Exercice



**Mes 2 VM sont :**

Server1-ubuntu (ou prod-ubuntu)

ssh dlaubo@192.168.100.19

Server2-ubuntu (ou test-ubuntu)

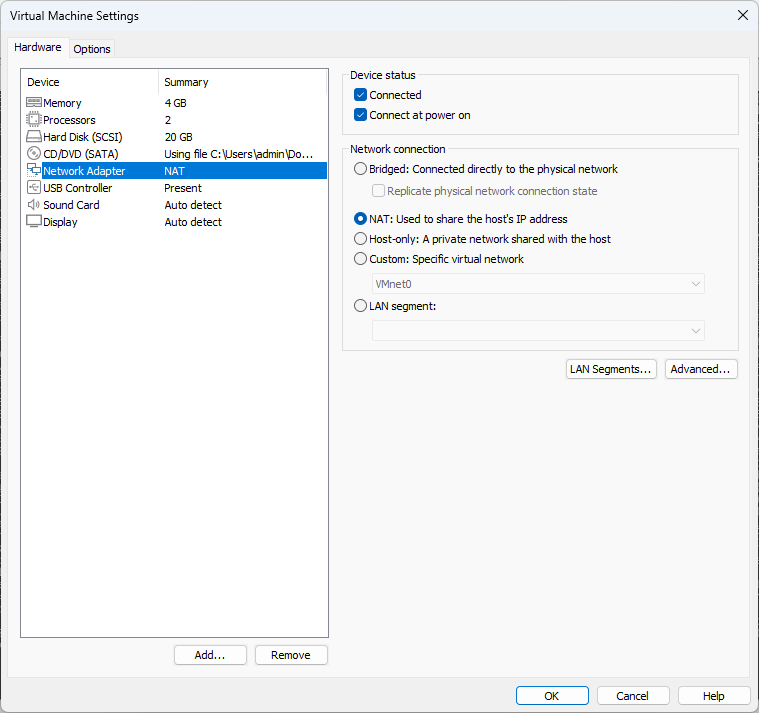
ssh dlaubo@192.168.100.18

**Configuration de prod-ubuntu**

Configurer les réseau ouvert NAT (permet la communication avec le serveur via @IP)

ssh dlaubo@192.168.100.19

Aller sur Settings de la machine prod > cocher le réseau NAT



Se connecter sur prod, exécuter la commande

Curl [www.google.com](http://www.google.com)

Le test réussi.

**Créer un déuxième utilisateur de type sudo testuser**

testuser / testuser

* sudo adduser testuser

Donner à testuser les droits de sudo

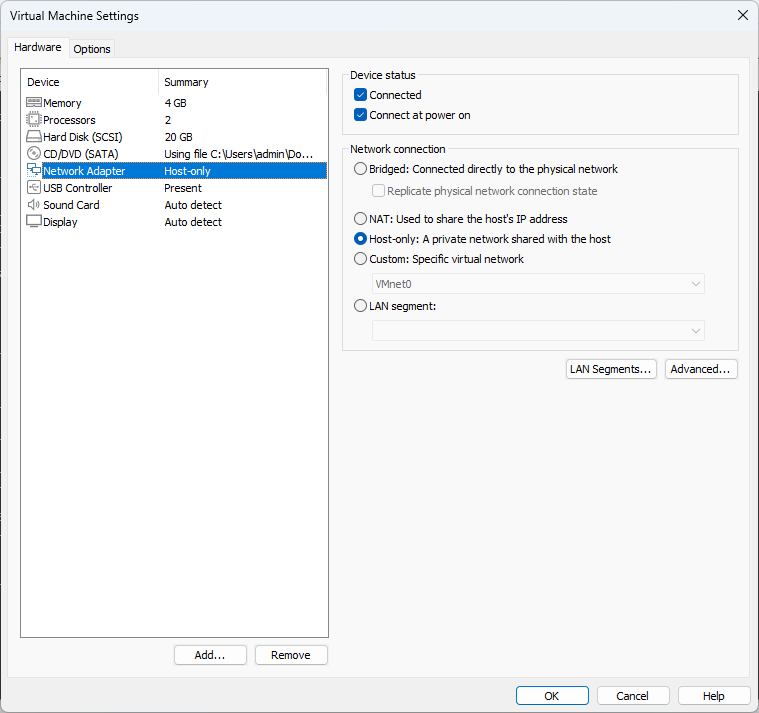
* sudo visudo

Ajouter la ligne suivante dans le groupe %sudo

testuser ALL=(ALL:ALL) ALL

**Configuration de test-ubuntu**

Utilisera le réseau hostonly : accès uniquement sur le réseau de VM’s



Se connecter sur test, exécuter la commande

Curl [www.google.com](http://www.google.com)

Le test échoue, la connexion ne passe pas !

