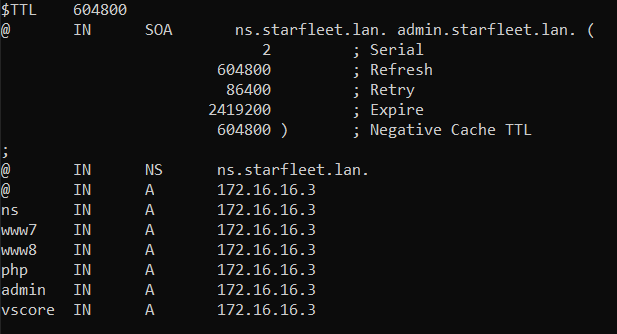
Éditez le fichier de configuration de Bind pour ajouter la nouvelle zone. **/etc/bind/named.conf.local**

On ajoute les informations de notre zone locale. Par exemple, pour gérer le domaine : **starfleet.lan**



Cela dit à **Bind9** qu'il s'agit du serveur maître pour la zone **starfleet.lan** et que la configuration de cette zone est stockée dans le fichier **/etc/bind/db.starflee.lan**

Créer le fichier de zone **nano /etc/bind/db.starfleet.lan**

Le fichier doit ressembler à ceci

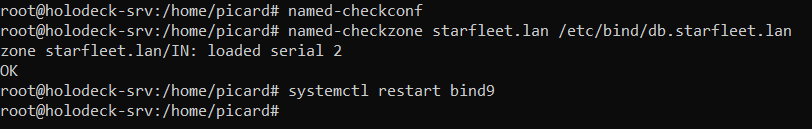
Le **@** désigne la zone racine (le domaine **starfleet.lan**).

Le serveur **NS** est configuré pour le serveur **ns.starfleet.lan** avec l'adresse IP du serveur DNS.

**starfleet.lan** est configuré avec l'adresse IP de notre serveur web.

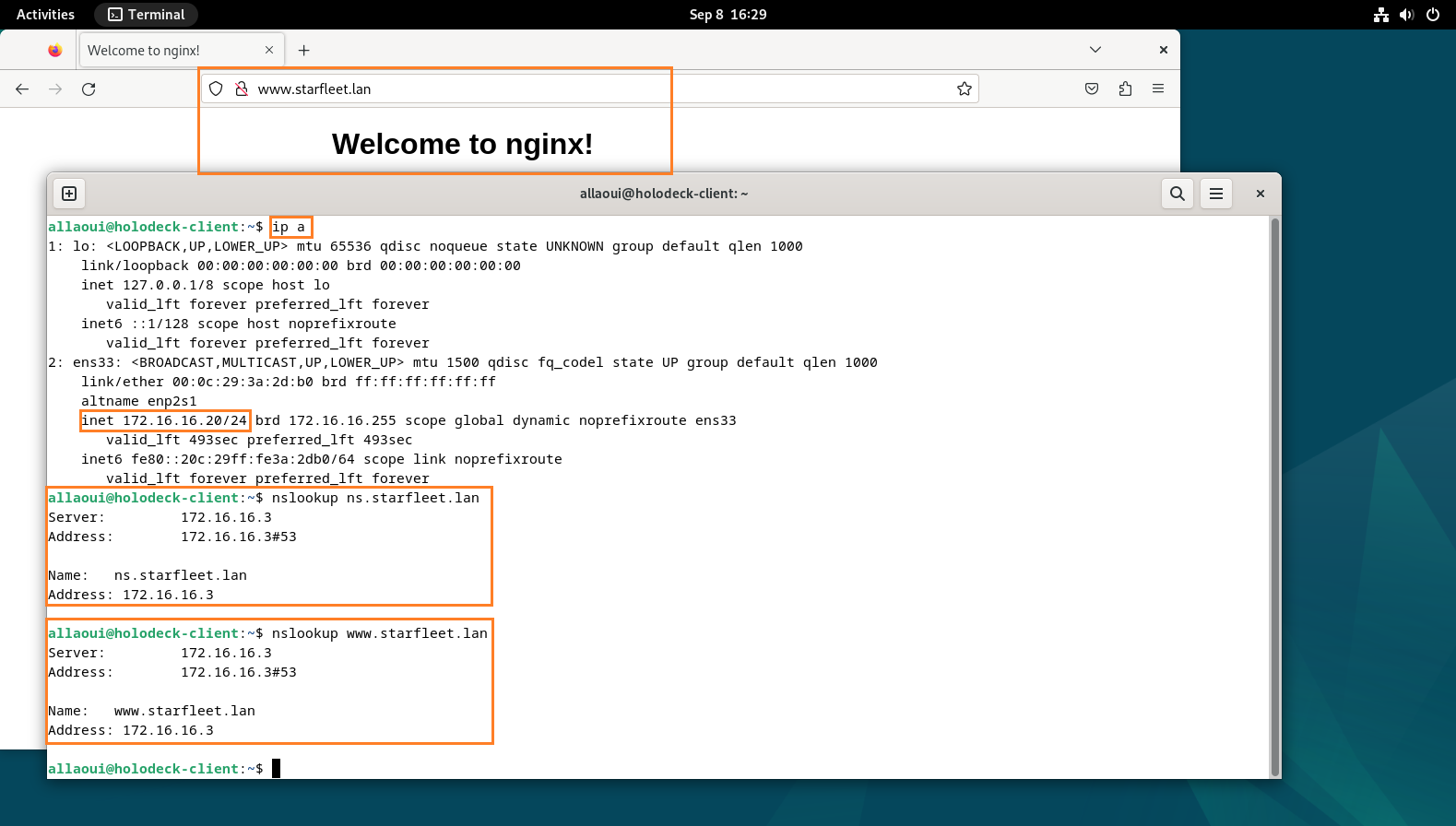
Ensuite, nous vérifions la configuration en utilisant les commandes suivantes :

**named-checkconf**

**named-checkzone starfleet.lan /etc/bind/db.starfleet.lan**

Si ces commandes ne renvoient aucune erreur, cela signifie que la configuration est correcte.

On redémarre le service Bind9 est on passe au test : **systemctl restart bind9**



Comme on peut le voir ici (image au-dessus), en exécutant la commande **ip a** sur notre machine cliente, une adresse IP (**172.16.16.20**) lui a bien été attribuée automatiquement.

Un peu plus bas, en effectuant un **nslookup** sur les noms de domaines **ns.starfleet.lan** et **www.starfleet.lan**, nous voyons qu'ils sont correctement résolus, ce qui prouve que la résolution DNS fonctionne correctement.

Ainsi, notre serveur DHCP et DNS est bien opérationnel !

## **Serveur Web Nginx (avec PHP 7.x et 8.x)**

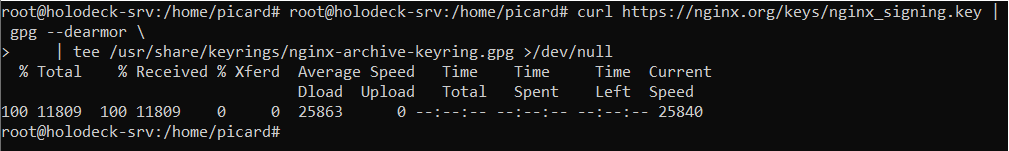
# Installation de Nginx :

Installer la dernière version de **NGINX** à partir du dépôt officiel (Instruction [ici](https://docs.nginx.com/nginx/admin-guide/installing-nginx/installing-nginx-open-source/#installing-prebuilt-debian-packages))

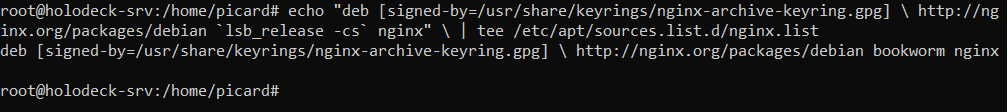
1. Installer les prérequis :

**apt install curl gnupg2 ca-certificates lsb-release debian-archive-keyring**

1. Télécharger et ajouter la clé publique du dépôt NGINX (pour qu'**apt** puisse vérifier l'authenticité des paquets) :

**curl https://nginx.org/keys/nginx\_signing.key | gpg --dearmor \ | tee /usr/share/keyrings/nginx-archive-keyring.gpg >/dev/null**

1. On configure le dépôt apt pour installer les paquets nginx **stables** :

**echo "deb [signed-by=/usr/share/keyrings/nginx-archive-keyring.gpg] \ http://nginx.org/packages/debian `lsb\_release -cs` nginx" \ | tee /etc/apt/sources.list.d/nginx.list**

