

Mini-labo 2 : Routeur/firewall :

Objectif :

- Séparer deux sous-réseaux (Lan1 et Lan2).
- Faire passer le trafic par une VM routeur/firewall.
- Mettre en place des règles pour autoriser, bloquer ou filtrer certains flux.

Prérequis :

- Réalisation du projet Mini-labo.
- Trois VM sous Linux (Debian ou Ubuntu).
- La troisième VM doit avoir deux cartes réseaux.

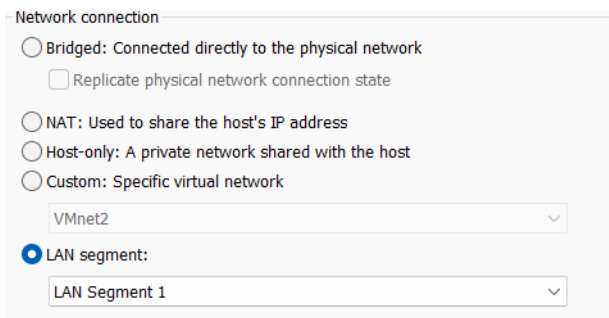
Rappel :

1- Réseau LAN :

Un réseau LAN est un réseau privé. Il n'est pas connecté à Internet.

Passons nos deux VM sur deux différent LAN.

VM1 :



Network connection

☐ Bridged: Connected directly to the physical network

☐ Replicate physical network connection state

☐ NAT: Used to share the host's IP address

☐ Host-only: A private network shared with the host

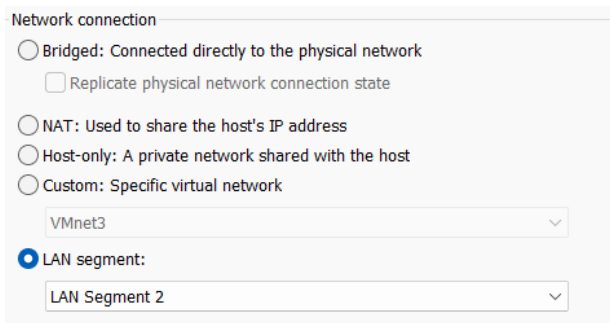
☐ Custom: Specific virtual network

VMnet2

☒ LAN segment:

LAN Segment 1

VM2 :



Network connection

☐ Bridged: Connected directly to the physical network

☐ Replicate physical network connection state

☐ NAT: Used to share the host's IP address

☐ Host-only: A private network shared with the host

☐ Custom: Specific virtual network

VMnet3

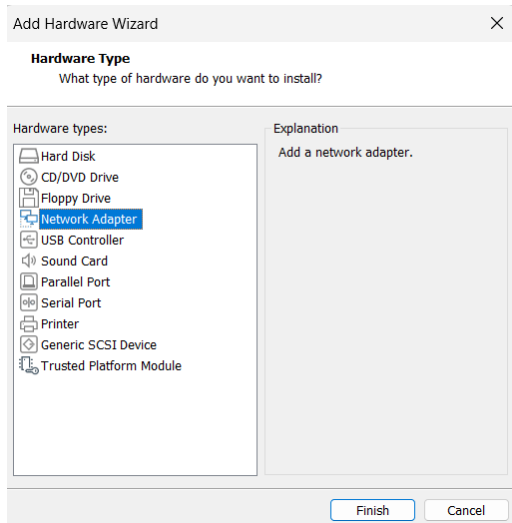
☒ LAN segment:

LAN Segment 2

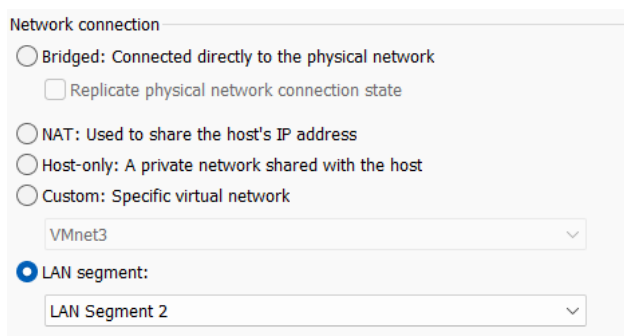
2- Ajout d'une carte réseau :

La troisième VM doit avoir deux cartes réseaux pour pouvoir communiquer avec les deux VM qui sont sur deux sous-réseaux différents.

