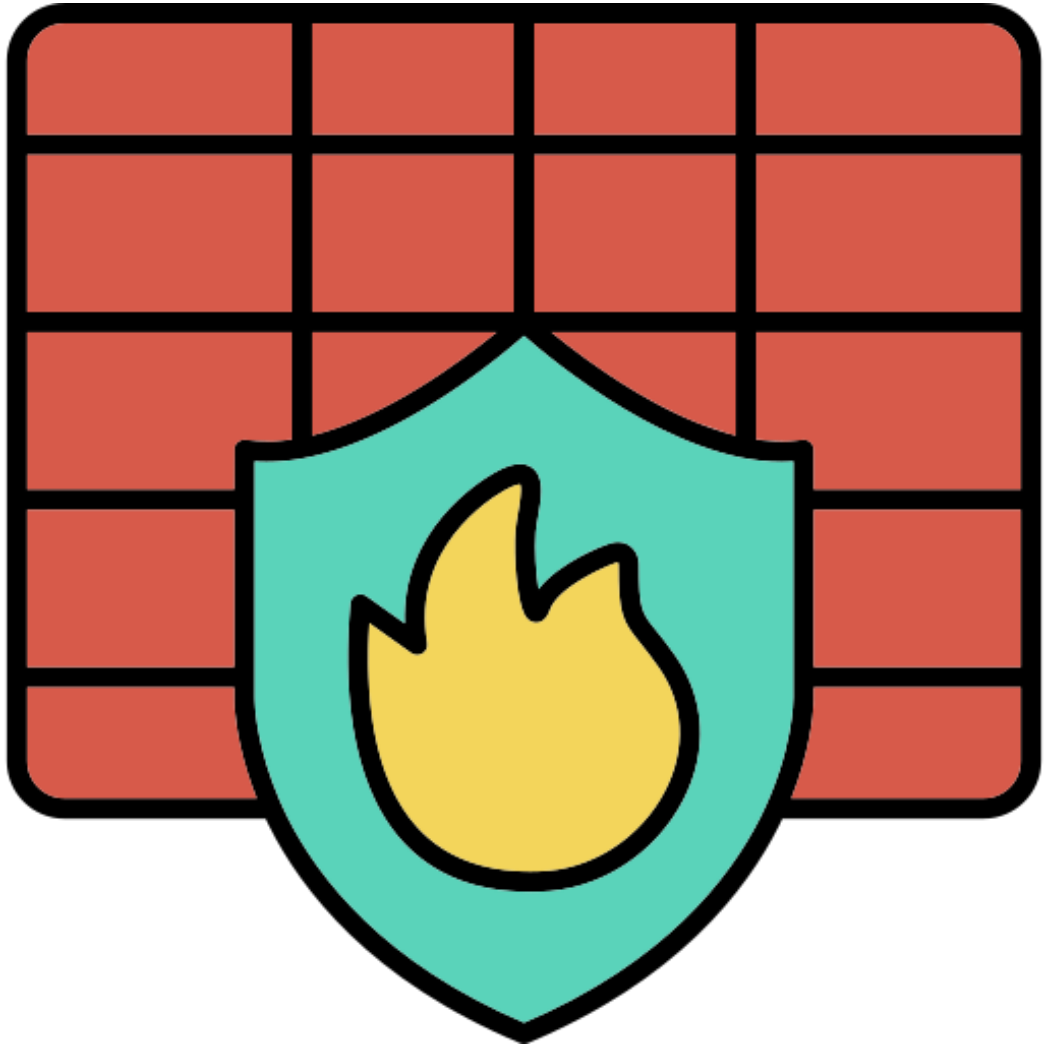
TP - IPtables



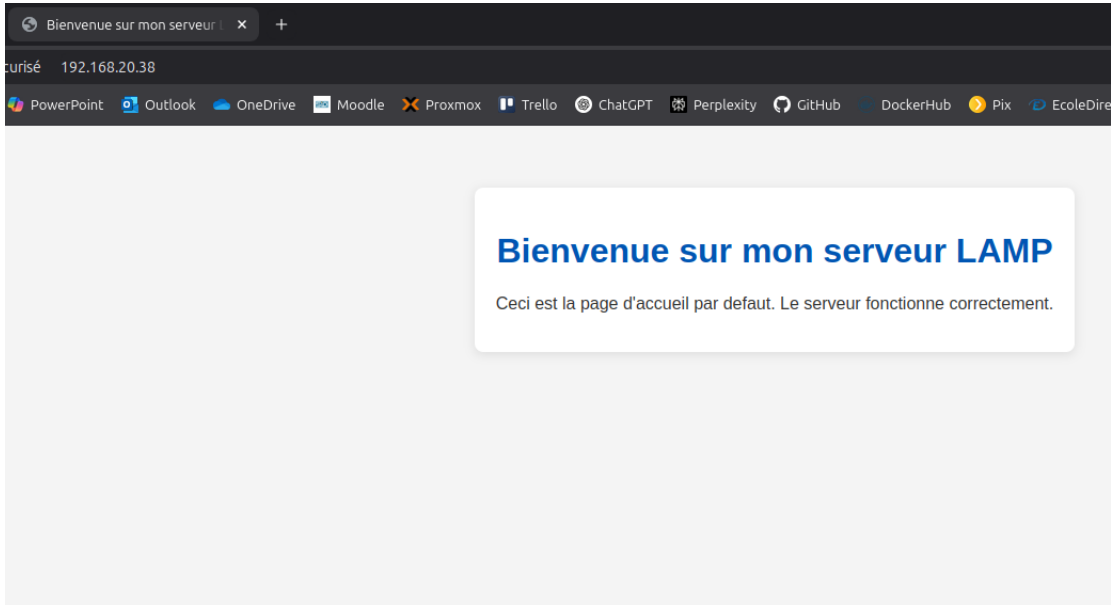
Présentation Firewall et IPtables

Le filtrage réseau par firewall (pare-feu) consiste a examiner les paquets qui entrent sur le réseau interne et une décision est prise sur le traitement à appliquer à ce dernier.

AVANTAGES	INCONVENIENTS
- Filtrage flexible qui permet un contrôle précis sur les paquets grâce à des règles détaillées.	- Complexité de configuration avec la création et la gestion des règles pouvant poser problèmes aux débutants.
- Gratuit et OpenSoruce	- Exposition aux erreurs humaines en cas de mauvaise configuration.
- Fiable et robuste car intégré au noyau Linux cela offre une performance optimale	- Limitée aux protocoles connus, difficultés avec certains protocoles personnalisés ou complexe.
- Support NAT qui cache les ressources internes et facilite le routage.	- Maintenance régulières et nécessaires au niveau de la gestion des règles et leurs mises à jour.

Mise en place de notre infrastructure IPtable

- Nous allons installer notre serveur web Apache2 : **apt-get install apache2**
- Puis nous rentrons l'IP de notre machine afin de voir que notre serveur Web est bien actif
- Nous installons ensuite iptables : **apt-get install iptables**
- Puis nous testons avec : **iptables –L**




```
root@Apache:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@Apache:~#
```

Empecher le ping sur l'adresse de loopback

- Création d'une machine personnelle :
- Prise en compte de la chaine dans les logs
- Prise en compte de l'action de la chaine
- Écriture de la chaine
- Vérification avec iptables –L

```
root@Apache:~# iptables -N Baptiste
root@Apache:~#
```

```
root@Apache:~# iptables -A Baptiste -j LOG
root@Apache:~# iptables -A Baptiste -j DROP
root@Apache:~# iptables -A INPUT -p icmp -s 127.0.0.1 -j Baptiste
root@Apache:~#
```

```
root@Apache:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Baptiste   icmp --  localhost            anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain Baptiste (1 references)
target     prot opt source               destination
LOG        all  --  anywhere             anywhere             LOG level warn
DROP       all  --  anywhere             anywhere
root@Apache:~#
```