1. **echo -e "Package: \*\nPin: origin nginx.org\nPin: release o=nginx\nPin-Priority: 900\n" \**

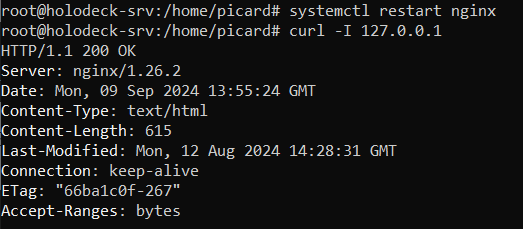
**| tee /etc/apt/preferences.d/99nginx**

1. Installer le paquet NGINX :

**apt update**

**apt install nginx**

1. systemctl restart nginx && curl -I 127.0.0.1



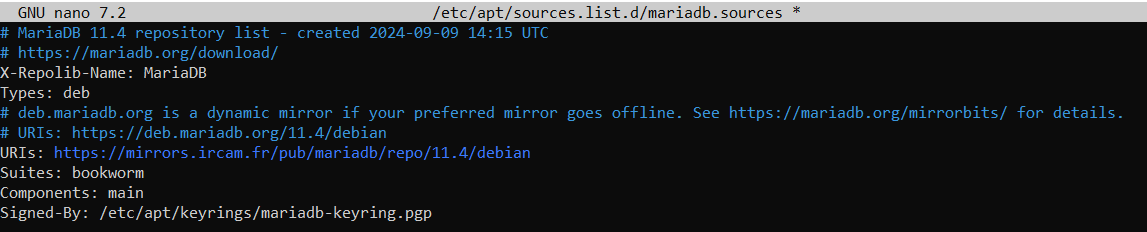
# Installation de MariaDB :

Installer la dernière version de **MariaDB** à partir du dépôt officiel (Instruction [ici](https://mariadb.org/download/?t=repo-config&d=Debian+12+%22Bookworm%22&v=11.4&r_m=icam))

1. Installer les prérequis : **apt-get install apt-transport-https curl**
2. Créer un dossier ou viendra loger la clé publique de MariaDB :

**mkdir -p /etc/apt/keyrings && curl -o /etc/apt/keyrings/mariadb-keyring.pgp 'https://mariadb.org/mariadb\_release\_signing\_key.pgp'**

1. On édite le fichier **.source** de mariadb nano **/etc/apt/sources.list.d/mariadb.sources**
2. On copie colle les info du repos MariaDB (Instruction [ici](https://mariadb.org/download/?t=repo-config&d=Debian+12+%22Bookworm%22&v=11.4&r_m=icam))

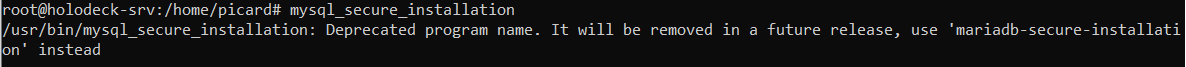


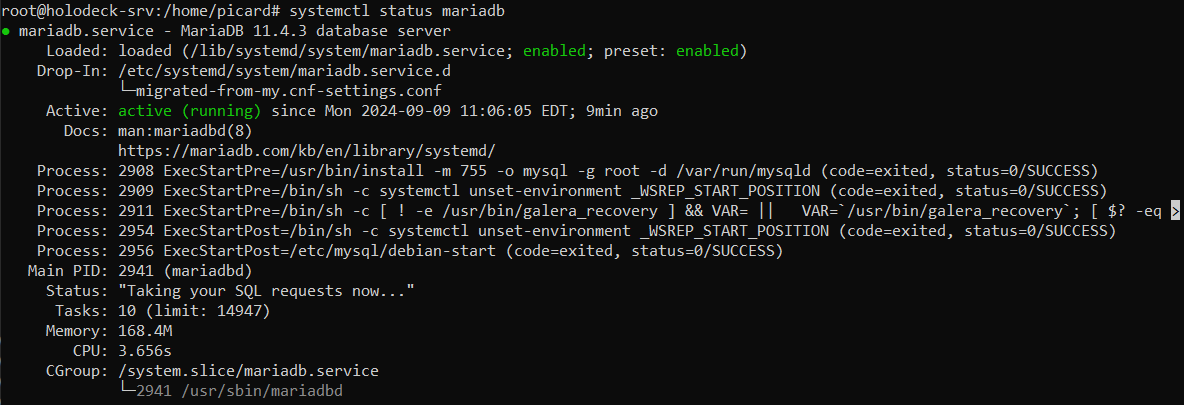
1. Maintenant on peut installer la dernière version de MariaDB **stable**

**apt update**

**apt install mariadb-server**

1. Si on verife avec la commande , on peut voir qu’on a bien la dernière version stable de MariaDB : **mariadb --version**
2. Une fois l'installation de MariaDB terminée, on va maintenant sécuriser son installation et personnaliser comme on le souhaite, avec la commande : **mysql\_secure\_installation**



1. On voit que la BDD MariaDB est bien en actif

# Installation de PHP 7.4 et 8.3

Pour installer et configurer deux versions de PHP sur notre serveur **Nginx** , on va effectuer plusieurs étapes. Tout d'abord, on va ajouter les dépôts nécessaires pour PHP 7.4 et PHP 8.3 (la version 7.4 n'est plus incluse par défaut dans les dépôts officiels de Debian).

* Installer les prérequis :

**apt update**

**apt install ca-certificates apt-transport-https software-properties-common -y**

* Ajouter le dépôt Sury :

**wget -qO - https://packages.sury.org/php/apt.gpg | tee /etc/apt/trusted.gpg.d/sury-php.gpg**

**sh -c 'echo "deb https://packages.sury.org/php/ bookworm main" > /etc/apt/sources.list.d/php.list'**

* Mettre à jour la liste des paquets :

**apt update**

* Installer PHP 7.4 et PHP 8.3 :

**apt install php7.4 php7.4-fpm php7.4-cli php7.4-mysql php7.4-curl php7.4-xml php7.4-mbstring**

**apt install php8.3 php8.3-fpm php8.3-cli php8.3-mysql php8.3-curl php8.3-xml php8.3-mbstring**

* Démarrer, activer et vérifier son statut :

**systemctl start php7.4-fpm && systemctl enable php7.4-fpm**

**systemctl start php8.3-fpm && systemctl enable php8.3-fpm**

**systemctl status php7.4-fpm**

**systemctl status php7.4-fpm**

Maintenant que les deux versions de PHP sont installées et fonctionnent, on va maintenant configurer Nginx pour qu'il utilise PHP 7.4 pour **www7.starfleet.lan** et PHP 8.3 pour **www8.starfleet.lan**

* Créer les dossiers sites-available et sites-enabled (Uniquement s’ils n’existent pas déjà)

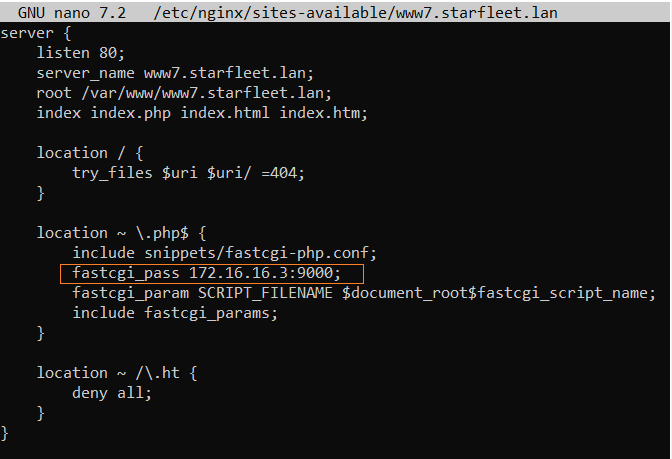
**mkdir /etc/nginx/sites-available && mkdir /etc/nginx/sites-enabled**

* Crée un fichier de configuration pour :

**nano /etc/nginx/sites-available/www7.starfleet.lan**

**nano /etc/nginx/sites-available/www8.starfleet.lan**

Le contenu du fichier [ici](https://drive.google.com/file/d/1E37WlbBZcos7oD086x4_3aePnBhu4GFS/view?usp=sharing) :



On indique notre serveur et le port sur lequel écouter :

**fastcgi\_pass 172.16.16.3 :9000 (pour php7.2)**

**fastcgi\_pass 172.16.16.3 :9001 (pour php8.2)**

Créer les répertoires web pour chaque site :