L'adresse IP des passerelles pour les deux sous-réseaux sont les mêmes.



Une fois la deuxième carte réseau installée nous devons maintenant connecter chaque carte réseau au Lan des deux autres VM.



3- Droits sudo :

On donne les droits sudo à notre utilisateur.

```
user@Mini-labo2-firewall:~$ su -  
Mot de passe :  
root@Mini-labo2-firewall:~# visudo
```

```
# User privilege specification  
root    ALL=(ALL:ALL) ALL  
user    ALL=(ALL:ALL) ALL
```

Etapes :

1- Activation du routage IP :

`Net.ipv4.ip_forward=1`, est le paramètre du noyau qui permet d'indiquer si Linux doit autoriser ou non le routage des adresses IPv4. Avec la valeur 1 on indique que le routage doit être activé.

```
user@Mini-labo2-firewall:~$ sudo sysctl -w net.ipv4.ip_forward=1
[sudo] Mot de passe de user :
net.ipv4.ip_forward = 1
```

On peut aussi le configurer de façon permanente.

```
user@Mini-labo2-firewall:~$ su -
Mot de passe :
root@Mini-labo2-firewall:~# nano /etc/sysctl.conf
root@Mini-labo2-firewall:~#
```

```
# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
# Enabling this option disables Stateless Address Autoconfiguration
# based on Router Advertisements for this host
net.ipv6.conf.all.forwarding=1

#####
# Additional settings - these settings can improve the network
# security of the host and prevent against some network attacks
# including spoofing attacks and man in the middle attacks through
# redirection. Some network environments, however, require that these
# settings are disabled so review and enable them as needed.
#
# Do not accept ICMP redirects (prevent MITM attacks)
net.ipv4.conf.all.accept_redirects = 0
net.ipv6.conf.all.accept_redirects = 0
# _or_
# Accept ICMP redirects only for gateways listed in our default
# gateway list (enabled by default)
net.ipv4.conf.all.secure_redirects = 1
```

2- Configuration du firewall :

Pour configurer le firewall, nous allons utiliser `nftables`.

On commence par vérifier que `nftables` est bien installé.

```
user@Mini-labo2-firewall:~$ sudo systemctl status nftables
○ nftables.service - nftables
   Loaded: loaded (/lib/systemd/system/nftables.service; disabled; preset: enabled)
   Active: inactive (dead)
     Docs: man:nft(8)
           http://wiki.nftables.org
```

On active le service.

```
user@Mini-labo2-firewall:~$ sudo systemctl enable nftables
Created symlink /etc/systemd/system/sysinit.target.wants/nftables.service → /lib/systemd/system/nftables.service.
```

On démarre le service puis on vérifie qu'il est bien activé.

```
user@Mini-labo2-firewall:~$ sudo systemctl start nftables
user@Mini-labo2-firewall:~$ sudo systemctl status nftables
• nftables.service - nftables
   Loaded: loaded (/lib/systemd/system/nftables.service; enabled; preset: enabled)
   Active: active (exited) since Thu 2025-10-02 14:35:32 CEST; 9s ago
     Docs: man:nft(8)
           http://wiki.nftables.org
   Process: 2537 ExecStart=/usr/sbin/nft -f /etc/nftables.conf (code=exited, status=0/SUCCESS)
    Main PID: 2537 (code=exited, status=0/SUCCESS)
       CPU: 22ms

oct. 02 14:35:32 Mini-labo2-firewall systemd[1]: Starting nftables.service - nftables...
oct. 02 14:35:32 Mini-labo2-firewall systemd[1]: Finished nftables.service - nftables.
```

Maintenant nous allons créer un fichier nommé **nftables.conf**. Ce fichier va contenir les règles de notre firewall. Nous allons autoriser le **ping** et le **ssh** mais bloqué le trafic **http**.

```
user@Mini-labo2-firewall:~$ su -
Mot de passe :
root@Mini-labo2-firewall:~# sudo nano /etc/nftables.conf
root@Mini-labo2-firewall:~#
```

```
#!/usr/sbin/nft -f

flush ruleset

table inet filter {
    chain input {
        type filter hook input priority filter;

        #Autorisation loopback (localhost)
        iif "lo" accept

        #Autorisation des connexions établies
        ct state established,related accept

        #Autorisation du protocole ICMP (ping)
        ip protocol icmp accept

        #Autorisation de SSH (port 22)
        tcp dport 22 accept

        #Blocage de HTTP (port 80)
        tcp dport 80 drop

        #Blocage du reste
        drop
    }

    chain forward {
        type filter hook forward priority filter;

        policy accept;
    }

    chain output {
        type filter hook output priority filter;

        policy accept;
    }
}
```

nft -f /etc/nftables.conf on peut utiliser cette commande pour recharger la configuration manuellement. Cela permet de s'assurer que les changements sont bien appliqués.