Empêcher le ping du client vers le serveur

- Empêcher le ping du poste serveur sur le poste client :

```
root@Apache:~# iptables -A INPUT -p icmp --icmp-type echo-request -s 192.168.20.119 -j DROP
root@Apache:~#
```

```
root@Apache:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Baptiste   icmp --  localhost            anywhere
DROP       icmp --  192.168.20.119       anywhere             icmp echo-request
```

```
bapt@PC-Baptiste:~$ ping 192.168.20.38
PING 192.168.20.38 (192.168.20.38) 56(84) bytes of data.
^C
--- 192.168.20.38 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6132ms
```

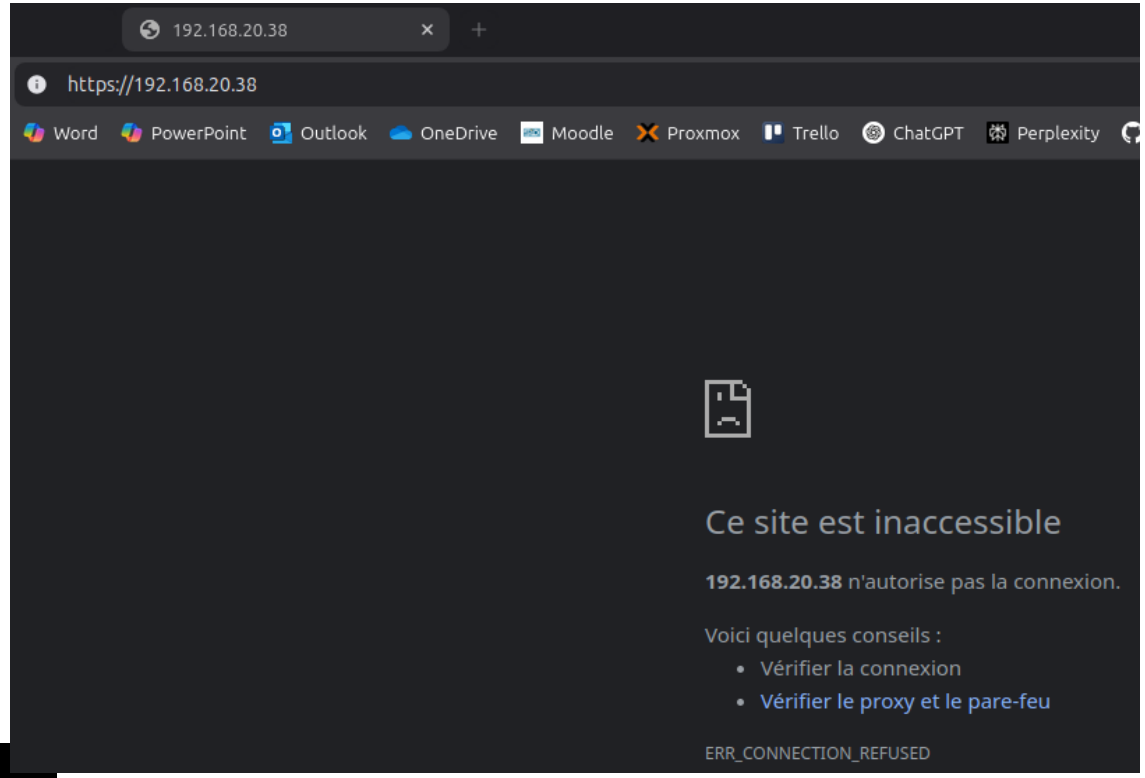
Permettre l'accès au serveur web uniquement en http

- Pour cela nous allons devoir accorder l'accès uniquement sur le port 80.
- Nous utilisons la commande :

```
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

- Puis nous vérifions avec iptables –L, nous voyons dans la colonne target que le port 80 est bien prit en compte.