**mkdir -p /var/www/www7.starfleet.lan && mkdir -p /var/www/www8.starfleet.lan**

Attribuer les droits:

**chown -R www-data:www-data /var/www/www7.starfleet.lan**

**chown -R www-data:www-data /var/www/www8.starfleet.lan**

Activer les sites dans Nginx :

**ln -s /etc/nginx/sites-available/www7.starfleet.lan /etc/nginx/sites-enabled/**

**ln -s /etc/nginx/sites-available/www8.starfleet.lan /etc/nginx/sites-enabled/**

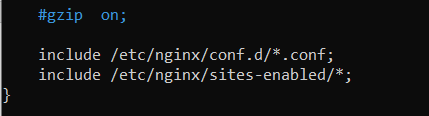
Crée un fichier **index.php** dans chaque répertoire pour tester les versions de PHP :

**echo "<?php phpinfo(); ?>" | tee /var/www/www7.starfleet.lan/index.php**

**echo "<?php phpinfo(); ?>" | tee /var/www/www8.starfleet.lan/index.php**

Ouvrir le fichier principal de configuration de Nginx : **nano /etc/nginx/nginx.conf** et ajouter la ligne :

**include /etc/nginx/sites-enabled/\*;**



Modifier le ficher **fastcgi-php.conf** , si existe pas le créer :

**mkdir /etc/nginx/snippets/ && nano /etc/nginx/snippets/fastcgi-php.conf**

Et ajoute ses lignes :

**fastcgi\_split\_path\_info ^(.+\.php)(/.+)$;**

**fastcgi\_index index.php;**

**include fastcgi\_params;**

**fastcgi\_param SCRIPT\_FILENAME $document\_root$fastcgi\_script\_name;**

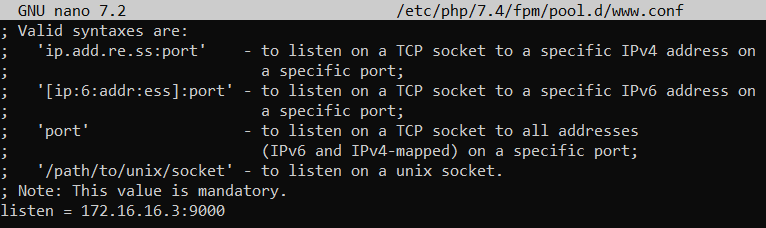
**fastcgi\_param PATH\_INFO $fastcgi\_path\_info;**

Ensuite on modifie PHP-FPM pour écouter sur TCP pour cela on va :

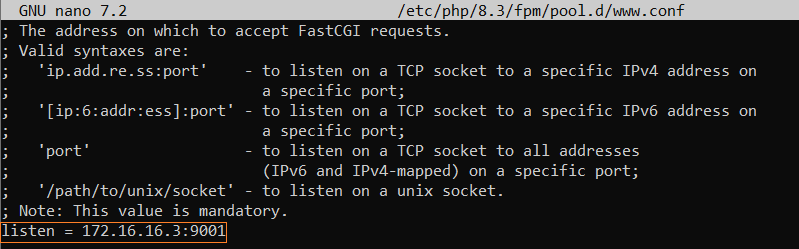
**nano /etc/php/7.4/fpm/pool.d/www.conf (pour PHP 7.4)**

**nano /etc/php/8.3/fpm/pool.d/www.conf (pour PHP 8.3)**

Et modifier la ligne : **listen = /run/php/php7.4-fpm.sock** en **listen = 172.16.16.3:9000**



**listen = /run/php/php8.3-fpm.sock** en **listen = 172.16.16.3:9001**

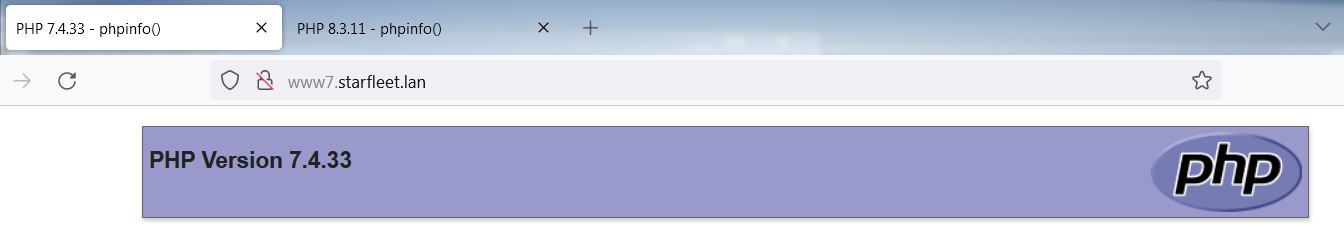


Ensuite on restart tout et on vérifie :

**systemctl restart php7.4-fpm && systemctl restart php8.3-fpm**

**systemctl restart nginx**

On peut voir que notre site **www7.starfleet.lan** utilise la version 7.4 de PHP



Et **www8.starfleet.lan** utilise la version 8.3 de PHP



# Installation de phpMyAdmin

1. Installation : **apt install phpmyadmin**
2. Création d'un lien symbolique pour PhpMyAdmin :

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

1. Attribuer les permissions :

**chown -R www-data:www-data /usr/share/phpmyadmin**

**chmod -R 755 /usr/share/phpmyadmin**

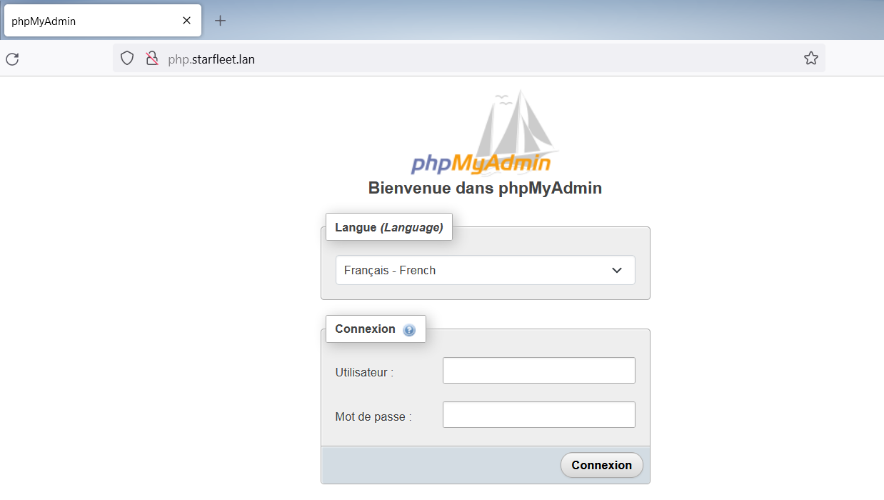
1. Créer le fichier de configurer et ajouter le contenue ([ici](https://drive.google.com/file/d/1HsJOt-ZoPlzcbMg_HpPdHb_ezA0FswML/view?usp=sharing)):

**nano /etc/nginx/sites-available/php.starfleet.lan**

1. Activer le site dans Nginx :

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

1. Redémarre les services PHP et Nginx :

**systemctl restart php8.3-fpm && systemctl restart nginx**

1. Maintenant on vérifie si **phpMyAdmin** est bien installer sur notre site **php.starfleet.lan**:

# Mettre en place HTTPS avec un certificat SSL pour nos sites

1. Installation de OpenSSL : **apt install openssl**
2. Créez un répertoire où va stocker les certificats et les clés :

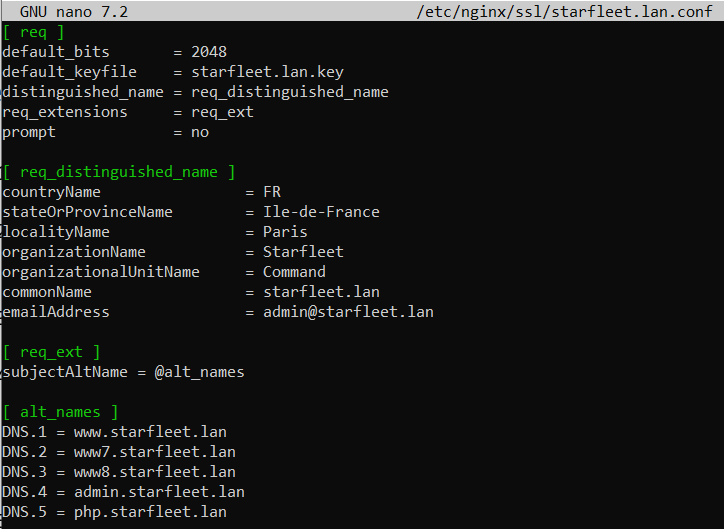
**mkdir -p /etc/nginx/ssl && cd /etc/nginx/ssl**