Maintenant nous redémarrons le service.

```
user@Mini-labo2-firewall:~$ sudo systemctl restart nftables
[sudo] Mot de passe de user :
```

Ensuite, on vérifie les règles actives.

```
user@Mini-labo2-firewall:~$ sudo nft list ruleset
table inet filter {
    chain input {
        type filter hook input priority filter; policy accept;
        iif "lo" accept
        ct state established,related accept
        ip protocol icmp accept
        tcp dport 22 accept
        tcp dport 80 drop
        drop
    }

    chain forward {
        type filter hook forward priority filter; policy accept;
    }

    chain output {
        type filter hook output priority filter; policy accept;
    }
}
```

3- Adresses IP :

Nous allons donner deux adresses IP à notre VM routeur/firewall, une pour chaque carte réseau.

Adresse IP du routeur/firewall pour la VM1 :

```
user@Mini-labo2-firewall:~$ sudo ip addr add 192.168.10.1/24 dev ens33
[sudo] Mot de passe de user :
user@Mini-labo2-firewall:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:f0:72:43 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.10.1/24 scope global ens33
        valid_lft forever preferred_lft forever
3: ens36: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:f0:72:4d brd ff:ff:ff:ff:ff:ff
    altname enp2s4
```

```
user@Mini-labo2-firewall:~$ ip route
192.168.10.0/24 dev ens33 proto kernel scope link src 192.168.10.1
```

Adresse IP de la VM1 :

```
user@Mini-labo-VM1:~$ sudo ip addr add 192.168.10.10/24 dev ens33
[sudo] Mot de passe de user :
user@Mini-labo-VM1:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:42:7d:f1 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.10.10/24 scope global ens33
        valid_lft forever preferred_lft forever
```

```
user@Mini-labo-VM1:~$ sudo ip route add default via 192.168.10.1
user@Mini-labo-VM1:~$ ip route
default via 192.168.10.1 dev ens33
192.168.10.0/24 dev ens33 proto kernel scope link src 192.168.10.10
```

Adresse IP du routeur/firewall pour la VM2 :

```
user@Mini-labo2-firewall:~$ sudo ip addr add 192.168.20.1/24 dev ens36
user@Mini-labo2-firewall:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:f0:72:43 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.10.1/24 scope global ens33
        valid_lft forever preferred_lft forever
3: ens36: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:f0:72:4d brd ff:ff:ff:ff:ff:ff
    altname enp2s4
    inet 192.168.20.1/24 scope global ens36
        valid_lft forever preferred_lft forever
```

```
user@Mini-labo2-firewall:~$ sudo ip addr add 192.168.20.1/24 dev ens36
user@Mini-labo2-firewall:~$ ip route
192.168.10.0/24 dev ens33 proto kernel scope link src 192.168.10.1
192.168.20.0/24 dev ens36 proto kernel scope link src 192.168.20.1
```

Adresse IP de la VM2 :

```
user@Mini-labo-VM2:~$ sudo ip addr add 192.168.20.10/24 dev ens33
[sudo] Mot de passe de user :
user@Mini-labo-VM2:~$ ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:0c:29:bb:40:18 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.20.10/24 scope global ens33
        valid_lft forever preferred_lft forever
    inet6 fe80::20c:29ff:febb:4018/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

```
user@Mini-labo-VM2:~$ sudo ip route add default via 192.168.20.1
user@Mini-labo-VM2:~$ ip route
default via 192.168.20.1 dev ens33
192.168.20.0/24 dev ens33 proto kernel scope link src 192.168.20.10
```

Résultat :

1- Ping :

Commençons par tester Ping depuis nos deux VM.

VM1 :