```
root@Apache:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Baptiste   icmp --  localhost            anywhere
DROP       icmp --  192.168.20.119       anywhere             icmp echo-request
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
```



Interdire l'accès à une seule interface

- Pour cela nous ajoutons une 2e IP à notre carte réseau : **nano /etc/network/interface**
- Avec un ip a, nous voyons la deuxième IP qui nous est attribué.
- Ensuite nous allons donc interdire cette IP.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp

auto eth0:0
iface eth0:0 inet dhcp
```

```
root@Apache:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default q
    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: eth0@if664: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP gr
1000
    link/ether bc:24:11:b3:20:5c brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.40.10.1/24 brd 10.40.10.255 scope global eth0:0
       valid_lft forever preferred_lft forever
    inet 192.168.20.38/24 brd 192.168.20.255 scope global dynamic eth0
       valid_lft 7175sec preferred_lft 7175sec
    inet 192.168.20.89/24 brd 192.168.20.255 scope global secondary dynamic eth0:0
```

```
root@Apache:~# iptables -A INPUT -s 192.168.20.89 -j DROP
```

Interdire l'accès d'une IP

- Ensuite nous vérifions cela par un iptables –L
- Puis nous mettons en commentaire notre carte réseau principale afin que l'autre prenne le relais et vérifier si notre règle est valide.

```
root@Apache:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source
Baptiste   icmp --  localhost
DROP       icmp --  192.168.20.119
ACCEPT     tcp  --  anywhere
DROP       all  --  192.168.20.89
```

```
auto lo
iface lo inet loopback

#auto eth0
#iface eth0 inet dhcp

auto eth0:0
iface eth0:0 inet dhcp
```

Refuser les connexions telnet

- Les connexions se font sur le port 23, il faut donc rejeter ces connexions.
- Nous voyons donc que cela ne répond pas, la règle est correctement appliquée.

```
root@Apache:~# iptables -A INPUT -p tcp --dport 23 -j DROP
```

```
root@Apache:~# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
Baptiste   icmp --  localhost            anywhere
DROP       icmp --  192.168.20.119       anywhere             icmp echo-request
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
DROP       all  --  192.168.20.89        anywhere
DROP       tcp  --  anywhere             anywhere             tcp dpt:telnet
```

```
bapt@PC-Baptiste:~$ telnet 192.168.20.89 23
Trying 192.168.20.89...
```

Mise en place de différentes règles

- Le poste client ne peut que consulter notre serveur Web : La première règle accepte que le port 80 et la deuxième rejette tout le reste

```
root@Apache:~# iptables -A INPUT -p tcp -s 192.168.20.37 --dport 80 -j ACCEPT
root@Apache:~# iptables -A INPUT -s 192.168.20.37 -j DROP
```

- Le poste client ne peut pas être pingué :

```
root@Apache:~# iptables -A OUTPUT -s 192.168.20.89 -j DROP
```

- Le poste client ne peut pas pinger le serveur :

```
root@Apache:~# iptables -A INPUT -s 192.168.20.37 -j DROP
```

- Notre serveur web est uniquement serveur web (on autorise que le port 80) :

```
root@Apache:~# iptables -A INPUT -p tcp -s 192.168.20.37 --dport 80 -j ACCEPT
```

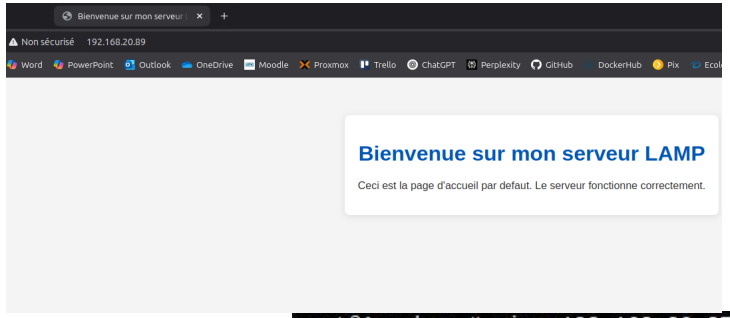
- Seules les connexions établies sont acceptées :

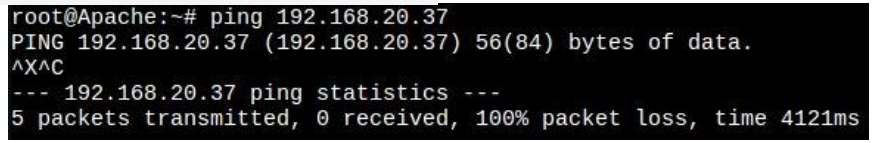
```
root@Apache:~# iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
```

