Document d'installation et d'utilisation

I- Installation

Préambule :

Étant dans un environnement de test et non de production, les mots de passe seront faibles.

1. Environnement :

-**machine physique hôte** – os variable

-logiciel Oracle Virtualbox avec :

-**VM1 : 'Master'** – Linux Debian 12

-**VM2 : 'Noeud'** – Linux Debian 12

-**VM3 : 'GUI'** – Windows 10

a

2. Configuration réseau Temporaire (pour installation des paquets)

=> Mode : NAT

<table>
<tr><th colspan="2">VM1/2 Linux</th><th>VM3 Windows</th></tr>
<tr><th>VM1 Master</th><th>VM2 Noeud</th><td rowspan="2">Panneau de config > Réseau et Internet > Centre Réseau et partage > Onglet de gauche 'Modifier les paramètres de la carte'
cliquer sur l'interface utilisé par virtualbox puis Propriétés > Protocole IPv4 > Obtenir une adresse IP automatiquement</td></tr>
<tr><td colspan="2">récupérer le nom de l'interface -> ip a
puis
nano /etc/network/interfaces
auto <nom_interface>
iface <nom> inet dhcp</td></tr>
</table>

Paquets/Logiciels à installer :

Master	Noeud	GUI
paquets systeme		*logiciel*
python3 python3-pip mariadb-server	python3 python3-pip	Python 3.12 : ! cocher la case 'Add to PATH' dans la 1e fenêtre d'installation ! + Git 2.52.0
modules python		
PyQt5 PyMySql	/	PyQt5 PyMySql

+ openssh-server pour le clonage de mon projet

Commandes :

VM1 :

apt update

apt install -y python3 python3-pip mariadb-server python3-pymysql python3-pyqt5

apt install -y openssh-server

systemctl status ssh

VM2 :

apt update

apt install -y python3 python3-pip python3-pymysql python3-pyqt5

apt install -y openssh-server

systemctl status ssh

VM3 :

python -m pip install pyqt5 pymysql

3. Clonage de mon repository GitHub

sur la machine hôte vers les VMs Linux :

Télécharger mon dossier projet 'sae302' depuis Github

puis dans powershell ou cmd :

scp -r <dossier local>\sae302 toto@192.168.106.10:/home/toto/Desktop

scp -r <dossier local>\sae302 toto@192.168.106.20:/home/toto/Desktop

puis :

```
The authenticity of host '192.168.106.10 (192.168.106.10)' can't be established.
ED25519 key fingerprint is SHA256:2GSN/M/kAHL6Aaa/KgjIJo3xGPzz3yJrolgaoDJCq3w.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
Warning: Permanently added '192.168.106.10' (ED25519) to the list of known hosts.
toto@192.168.106.10's password:
```

répondre 'yes' à la question puis remplir le mot de passe 'toto'

sur la VM Windows :

powershell

cd C:\Users\toto\Desktop

git clone https://github.com/chlo070/sae302.git

cd sae302\tor_sondag

4. Configuration réseau fixe (test de la communication oignon)

=> Mode : Réseau privé hôte (Host-Only)

Adressage ip des 3 machines :

VM1 => 192.168.x.10/24

VM2 => 192.168.x.20/24

VM3 => 192.168.x.30/24

avec 'x' dépendant de l'adresse réseau de l'interface privé de l'hôte

Carte Réseau privé de la MP :

VirtualBox > 'Outils' > 'Host-only Networks'

Si l'onglet est vide, créer une interface, puis récupérer son adresse IPv4 en .1/24.

On obtient donc l'adresse du réseau privé hôte utilisé -> adapter les adresses des machines en fonction

Configuration dans les VM :

<table>
<tr><th colspan="2">VM1/2 Linux</th><th>VM3 Windows</th></tr>
<tr><th>VM1 Master</th><th>VM2 Noeud</th><td rowspan="4">Panneau de config > Réseau et Internet > Centre Réseau et partage > Onglet de gauche 'Modifier les paramètres de la carte'
cliquer sur l'interface utilisé par virtualbox puis Propriétés > Protocole IPv4 > adresse statique :
192.168.x.30
Sous-réseau 255.255.255.0</td></tr>
<tr><td colspan="2">nano /etc/network/interfaces
auto <nom_interface> iface <nom> inet static</td></tr>
<tr><td>address 192.168.x.10
netmask 255.255.255.0</td><td>address 192.168.x.20
netmask 255.255.255.0</td></tr>
<tr><td colspan="2">Ctrl+X puis Y
systemctl restart networking
ou service networking restart</td></tr>
</table>

Vérification : pinger les machines depuis l'hôte puis entre elles

Avertissement : [
Sous Windows (hôte ou VM), il peut y avoir des règles de pare-feu bloquant les ping.
-> ouvrir 'wf.msc' (menu Démarrer)
-> activer les règles suivantes pour les profils Privé ET Public :
onglet 'Règles de trafic entrant'
Partage de fichiers et imprimantes (demande écho – Trafic entrant ICMPv4)
onglet 'Règles de trafic sortant'
Partage de fichiers et imprimantes (demande écho – Trafic sortant ICMPv4)
]

Configuration de MariaDB sur le serveur Master

Par défaut les connexions à distance sont bloquées. Pour les autoriser :
nano /etc/mysql/mariadb.conf.d/50-server.cnf
modifier : '*bind-address = 127.0.0.1'* (connexions locales uniquement) en '*bind-addresss = 0.0.0.0'*
puis redémarrer : *systemctl restart mariadb*

Création de la base de données et des tables via un script sql :
mysql -u root -p < crea_bdd.sql
connexion à la base : *mariadb -u toto -ptoto tor_sondag ;*
Vérification : *show tables ;*

⇨ L'environnement est maintenant prêt pour les tests de communication.

II- Utilisation

Ordre logique :

1. démarrage du Master en écoute
2. démarrage de l'interface graphique de supervision du Master
3. démarrage des routeurs qui s'enregistrent auprès du Master
4. démarrage du client B en écoute
5. démarrage de l'interface graphique du client A
6. envoi d'un message test

Ordre sur les VM :

1. VM Master :

python3 master.py

2. VM GUI

python master_gui.py

3. VM Nœud

python3 noeud.py --role routeur --port 5001
python3 noeud.py --role routeur --port 5002
python3 noeud.py --role routeur --port 5003

4. VM Nœud

python noeud.py --role clientb --port 6000

5. VM GUI

python client_gui.py
puis envoi d'un message test

Observations :

logs de la construction oignon du message chiffré par A

réception du message par B

forward du payload chiffré pour les routeurs sauf le dernier saut qui forward le message en clair

logs + enregistrement des routeurs dans la base de données visibles après actualisation dans l’interface graphique du master