```
user@Mini-labo-VM1:~$ ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1) 56(84) bytes of data.
64 bytes from 192.168.10.1: icmp_seq=1 ttl=64 time=0.715 ms
64 bytes from 192.168.10.1: icmp_seq=2 ttl=64 time=1.85 ms
64 bytes from 192.168.10.1: icmp_seq=3 ttl=64 time=2.13 ms
64 bytes from 192.168.10.1: icmp_seq=4 ttl=64 time=1.85 ms
^C
--- 192.168.10.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3039ms
rtt min/avg/max/mdev = 0.715/1.634/2.125/0.542 ms
user@Mini-labo-VM1:~$ ping 192.168.20.1
PING 192.168.20.1 (192.168.20.1) 56(84) bytes of data.
64 bytes from 192.168.20.1: icmp_seq=1 ttl=64 time=0.476 ms
64 bytes from 192.168.20.1: icmp_seq=2 ttl=64 time=1.57 ms
64 bytes from 192.168.20.1: icmp_seq=3 ttl=64 time=1.36 ms
64 bytes from 192.168.20.1: icmp_seq=4 ttl=64 time=1.12 ms
^C
--- 192.168.20.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3030ms
rtt min/avg/max/mdev = 0.476/1.129/1.565/0.408 ms
user@Mini-labo-VM1:~$ ping 192.168.20.10
PING 192.168.20.10 (192.168.20.10) 56(84) bytes of data.
64 bytes from 192.168.20.10: icmp_seq=1 ttl=63 time=1.19 ms
64 bytes from 192.168.20.10: icmp_seq=2 ttl=63 time=2.43 ms
64 bytes from 192.168.20.10: icmp_seq=3 ttl=63 time=1.92 ms
64 bytes from 192.168.20.10: icmp_seq=4 ttl=63 time=3.05 ms
^C
--- 192.168.20.10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.191/2.147/3.046/0.680 ms
```

VM2 :

```
user@Mini-labo-VM2:~$ ping 192.168.20.1
PING 192.168.20.1 (192.168.20.1) 56(84) bytes of data.
64 bytes from 192.168.20.1: icmp_seq=1 ttl=64 time=0.646 ms
64 bytes from 192.168.20.1: icmp_seq=2 ttl=64 time=3.18 ms
64 bytes from 192.168.20.1: icmp_seq=3 ttl=64 time=2.72 ms
64 bytes from 192.168.20.1: icmp_seq=4 ttl=64 time=2.86 ms
^C
--- 192.168.20.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3006ms
rtt min/avg/max/mdev = 0.646/2.350/3.176/0.997 ms
user@Mini-labo-VM2:~$ ping 192.168.10.1
PING 192.168.10.1 (192.168.10.1) 56(84) bytes of data.
64 bytes from 192.168.10.1: icmp_seq=1 ttl=64 time=0.719 ms
64 bytes from 192.168.10.1: icmp_seq=2 ttl=64 time=1.99 ms
64 bytes from 192.168.10.1: icmp_seq=3 ttl=64 time=1.88 ms
64 bytes from 192.168.10.1: icmp_seq=4 ttl=64 time=1.93 ms
^C
--- 192.168.10.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3019ms
rtt min/avg/max/mdev = 0.719/1.630/1.991/0.527 ms
user@Mini-labo-VM2:~$ ping 192.168.10.10
PING 192.168.10.10 (192.168.10.10) 56(84) bytes of data.
64 bytes from 192.168.10.10: icmp_seq=1 ttl=63 time=1.36 ms
64 bytes from 192.168.10.10: icmp_seq=2 ttl=63 time=3.34 ms
64 bytes from 192.168.10.10: icmp_seq=3 ttl=63 time=3.89 ms
64 bytes from 192.168.10.10: icmp_seq=4 ttl=63 time=5.01 ms
^C
--- 192.168.10.10 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.362/3.398/5.010/1.321 ms
```

2- SSH :

a- Installation de SSH sur les VM :

On passe les cartes réseaux de nos deux machines sur NAT. Une fois l'installation faite on repasse en Lan segment 1 pour VM1 et Lan segment 2 pour VM2.

VM1 :

```
user@Mini-labo-VM1:~$ sudo apt update
[sudo] Mot de passe de user :
Atteint :1 http://security.debian.org/debian-security bookworm-security InRelease
Atteint :2 http://deb.debian.org/debian bookworm InRelease
Atteint :3 http://deb.debian.org/debian bookworm-updates InRelease
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
6 paquets peuvent être mis à jour. Exécutez « apt list --upgradable » pour les voir.
user@Mini-labo-VM1:~$ sudo apt install openssh-server -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  openssh-sftp-server runit-helper
```

VM2 :