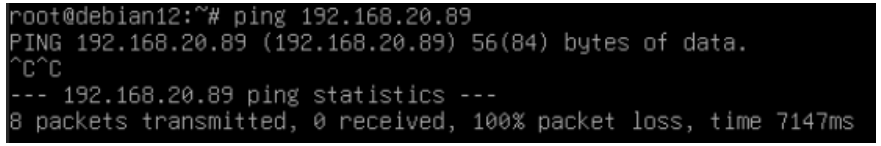
Mise en place de différentes règles

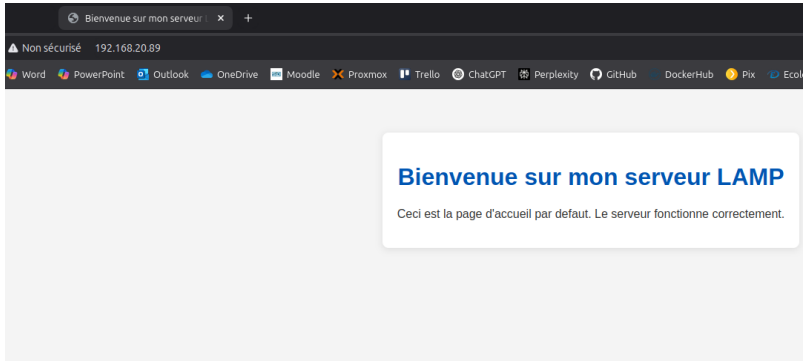
Tests

- Le poste client ne peut que consulter notre serveur Web :
- Le poste client ne peut pas être pingué :
- Le poste client ne peut pas pinger le serveur :
- Notre serveur web est uniquement serveur web :
- Seules les connexions établies sont acceptées :









TP 2 – IPtables

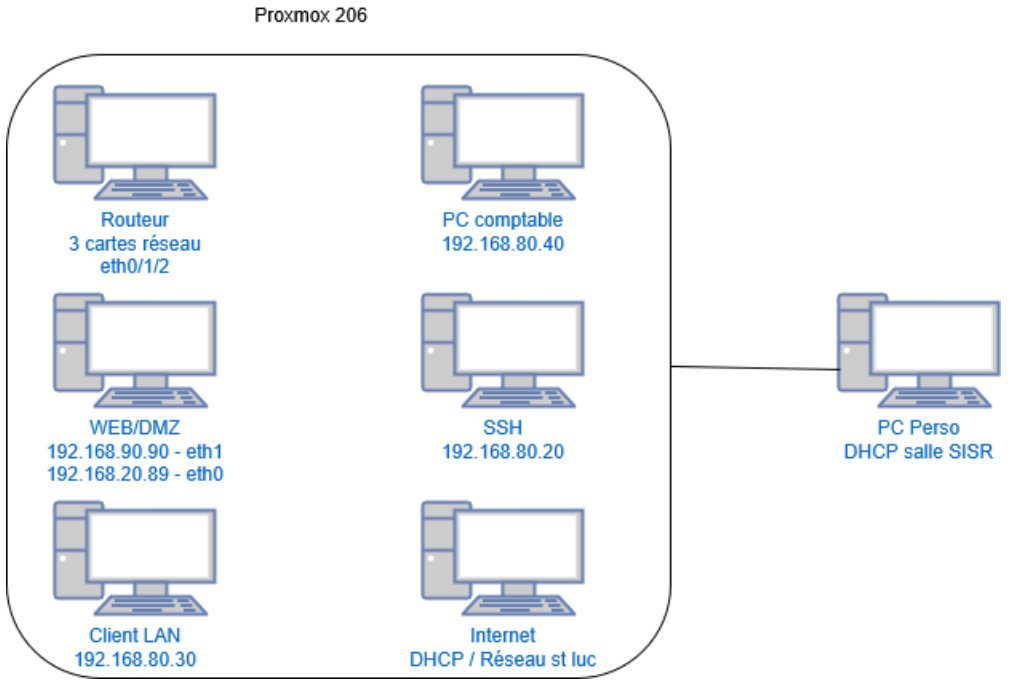
- Ajout de 2 cartes réseaux supplémentaires sur notre machine IPtables
- La carte eth0 sera notre WAN
- La carte eth1 sera notre DMZ
- La carte eth2 sera notre LAN