1. Générer une clé privée pour notre certificat :

**openssl genrsa -out starfleet.lan.key 2048**

1. On va créer un fichier de configuration et inclure tous nos sous-domaines ([ici](https://drive.google.com/file/d/16HnsmoeVTz-XqgP2CvbNvrwvLw9QqGCd/view?usp=sharing)) :

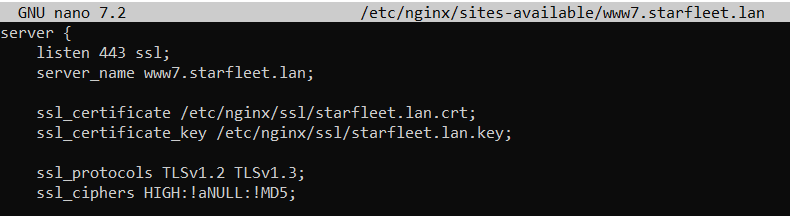
**nano /etc/nginx/ssl/starfleet.lan.conf**



1. On génère maintenant le certificat auto-signé :

**openssl req -new -x509 -sha256 -days 365 -key starfleet.lan.key -out starfleet.lan.crt -config starfleet.lan.conf**

1. Ensuite on modifie la configuration de chaque site pour activer HTTPS et utiliser le certificat :



1. Maintenant on redémarre Nginx pour appliquer les nouvelles configurations :

**systemctl restart nginx**

1. Maintenant, on passe au test de chaque domaine :



1. Tout fonctionne donc parfait !!
2. **GRANT ALL PRIVILEGES ON \*.\* TO 'phpmyadmin'@'localhost' WITH GRANT OPTION;**
3. **FLUSH PRIVILEGES;**
4. **systemctl restart mysql**

# Installation de l’Administration avec Zabbix Server

1. Ajoutez le dépôt Zabbix :

**wget https://repo.zabbix.com/zabbix/7.0/debian/pool/main/z/zabbix-release/zabbix-release\_7.0-2+debian12\_all.deb**

1. Créer le un fichier **bashrc** et ajoute des chemins supplémentaires au **$PATH** afin de pouvoir utiliser les commandes situées dans **/usr/local/sbin**, **/usr/sbin**, et **/sbin** sans spécifier leur chemin complet. :

**echo "export PATH=$PATH:/usr/local/sbin:/usr/sbin:/sbin" | tee bashrc**

1. On déplace le fichier **bashrc** et le rend exécutable. Ensuite, la commande **source** recharge le fichier pour appliquer les changements :

**mv bashrc ~/.bashrc && chmod +x ~/.bashrc && source ~/.bashrc**

1. On installe le fichier **.deb** téléchargé précédemment, ce qui va ajouter les dépôts Zabbix à notre système :

**dpkg -i zabbix-release\_7.0-2+debian12\_all.deb**

1. Ensuite on fait un **apt update** pour mettre à jour la liste des paquets pour inclure ceux des dépôts Zabbix nouvellement ajoutés : **apt update**
2. Installation de Zabbix et des composants requis :

**apt install zabbix-server-mysql zabbix-frontend-php zabbix-nginx-conf zabbix-sql-scripts zabbix-agent**

1. Configuration de MariaDB, on va créer la base de donner et un utilisateur :

**CREATE DATABASE zabbix CHARACTER SET utf8mb4 COLLATE utf8mb4\_bin;**

**CREATE USER 'zabbix'@'localhost' IDENTIFIED BY 'Azerty1234';**

**GRANT ALL PRIVILEGES ON zabbix.\* TO 'zabbix'@'localhost';**

**FLUSH PRIVILEGES;**

**EXIT;**

1. Import les scripts SQL pour Zabbix dans la base de données :

**zcat /usr/share/zabbix-sql-scripts/mysql/server.sql.gz | mariadb --default-character-set=utf8mb4 -u zabbix -p zabbix**

1. Désactive la vérification stricte des créateurs de fonctions pour permettre à Zabbix d'utiliser des fonctions SQL dans le serveur MariaDB sans problème :

**mariadb –u root -p**

**set global log\_bin\_trust\_function\_creators = 0;**

**quit;**

1. Configuration du serveur Zabbix , on modifie le fichier de configuration de Zabbix, et s'assurer que les valeurs suivantes sont correctement définies :

**DBName=zabbix**

**DBUser=zabbix**

**DBPassword=Azerty1234** #Votre mot de passe

1. Crée et édite un fichier de configuration pour le serveur web ([ici](https://drive.google.com/file/d/17XZE3M8Ng1VGRZIDdWwb-61Wo4S4VR4i/view?usp=sharing)) :

**nano /etc/nginx/sites-available/admin.starfleet.lan**

1. Ensuite on crée un lien symbolique pour activer le nouveau site dans NGINX :

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

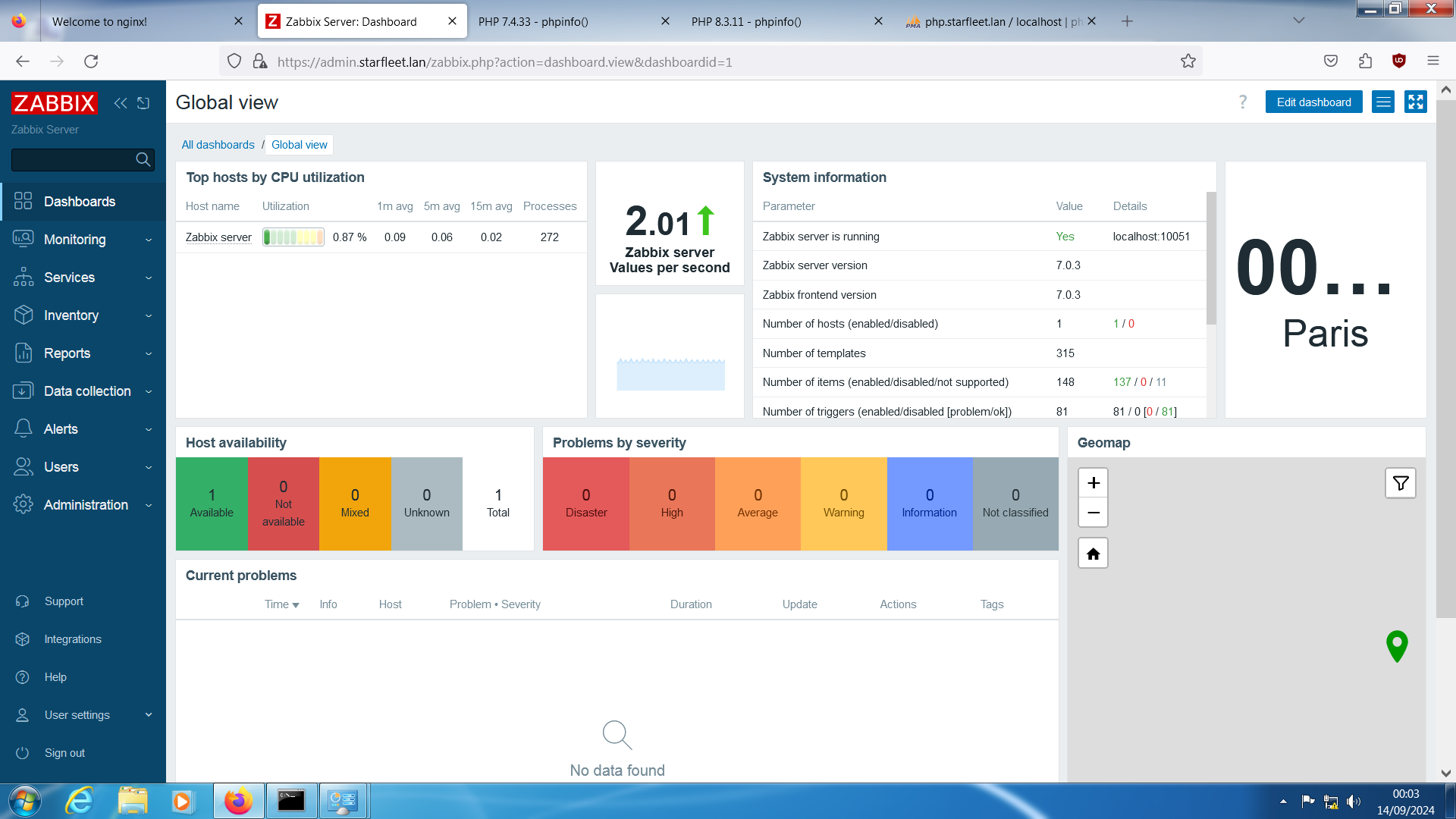
1. On active et démarre les services Zabbix server et Agent :