```
user@Mini-labo-VM2:~$ sudo apt update
[sudo] Mot de passe de user :
Atteint :1 http://security.debian.org/debian-security bookworm-security InRelease
Atteint :2 http://deb.debian.org/debian bookworm InRelease
Atteint :3 http://deb.debian.org/debian bookworm-updates InRelease
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
6 paquets peuvent être mis à jour. Exécutez « apt list --upgradable » pour les voir.
user@Mini-labo-VM2:~$ sudo apt install openssh-server -y
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
Les paquets supplémentaires suivants seront installés :
  openssh-sftp-server runit-helper
```

b- Test :

VM1 :


```
user@Mini-labo-VM1:~$ ssh user@192.168.20.10
The authenticity of host '192.168.20.10 (192.168.20.10)' can't be established.
ED25519 key fingerprint is SHA256:e+UY9Gt6ZxNec5wxcE4winwJnrt+ouN36jZwskLEeiQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.20.10' (ED25519) to the list of known hosts.
user@192.168.20.10's password:
Linux Mini-labo-VM2 6.1.0-39-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.148-1 (2025-08-26) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
user@Mini-labo-VM2:~$ ping 192.168.10.10
PING 192.168.10.10 (192.168.10.10) 56(84) bytes of data.
64 bytes from 192.168.10.10: icmp_seq=1 ttl=63 time=0.956 ms
64 bytes from 192.168.10.10: icmp_seq=2 ttl=63 time=1.68 ms
64 bytes from 192.168.10.10: icmp_seq=3 ttl=63 time=1.87 ms
^C
--- 192.168.10.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.956/1.501/1.872/0.393 ms
```

VM2 :

```
user@Mini-labo-VM2:~$ ssh user@192.168.10.10
The authenticity of host '192.168.10.10 (192.168.10.10)' can't be established.
ED25519 key fingerprint is SHA256:z0jqY/rac/L3fNsp9B3mpzs16JfIFLuaG+AoOo8Lktg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.10.10' (ED25519) to the list of known hosts.
user@192.168.10.10's password:
Linux Mini-labo-VM1 6.1.0-40-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.153-1 (2025-09-20) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
```

3- HTTP :

a- Installation de curl :

On passe nos deux VM en NAT pour pouvoir installer curl, une fois l'installation faite on repasse en Lan segment 1 pour VM1 et en Lan segment 2 pour VM2.

VM1 :

```
user@Mini-labo-VM1:~$ sudo apt install curl
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
Lecture des informations d'état... Fait
```

VM2 :

```
user@Mini-labo-VM2:~$ sudo apt install curl
[sudo] Mot de passe de user :
Lecture des listes de paquets... Fait
Construction de l'arbre des dépendances... Fait
```

b- Test :

On finit avec le test du protocole HTTP pour vérifier que ce dernier est bien bloqué par notre firewall.

VM1 :

```
user@Mini-labo-VM1:~$ curl http://192.168.20.10
curl: (7) Failed to connect to 192.168.20.10 port 80 after 1 ms: Couldn't connect to server
user@Mini-labo-VM1:~$
```

VM2 :

```
user@Mini-labo-VM2:~$ curl http://192.168.10.10
curl: (7) Failed to connect to 192.168.10.10 port 80 after 2 ms: Couldn't connect to server
user@Mini-labo-VM2:~$
```

Conclusion et apprentissage :

1- Conclusion :

Ce projet a permis la mise en place de deux sous-réseaux distincts reliés par une VM jouant le rôle de routeur/firewall.

Nftables a été utilisé pour configurer le firewall afin de contrôler et filtrer les flux réseau entre les deux LAN. Les règles tester ont permis de constater leur impact sur la communication entre les VM.

Ce projet démontre l'importance du rôle d'un firewall dans la sécurisation et la gestion du trafic réseau. Avec cette architecture simple, le fonctionnement de base d'un réseau d'entreprise a été reproduite.

2- Apprentissage :

- Activation du routage IP sur une machine Linux pour relier deux sous-réseaux.
- Découverte et utilisation de **nftables** pour :
- Autorisation certains flux (ICMP, SSH)
- Blocage d'autres flux (HTTP)
- Visualisation et administration des règles de firewall
- Vérification pratique du filtrage avec des outils simples comme **ping**, **ssh** et **curl**.