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet dhcp

auto eth1
iface eth1 inet static
        address 192.168.90.10
        netmask 255.255.255.0

auto eth2
iface eth2 inet static
        address 192.168.80.10
        netmask 255.255.255.0
```



Gestion de la DMZ

- Objectif 1 : Notre DMZ est accessible de partout, Internet et notre réseau local :

```
root@Apache:~# iptables -A FORWARD -i eth+ -o eth1 -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT
```

- Eth+ concerne toutes les cartes réseau et eth1 est notre DMZ sur laquelle nous allons appliquer la règle.
- On test avec un ping d'une machine cliente :

```
root@debian12:~# ping 192.168.90.10
PING 192.168.90.10 (192.168.90.10) 56(84) bytes of data.
64 bytes from 192.168.90.10: icmp_seq=1 ttl=64 time=0.148 ms
64 bytes from 192.168.90.10: icmp_seq=2 ttl=64 time=0.166 ms
^C
--- 192.168.90.10 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1005ms
rtt min/avg/max/mdev = 0.148/0.157/0.166/0.009 ms
```

```
root@Apache:~# ping 192.168.90.10
PING 192.168.90.10 (192.168.90.10) 56(84) bytes of data.
64 bytes from 192.168.90.10: icmp_seq=1 ttl=64 time=0.037 ms
64 bytes from 192.168.90.10: icmp_seq=2 ttl=64 time=0.030 ms
64 bytes from 192.168.90.10: icmp_seq=3 ttl=64 time=0.029 ms
^C
--- 192.168.90.10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2072ms
rtt min/avg/max/mdev = 0.029/0.032/0.037/0.003 ms
root@Apache:~#
```

Gestion de la DMZ

- Objectif 2 : Nous interdisons les pings depuis l'extérieur pour éviter les attaque DDOS :

```
root@Apache:~# iptables -A INPUT -i eth1 -p icmp --icmp-type echo-request -j DROP
```

- Test depuis un client sur le WAN :

```
bapt@PC-Baptiste:~$ ping 192.168.20.89
PING 192.168.20.89 (192.168.20.89) 56(84) bytes of data.
From 192.168.20.67 icmp_seq=1 Destination Host Unreachable
From 192.168.20.67 icmp_seq=2 Destination Host Unreachable
From 192.168.20.67 icmp_seq=3 Destination Host Unreachable
^C
--- 192.168.20.89 ping statistics ---
5 packets transmitted, 0 received, +3 errors, 100% packet loss, time 4103ms
pipe 4
```

Gestion de la DMZ

- Objectif 3 : On accorde le ping du serveur Web depuis le LAN :

```
root@Routeur:~# iptables -A INPUT -i eth2 -p icmp --icmp-type echo-request -j ACCEPT
```

- Test sur le client depuis le LAN :

```
root@Client-LAN:~# ping 192.168.20.148
PING 192.168.20.148 (192.168.20.148) 56(84) bytes of data.
64 bytes from 192.168.20.148: icmp_seq=1 ttl=63 time=0.323 ms
64 bytes from 192.168.20.148: icmp_seq=2 ttl=63 time=0.121 ms
64 bytes from 192.168.20.148: icmp_seq=3 ttl=63 time=0.145 ms
64 bytes from 192.168.20.148: icmp_seq=4 ttl=63 time=0.063 ms
^C
--- 192.168.20.148 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3051ms
```

Gestion de la DMZ

- Objectif 4 : Le serveur SSH peut se connecter aux postes de la DMZ.

```
root@Apache:~# iptables -A OUTPUT -o eth1 -p tcp --dport 22 -j ACCEPT
root@Apache:~# iptables -A INPUT -i eth1 -p tcp --sport 22 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

- La première règle sert à autoriser le trafic SSH vers la DMZ et la deuxième règle permet d'autoriser les réponses entrantes.