**systemctl enable zabbix-server zabbix-agent**

**systemctl start zabbix-server zabbix-agent**

1. Et enfin on redémarre NGINX pour prendre en compte les changements et passer au test :

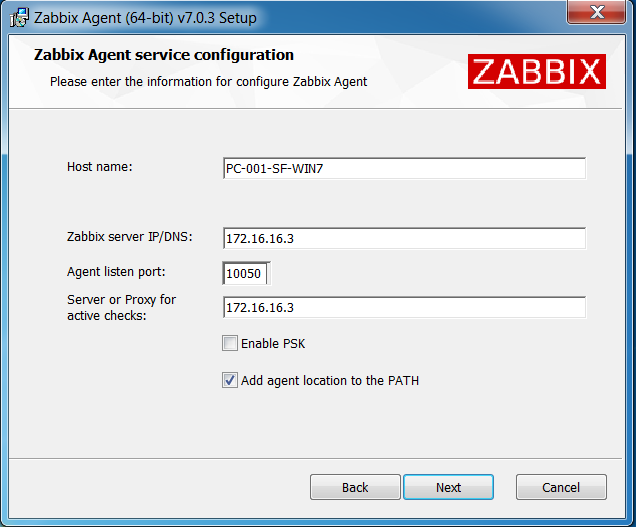
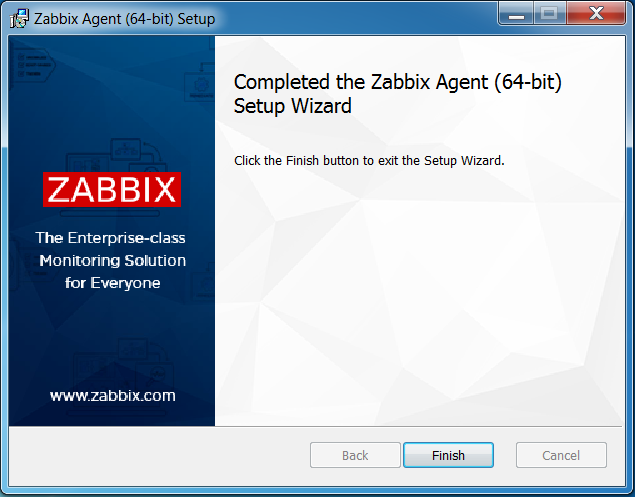
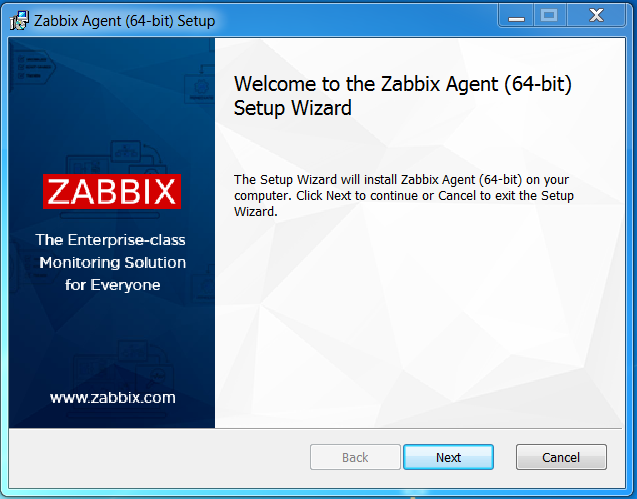
**systemctl restart nginx**



# Installation de l’Agent Zabbix sur nos PC clients

#### SUR WINDOWS :

1. Allez sur le site de Zabbix pour télécharger l'agent correspondant à notre OS [ici](https://www.zabbix.com/download_agents)
2. Procédez ensuite à l'installation en suivant les instructions.



**HOSTNAME** : NOMDUPC | **ZABBIX SERVER** : IPDUSRV | **SERVER** : IPDEDUSRV

1. Après avoir configuré l'agent, vérifiez si le Zabbix Agent a bien démarré et est en cours d'exécution dans les services (**services.msc**)



Comme on peut le voir il a bien démarré et est en cours d'exécution

#### SUR LINUX :

1. Ajoutez le dépôt Zabbix :

**wget https://repo.zabbix.com/zabbix/7.0/debian/pool/main/z/zabbix-release/zabbix-release\_7.0-2+debian12\_all.deb**

1. Créer le un fichier **bashrc** et ajoute des chemins supplémentaires au **$PATH** afin de pouvoir utiliser les commandes situées dans **/usr/local/sbin**, **/usr/sbin**, et **/sbin** sans spécifier leur chemin complet. :

**echo "export PATH=$PATH:/usr/local/sbin:/usr/sbin:/sbin" | tee bashrc**

1. On déplace le fichier **bashrc** et le rend exécutable. Ensuite, la commande **source** recharge le fichier pour appliquer les changements :

**mv bashrc ~/.bashrc && chmod +x ~/.bashrc && source ~/.bashrc**

1. On installe le fichier **.deb** téléchargé précédemment, ce qui va ajouter les dépôts Zabbix à notre système :

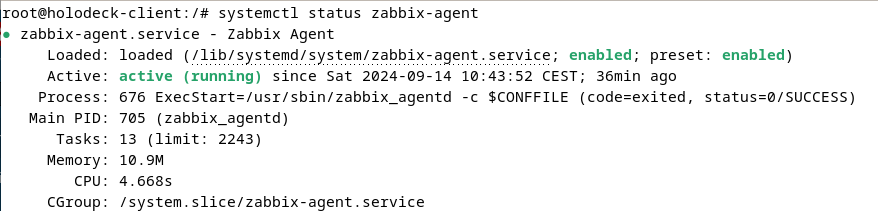
**dpkg -i zabbix-release\_7.0-2+debian12\_all.deb**

1. Ensuite on fait un **apt update** pour mettre à jour la liste des paquets pour inclure ceux des dépôts Zabbix nouvellement ajoutés : **apt update**
2. Installer l’Agent : **apt install zabbix-agent –y**
3. Modifier le fichier de configuration de Zabbix, décommenter et remplacer les IP (**Server=127.0.0.1**, **ServerActive=127.0.0.1**) par celle du serveur Zabbix, et remplacer "**Hostname=Zabbix server**" par le NOMDUPC



1. On active, démarre l'agent et vérifie son statut : **systemctl enable zabbix-agent**

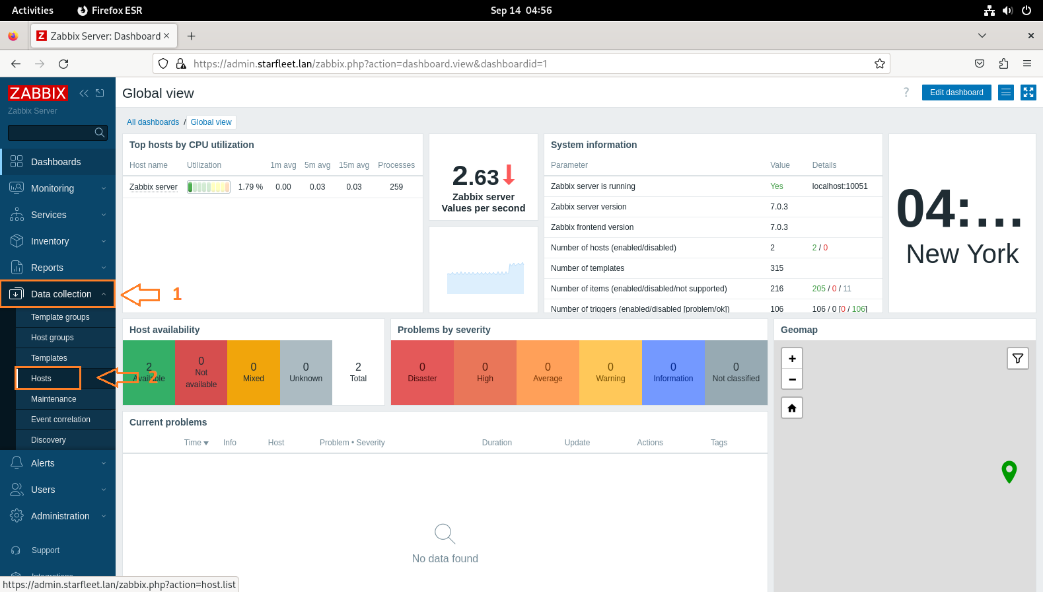
**systemctl start zabbix-agent systemctl status zabbix-agent**