**Créer un déuxième utilisateur de type sudo testuser**

testuser / testuser

* sudo adduser testuser

Donner à testuser les droits de sudo

* sudo visudo

Ajouter la ligne suivante dans le groupe %sudo

testuser ALL=(ALL:ALL) ALL

**Utiliser des clés privés et publics et sans mot de passe pour accéder aux serveurs prod et test**

D’abord, désactiver la configuration SSH de serveurs test et prod sut le réseau NAT.

Voici les fichiers à générer pour les privé / public

* Une clé privée : ~/.ssh/id\_rsa
* Une clé publique : ~/.ssh/id\_rsa.pub

**Les commandes à exécuter :**

ssh-keygen

Cette commande génère les 2 clés privé / public

**Exemple de dépôt de la clé publique sur les serveurs Prod et test**

ssh-copy-id [dlaubo@192.168.100.19](mailto:dlaubo@192.168.100.19)

ssh-copy-id [dlaubo@192.168.100.18](mailto:dlaubo@192.168.100.18)

nous pouvons utiliser l’option -i id\_rsa.pub pour préciser la clé si le fichier porte un nom différent de celui donné par défault.

Vérifier sur le serveur Prod et test dans le répertoire ~/.ssh que la clé est bien ajoutée dans les hosts autorisés et les authorized\_keys

**Maintenant la connexion ssh sur serveur prod et tes se fait par clé ssh et pas par une authentificaion**

**Configurer le FireWall**

sudo apt update

sudo apt install ufw -y

Les commandes de ‘ufw’

sudo ufw status

sudo ufw start

sudo systemctl ufw start

sudo systemctl start ufw

sudo ufw start

sudo ufw status

sudo systemctl start ufw

sudo ufw status

sudo ufw enable

sudo ufw default deny incoming

sudo ufw default allow outgoing

sudo ufw allow ssh

sudo ufw allow http

sudo ufw allow https

sudo ufw allow 8080

sudo ufw enable

sudo ufw status verbose

**Via cette commande on va les règles du FireWall**

dlaubo@server1-ubuntu:~$ sudo ufw status verbose

Status: active

Logging: on (low)

Default: deny (incoming), allow (outgoing), disabled (routed)

New profiles: skip

To Action From

-- ------ ----

22/tcp ALLOW IN Anywhere

80/tcp ALLOW IN Anywhere

443 ALLOW IN Anywhere

8080 ALLOW IN Anywhere

22/tcp (v6) ALLOW IN Anywhere (v6)

80/tcp (v6) ALLOW IN Anywhere (v6)

443 (v6) ALLOW IN Anywhere (v6)

8080 (v6) ALLOW IN Anywhere (v6)

**Installer NGINX sur les serveurs Prod et test**

Configurer les ports 8080 de NGINX sur les serveurs Prod et test sur le réseau NAT et donner les respectivement : 5022 (prod) et 5021 (test) pour qu’un appel extérieur via la machine physique (portable PC) soit rediriger sur le port http 80 de NGINX sur les serveurs.

Faire également, 5122 (prod) et 5121 (test) pour qu’un appel extérieur via la machine physique (portable PC) soit rediriger sur le port https 443 de NGINX sur les serveurs.

**Pour faire cette configuration, suivre les étapes suivantes sur chaque serveur prod et test**

# Activer le routage IP

sudo nano /etc/sysctl.conf

# Décommentez ou ajoutez la ligne suivante

# net.ipv4.ip\_forward=1

sudo sysctl -p

# Ajouter la règle de redirection de port

#sur serveur prod

sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 5022

sudo iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-port 5122

#sur serveur test

sudo iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT --to-port 5021

sudo iptables -t nat -A PREROUTING -p tcp --dport 443 -j REDIRECT --to-port 5121

# Installer iptables-persistent pour sauvegarder les règles

sudo apt install iptables-persistent -y

# Sauvegarder les règles iptables

sudo netfilter-persistent save

.

# Vérifier les règles iptables

sudo iptables -t nat -L

Vérifier que le FireWall est enable et le port 22 est active, sinon

sudo ufw allow 22

Et redémarrer le serveur.

**Voici les étapes à suivre pour installer NGING selon CHATGPT**

Pour installer Nginx sur Ubuntu, suivez ces étapes simples. Ce guide couvre l'installation de Nginx, le démarrage et l'arrêt du service, ainsi que la configuration de base.