- Test de vérification :

```
root@SSH-IPtable:~# ssh root@192.168.90.90
root@192.168.90.90's password:
Linux WEB-DMZ 6.8.4-2-pve #1 SMP PREEMPT_DYNAMIC PMX 6.8.4-2 (2024-04-10T17:36Z)
4

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.
Last login: Fri Mar 21 12:46:00 2025 from 192.168.80.20
root@WEB-DMZ:~#
```

Gestion des postes du LAN

- Objectif 1 : Nous allons autoriser le ping de tous les postes du réseau privé sur l'interface LAN du routeur.

```
root@Routeur:~# iptables -A OUTPUT -o eth0 -p icmp --icmp-type echo-reply -j ACCEPT
```

```
root@Routeur:~# iptables -A INPUT -i eth0 -p icmp --icmp-type echo-request -j ACCEPT
```

```
root@Routeur:~# ip a
1: lo: <LOOPBACK,UP,LOWER_
    link/loopback 00:00:00
    inet 127.0.0.1/8 scope
       valid_lft forever p
    inet6 ::1/128 scope hos
       valid_lft forever p
2: eth0@if579: <BROADCAST,
en 1000
    link/ether bc:24:11:77
    inet 192.168.20.158/24
```

```
root@WEB-DMZ:~# ping 192.168.20.158
PING 192.168.20.158 (192.168.20.158) 56(84) bytes of data.
64 bytes from 192.168.20.158: icmp_seq=1 ttl=64 time=0.209 ms
64 bytes from 192.168.20.158: icmp_seq=2 ttl=64 time=0.073 ms
^C
--- 192.168.20.158 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1063ms
rtt min/avg/max/mdev = 0.073/0.141/0.209/0.068 ms
root@WEB-DMZ:~#
```

```
root@SSH-IPtable:~# ping 192.168.20.158
PING 192.168.20.158 (192.168.20.158) 56(84) bytes of data.
64 bytes from 192.168.20.158: icmp_seq=1 ttl=64 time=0.061 ms
64 bytes from 192.168.20.158: icmp_seq=2 ttl=64 time=0.060 ms
64 bytes from 192.168.20.158: icmp_seq=3 ttl=64 time=0.045 ms
^C
--- 192.168.20.158 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2053ms
rtt min/avg/max/mdev = 0.045/0.055/0.061/0.007 ms
root@SSH-IPtable:~#
```

```
root@Client-LAN:~# ping 192.168.20.158
PING 192.168.20.158 (192.168.20.158) 56(84) bytes of data.
64 bytes from 192.168.20.158: icmp_seq=1 ttl=64 time=0.119 ms
64 bytes from 192.168.20.158: icmp_seq=2 ttl=64 time=0.075 ms
64 bytes from 192.168.20.158: icmp_seq=3 ttl=64 time=0.074 ms
^C
--- 192.168.20.158 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2068ms
rtt min/avg/max/mdev = 0.074/0.089/0.119/0.021 ms
root@Client-LAN:~#
```

Gestion des postes du LAN

- Objectif 2 : nous autorisons le routage des paquets provenant du réseau privé vers la DMZ.

```
root@Routeur:~# iptables -A FORWARD -i eth2 -o eth1 -j ACCEPT
```

- Test :

```
root@Client-LAN:~# ping 192.168.90.90
PING 192.168.90.90 (192.168.90.90) 56(84) bytes of data.
64 bytes from 192.168.90.90: icmp_seq=1 ttl=63 time=0.125 ms
64 bytes from 192.168.90.90: icmp_seq=2 ttl=63 time=0.067 ms
64 bytes from 192.168.90.90: icmp_seq=3 ttl=63 time=0.069 ms
64 bytes from 192.168.90.90: icmp_seq=4 ttl=63 time=0.071 ms
^C
--- 192.168.90.90 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3067ms
rtt min/avg/max/mdev = 0.067/0.083/0.125/0.024 ms
root@Client-LAN:~#
```

Gestion des postes du LAN

- Objectif 3 : nous allons camoufler les adresses IP des postes du réseau LAN qui sortent sur le WAN.

```
root@Routeur:~# iptables -t nat -A POSTROUTING -o eth2 -j MASQUERADE
```

- Test avec Wireshark impossible en raison de règles déjà présentes sur le réseau de la salle, il m’est donc impossible de capturer la trame demandée pour la vérification.