****

Comme on peut le voir il Zabbix Agent a bien démarré et est en cours d'exécution

Maintenant allons ajouter nos PC’s sur le Serveur Zabbix via l’interface

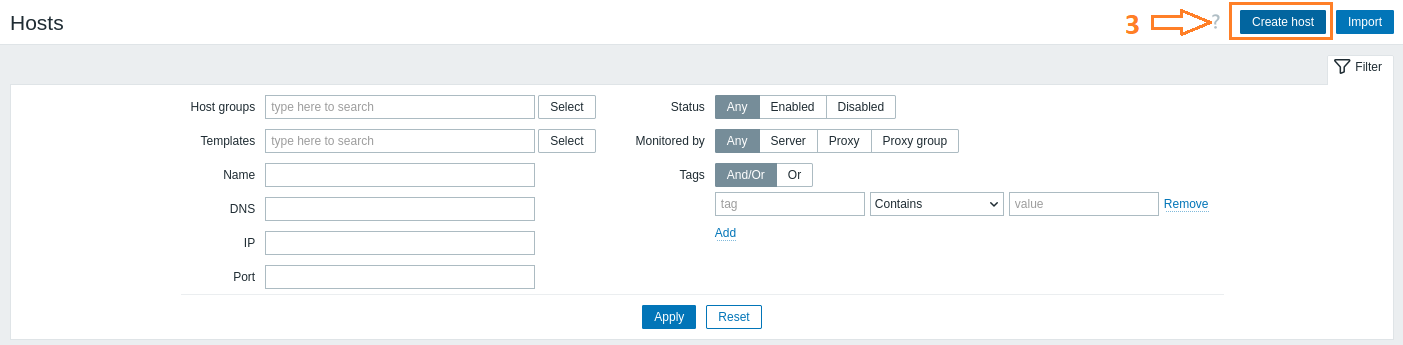
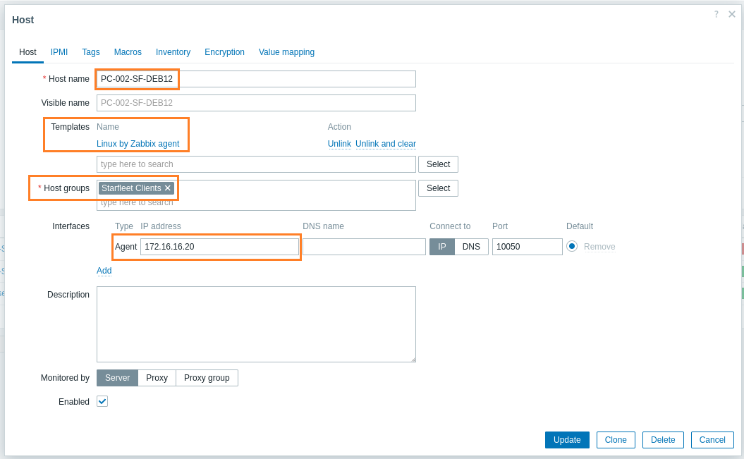
#### AJOUTS DES PCS CLIENTS SUR LE SERVEUR ZABBIX

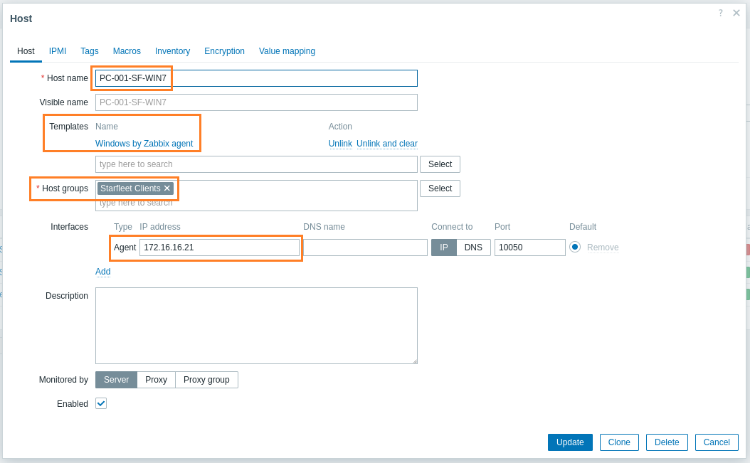


1 – Sur l'interface, cliquez sur **Data collection**,

2 – Ensuite, cliquez sur **Host**

3 – Dans **Host**, cliquez en haut à droite sur Create host

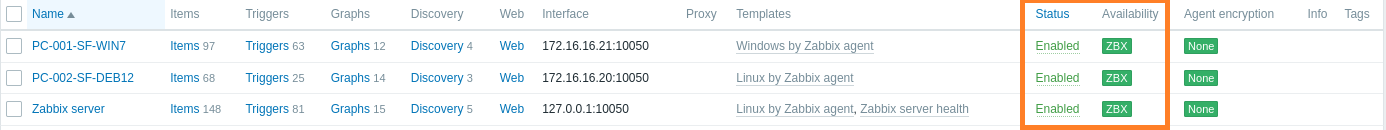


 **HOSTNAME** : NOMDUPC | **TEMPLATE** : SELON L’OS | **HOSTGROUPS** : GROUPES | **INTERFACES** : IPCLIENT

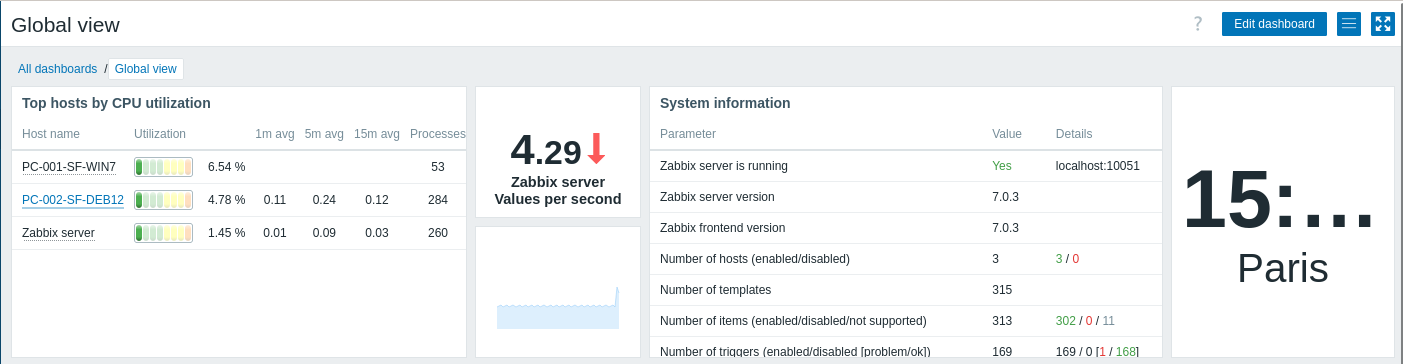
Choisissez le **Template** correspondant à l'OS du client (Windows ou Linux by Zabbix Agent).