### Étapes pour installer Nginx sur Ubuntu

#### 1. Mettre à jour le système

Avant d'installer Nginx, il est recommandé de mettre à jour votre liste de paquets pour vous assurer que vous obtenez la dernière version disponible. Ouvrez un terminal et exécutez :

bash

Copier le code

sudo apt update sudo apt upgrade -y

#### 2. Installer Nginx

Pour installer Nginx, utilisez la commande suivante :

bash

Copier le code

sudo apt install nginx -y

#### 3. Vérifier l'installation

Une fois l'installation terminée, vous pouvez vérifier que Nginx a été correctement installé et qu'il fonctionne en vérifiant le statut du service :

bash

Copier le code

sudo systemctl status nginx

Vous devriez voir une sortie indiquant que le service Nginx est actif et en cours d'exécution.

#### 4. Démarrer et arrêter Nginx

Vous pouvez démarrer, arrêter et redémarrer le service Nginx en utilisant les commandes suivantes :

* Pour démarrer Nginx :

bash

Copier le code

sudo systemctl start nginx

* Pour arrêter Nginx :

bash

Copier le code

sudo systemctl stop nginx

* Pour redémarrer Nginx :

bash

Copier le code

sudo systemctl restart nginx

* Pour recharger Nginx (utile après avoir modifié les fichiers de configuration sans arrêter le service) :

bash

Copier le code

sudo systemctl reload nginx

#### 5. Activer Nginx au démarrage

Pour s'assurer que Nginx démarre automatiquement au démarrage du système, utilisez la commande suivante :

bash

Copier le code

sudo systemctl enable nginx

#### 6. Configurer le pare-feu

Si vous utilisez ufw pour gérer votre pare-feu, vous devez autoriser le trafic HTTP (port 80) et HTTPS (port 443). Utilisez les commandes suivantes :

* Pour autoriser le trafic HTTP :

bash

Copier le code

sudo ufw allow 'Nginx HTTP'

* Pour autoriser le trafic HTTPS :

bash

Copier le code

sudo ufw allow 'Nginx HTTPS'

Pour vérifier les règles du pare-feu, utilisez :

bash

Copier le code

sudo ufw status

#### 7. Vérifier l'installation de Nginx

Ouvrez votre navigateur web et accédez à l'adresse IP de votre serveur ou à localhost si vous travaillez sur une machine locale. Vous devriez voir la page par défaut de Nginx, ce qui indique que Nginx est correctement installé et fonctionne.

arduino

Copier le code

http://your\_server\_ip

ou

arduino

Copier le code

http://localhost

### Configuration de base de Nginx

Les fichiers de configuration de Nginx se trouvent dans le répertoire /etc/nginx. Les fichiers importants incluent :

* /etc/nginx/nginx.conf : le fichier de configuration principal de Nginx.
* /etc/nginx/sites-available/ : ce répertoire contient les configurations de site disponibles.
* /etc/nginx/sites-enabled/ : ce répertoire contient les liens symboliques vers les fichiers de configuration des sites activés.

#### 1. Créer une nouvelle configuration de site

Pour créer une nouvelle configuration de site, vous pouvez créer un nouveau fichier dans le répertoire sites-available et ensuite créer un lien symbolique vers ce fichier dans le répertoire sites-enabled.

Par exemple, pour créer une configuration pour example.com :

bash

Copier le code

sudo nano /etc/nginx/sites-available/example.com

Ajoutez la configuration de base suivante :

nginx

Copier le code

server { listen 80; server\_name example.com www.example.com; root /var/www/example.com; index index.html index.htm; location / { try\_files $uri $uri/ =404; } }

#### 2. Activer la configuration du site

Créez un lien symbolique vers le fichier de configuration dans sites-enabled :

bash

Copier le code

sudo ln -s /etc/nginx/sites-available/example.com /etc/nginx/sites-enabled/

#### 3. Vérifier la configuration de Nginx

Avant de recharger Nginx, vérifiez la syntaxe de la configuration pour vous assurer qu'il n'y a pas d'erreurs :

bash

Copier le code

sudo nginx -t

#### 4. Recharger Nginx

Rechargez Nginx pour appliquer les modifications :

bash

Copier le code

sudo systemctl reload nginx

### Conclusion

En suivant ces étapes, vous avez installé Nginx sur Ubuntu, configuré le service pour démarrer automatiquement, autorisé le trafic HTTP et HTTPS via le pare-feu, et créé une configuration de site de base. Vous pouvez maintenant commencer à déployer vos sites web et applications sur votre serveur Nginx.

**Redirection des appels avec ngrok**

|  |  |  |
| --- | --- | --- |
| URL Ngrok | URL sur machine physique | Serveur et application |
| <https://8c53-212-114-26-208.ngrok-free.app> | <http://192.168.100.19:5022> | Appel Prod et nginx localhost:80 |
|  |  |  |

How to Use SSH to Access a Linux Machine from Windows

<https://www.codeproject.com/Articles/497728/HowplustoplusUseplusSSHplustoplusAccessplusaplusLi>