Gestions des postes du LAN

- Objectif 4 : Le poste du comptable ne peut pas aller sur internet, nous allons donc fixer la règle sur son adresse MAC, puisque son IP peut changer contrairement a son adresse MAC.

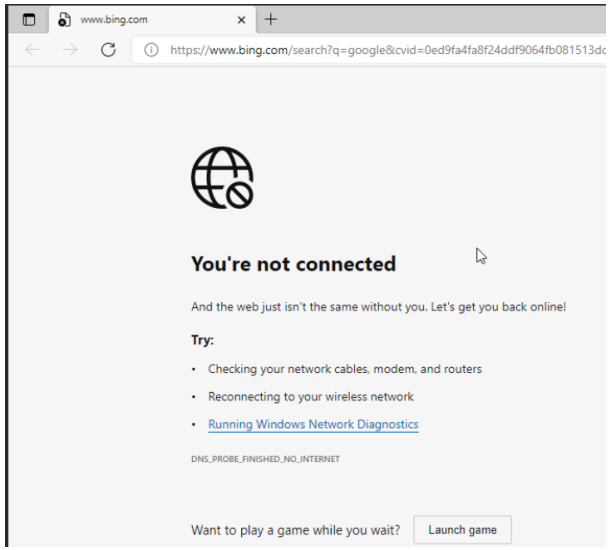
```
root@Routeur:~# iptables -A FORWARD -m mac --mac-source BC:24:11:66:FF:2D -j DROP
```

- Test de vérifications :

```
C:\Users\sio>ping 192.168.20.254

Pinging 192.168.20.254 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.20.254:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

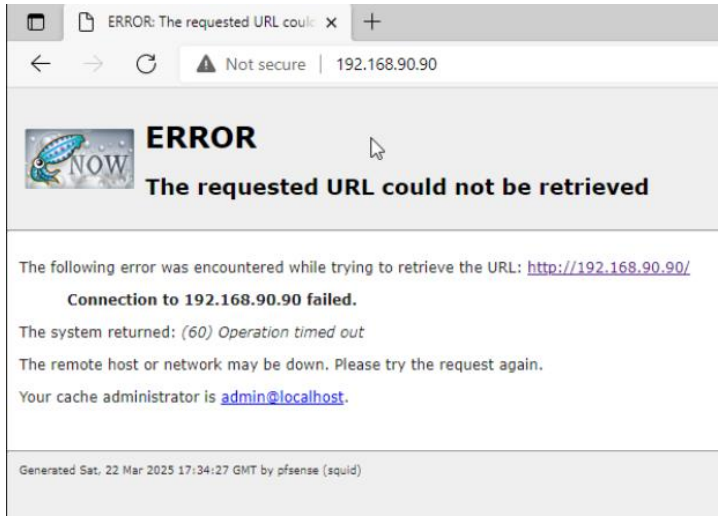


Gestion des postes du LAN

- Objectif 5 : autoriser que les connexions déjà établies

```
root@Routeur:~# iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

- Test sur un client :



Gestion des postes du LAN

- Objectif 6 : ajouter les paquets rejeter dans un fichier « rejeter » dans le répertoire /var/log/

```
root@Routeur:~# iptables -A INPUT -j LOG --log-prefix "PAQUET REJETE: " --log-level 4
```

- Ensuite dans le fichier de conf de rsyslog, nous allons ajouter le message et la redirection

```
root@Routeur:~# cat /var/log/rejeter
:msg,contains,"PAQUET REJETE:" -/var/log/rejeter
& stop
```

- Puis on restart notre service rsyslog : **systemctl restart rsyslog**
- Enfin on ajoute une dernière règle pour rejeter les paquets non autorisés :

```
root@Routeur:~# iptables -A INPUT -j REJECT
```