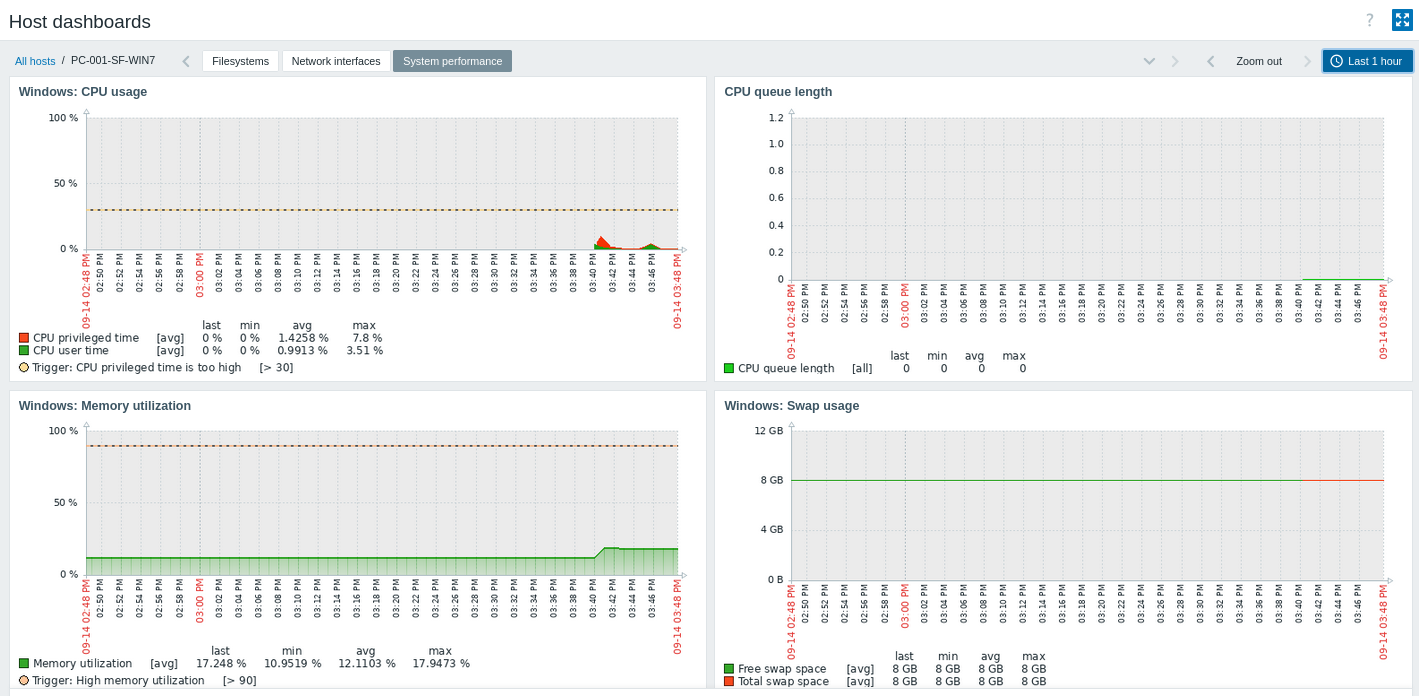
Dans **Interfaces**, sélectionnez **Agent** et renseignez l'IP du client. Enregistrez, puis répétez cette opération pour chaque client.

Maintenant, après avoir ajouté nos machines clientes, on peut voir qu'elles sont disponibles et actives.



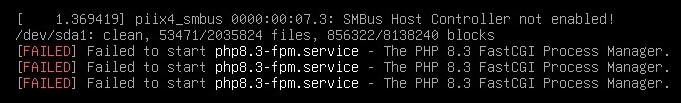
NOUS POUVONS À PRÉSENT SURVEILLER NOS MACHINES, VOIR LEUR ÉTAT ET LEURS PERFORMANCES





INSTALLATION DE ZABBIX SERVER ET CLIENTS DONE

# Solution (Si PHP8.3-FPM démarre pas) (vidéo):



1. Crée une copie de sauvegarde du fichier **php8.3-fpm.service** dans Systemd :

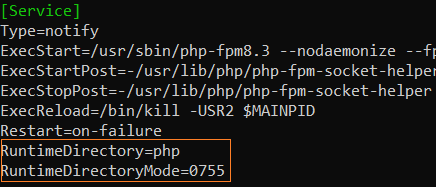
**cp /lib/systemd/system/php8.3-fpm.service /lib/systemd/system/php8.3-fpm.service.old**

1. Editer le fichier et ajouter ces deux lignes après la section [Service] :

**nano /lib/systemd/system/php8.3-fpm.service**

**RuntimeDirectory=php**

**RuntimeDirectoryMode=0755**

****

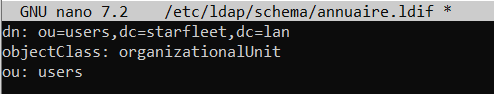
1. On enregistre, on recharge Systemd, puis on redémarre php8.3-fpm :

**systemctl daemon-reload && restart php8.3-fpm**

Mise en place d'un annuaire LDAP :

1. On met à jour les paquets et installe OpenLDAP : **apt update && apt install slapd ldap-utils**
2. Ensuite, on crée un schéma LDAP où nous allons définir nos unités organisationnelles (OU) :

**nano /etc/ldap/schema/annuaire.ldif**



**dn: ou=users,dc=starfleet,dc=lan**

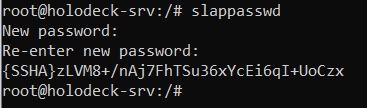
**objectClass: organizationalUnit**

**ou: users**

1. Ensuite, nous allons appliquer et ajouter l'OU au serveur LDAP avec la commande :

**ldapadd -x -D cn=admin,dc=starfleet,dc=lan -W -f /etc/ldap/schema/annuaire.ldif**

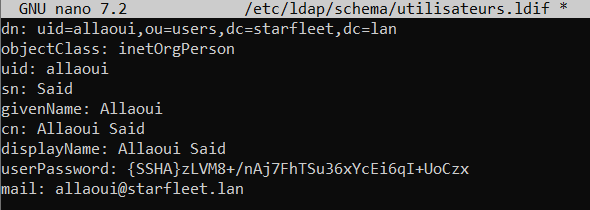


1. Ensuite on va créer une liste avec nos utilisateurs, mais avant on va générer un mot de passe pour chaque utilisateur avec la commande : **slappasswd**

{SSHA}zLVM8+/nAj7FhTSu36xYcEi6qI+UoCzx

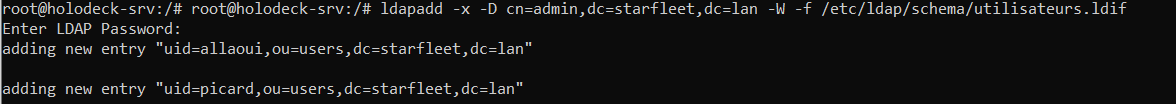
1. On crée un fichier LDIF et, à l'intérieur, on ajoute les informations de nos utilisateurs :

**nano /etc/ldap/schema/utilisateurs.ldif**

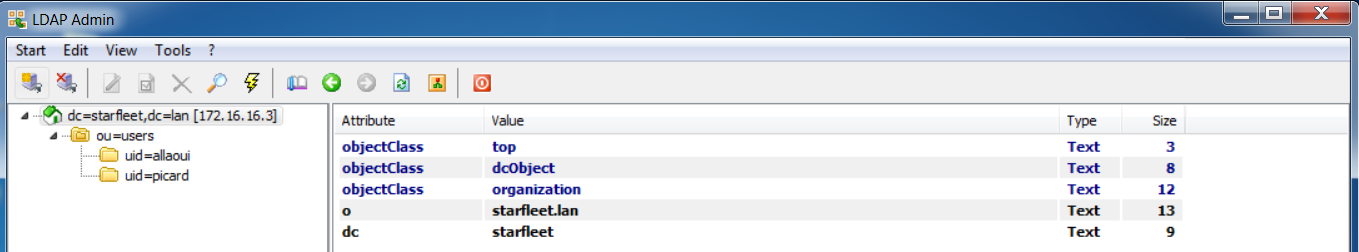


1. On ajoute ces utilisateurs au serveur LDAP :

**ldapadd -x -D cn=admin,dc=starfleet,dc=lan -W -f /etc/ldap/schema/utilisateurs.ldif**



1. Nous avons maintenant un annuaire fonctionnel (mais non sécurisé) :



Screenshot prise sur une machine cliente test avec le logiciel LDAP Admin (On voit que notre serveur n’est pas sécure)

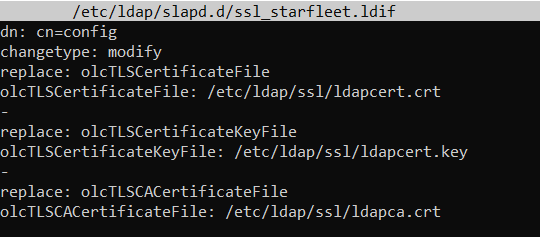
Sécurisation de LDAPS (SSL/TLS) :

1. Créez les certificats nécessaires (commande à saisir one by one) :

**mkdir /etc/ldap/ssl/**

* **openssl genrsa -out /etc/ldap/ssl/ldapca.key 2048**
* **openssl req -x509 -new -nodes -key /etc/ldap/ssl/ldapca.key -sha256 -days 3650 -out /etc/ldap/ssl/ldapca.crt**
* **openssl genrsa -out /etc/ldap/ssl/ldapcert.key 2048**
* **openssl req -new -key /etc/ldap/ssl/ldapcert.key -out /etc/ldap/ssl/ldapcert.csr**
* **openssl x509 -req -in /etc/ldap/ssl/ldapcert.csr -CA /etc/ldap/ssl/ldapca.crt -CAkey /etc/ldap/ssl/ldapca.key -CAcreateserial -out /etc/ldap/ssl/ldapcert.crt -days 365 -sha256**

1. Ensuite, nous allons configurer OpenLDAP pour utiliser les certificats. Pour cela, nous allons créer un fichier LDIF et y saisir les instructions : **nano /etc/ldap/slapd.d/ssl\_starfleet.ldif**



1. On attribue les permissions appropriées à OpenLDAP:

**chgrp -R openldap /etc/ldap/ssl/**

**chmod 0640 /etc/ldap/ssl/ldapcert.crt /etc/ldap/ssl/ldapcert.key /etc/ldap/ssl/ldapca.crt**

1. Et on applique les modifications :

**ldapmodify -Y EXTERNAL -H ldapi:/// -f /etc/ldap/slapd.d/ssl\_starfleet.ldif**

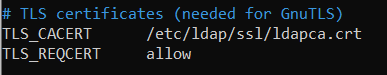
1. On édite le fichier de slapd et activer LDAPS : **nano /etc/default/slapd**

**SLAPD\_SERVICES="ldap:/// ldapi:/// ldaps:///"**



1. On édite aussi le fichier **/etc/ldap/ldap.conf** pour ajouter le certificat CA :

**nano /etc/ldap/ldap.conf**



**TLS\_CACERT /etc/ldap/ssl/ldapca.crt**

**TLS\_REQCERT allow**

1. Ensuite on redémarre pour appliquer les changements : **systemctl restart slapd**
2. On vérifie à nos nouveau si notre serveur est sécurisé :

EH BAH ON DIRAIT QUE OUI

Serveur FTPS :

1. Installation de ProFTPD : **apt install proftpd proftpd-mod-crypto –y**
2. On génère un certificat SSL :

**openssl req -x509 -newkey rsa:4096 -keyout /etc/ssl/private/proftpd.key -out /etc/ssl/certs/proftpd.crt -nodes -days 365**

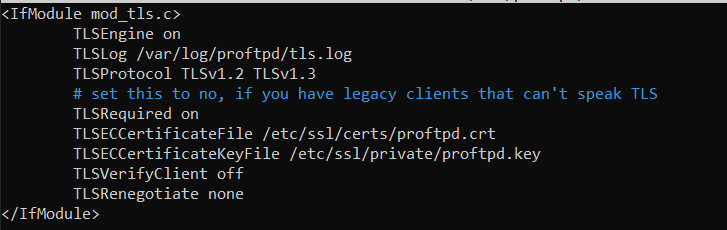
1. Modifie les permissions des certs SSL :

**chmod 0600 /etc/ssl/private/proftpd.key**

**chmod 0640 /etc/ssl/certs/proftpd.crt**

1. On édite nano **/etc/proftpd/modules.conf** et décommente : **LoadModule mod\_tls.c**
2. Ensuite on édite le fichier **nano /etc/proftpd/tls.conf ,** on vide tout le contenue remplace par :

**rm /etc/proftpd/tls.conf && nano /etc/proftpd/tls.conf**



1. On édite **nano /etc/proftpd/proftpd.conf** , décommente **Include /etc/proftpd/tls.conf** :

On ajoute tout en bas : **DefaultRoot /var/www/starfleet.lan**

1. On a fini avec ProFTPD : **proftpd --configtest** pour tester si on a aucune erreur puis

**systemctl restart proftpd**

1. Créer les répertoires et régle les permissions :

**mkdir -p /var/www/starfleet.lan**

**chown root:root /var/www/starfleet.lan**

**chmod 755 /var/www/starfleet.lan**

1. Créer les sous-répertoires pour les utilisateurs et les partages public :

**mkdir -p /var/www/starfleet.lan/public**

**mkdir -p /var/www/starfleet.lan/public/DOCUMENTS\_RH /var/www/starfleet.lan/public/PHOTOS /var/www/starfleet.lan/public/PROJETS /var/www/starfleet.lan/public/RESSOURCES /var/www/starfleet.lan/public/VIDEOS**

1. Créer un répertoire pour un utilisateur spécifique :

**mkdir -p /var/www/starfleet.lan/users/allaoui**

**chown root:root /var/www/starfleet.lan/users/allaoui**

**chmod -R 774 /var/www/starfleet.lan/users/allaoui**

**chown allaoui:allaoui /var/www/starfleet.lan/users/allaoui**

1. Pour créer un utilisateur : **useradd allaoui** && **passwd allaoui**

On maintenant un serveur FTPS configuré avec des utilisateurs restreints à leur propre répertoire (chroot), et les transferts de fichiers sont sécurisés grâce à TLS.

Installer Visual Studio Code Server :

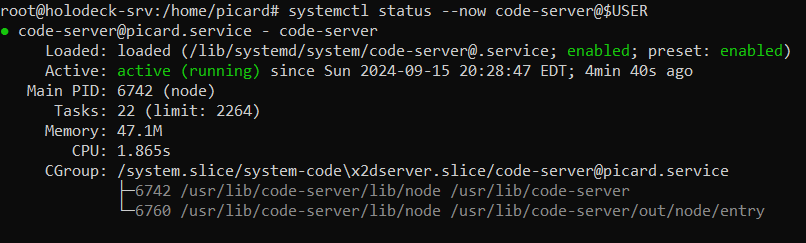
1. Téléchargez et installez Visual Studio Code Server : **curl -fsSL https://code-server.dev/install.sh | sh**
2. Ensuite on l’active et démarre le service :

**systemctl enable --now code-server@$USER**

**systemctl start --now code-server@$USER**

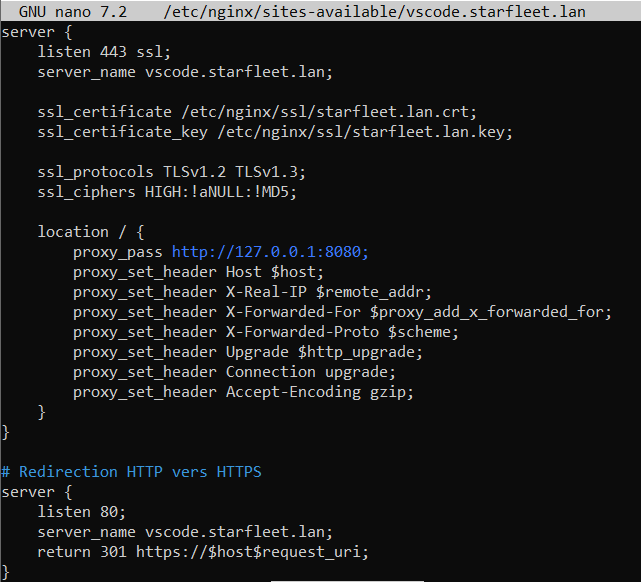
1. On vérifie ensuite son statut pour s'assurer qu'il est bien en cours :

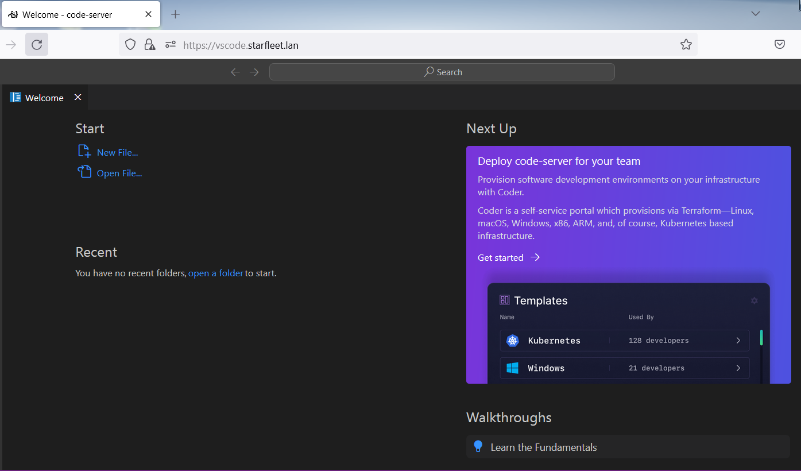
**systemctl status --now code-server@$USER**



1. Créer le fichier de configuration de notre site **vscode.starfleet.lan**:

**nano /etc/nginx/sites-available/vscode.starfleet.lan**



1. On active le site : **ln -s /etc/nginx/sites-available/vscode.starfleet.lan /etc/nginx/sites-enabled/**
2. Et on reload Nginx pour appliquer les config : **systemctl reload nginx**