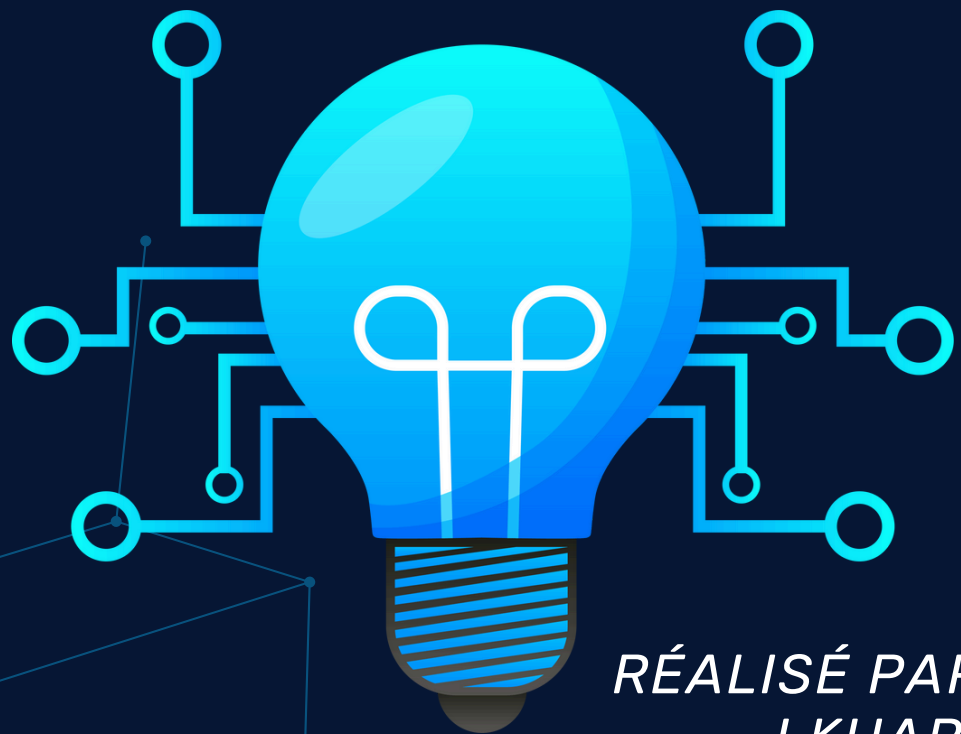
RAPPORT DU PROJET ASSR : PARE-FEU AVEC IPTABLES

Outil : Ubuntu dans VirtualBox

(FOCUS SUR TABLE MANGLE)



RÉALISÉ PAR : IMANE LKHARAT
AYA LAGHNAM
TAHA ADANAN

ENCADRÉS PAR :MME SOUKAINA MIHI

1. Introduction:

Dans le cadre du module Administration des Systèmes et Services Réseaux (ASSR), ce projet a pour objectif la mise en place d'un pare-feu sous Linux à l'aide d'IPTables, avec un accent particulier sur la table mangle.
IPTables est un outil puissant de filtrage des paquets réseau utilisé dans les systèmes Linux. Il fonctionne à travers plusieurs tables, chacune ayant un rôle spécifique :

- La table filter permet de bloquer ou d'autoriser le trafic.
- La table nat sert à la redirection ou la traduction d'adresses IP.
- La table mangle, sujet principal de ce rapport, permet d'effectuer des modifications sur les paquets, comme le marquage, utilisé pour le routage avancé ou la gestion de la qualité de service (QoS).

L'environnement de travail repose sur une machine virtuelle Ubuntu configurée dans VirtualBox/VMware, où toutes les règles ont été testées. Le projet couvre :

- L'installation et la configuration d'IPTables
- La construction du pare-feu
- L'utilisation de la table mangle pour marquer les paquets
- Le contrôle de la connexion à certains ports

Ce rapport documente les étapes techniques suivies et les configurations appliquées, avec des captures d'écran à l'appui pour illustrer les résultats obtenus.

2. Présentation d'IPTables:

2.1. Qu'est-ce qu'IPTables ?

IPTables est un outil en ligne de commande pour configurer les règles du pare-feu intégré au noyau Linux. Il repose sur le framework Netfilter, et permet la gestion des paquets via plusieurs tables :

- **filter** : table par défaut pour autoriser/bloquer les connexions (vue rapidement)
- **nat :** pour la redirection ou la traduction d'adresses (vue rapidement)
- **mangle :** pour modifier les entêtes des paquets ou marquer des paquets (notre table cible)
- **raw :** pour gérer les connexions sans état (rarement utilisée)

3. Installation et préparation de l'environnement:

3.1. Lancer la VM Ubuntu:

- Utilisation de VirtualBox sur système hôte Windows
- Installation d'Ubuntu Desktop ou Server

3.2. Installation d'IPTables:

- **Explications des commandes d'installation :**

apt purge ufw

La commande **apt purge ufw** désinstalle complètement UFW (Uncomplicated Firewall) de ton système, y compris ses fichiers de configuration. Cela permet de libérer de l'espace et d'éviter que UFW n'interfère avec la gestion manuelle de ton pare-feu via iptables.

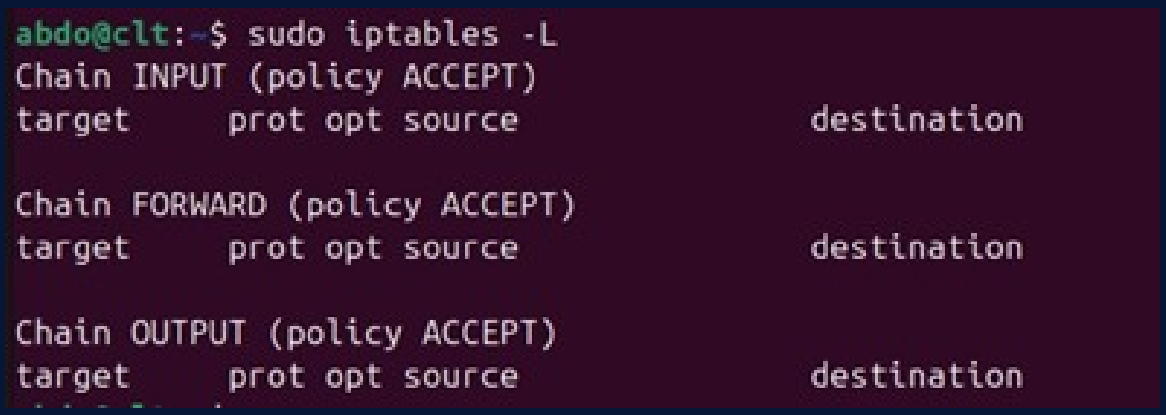
```
abdo@clt:~$ sudo apt purge ufw
[sudo] password for abdo:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages will be REMOVED:
  ufw*
0 upgraded, 0 newly installed, 1 to remove and 91 not upgraded.
After this operation, 869 kB disk space will be freed.
Do you want to continue? [Y/n] y
(Reading database ... 152985 files and directories currently installed.)
Removing ufw (0.36.2-6) ...
Skip stopping firewall: ufw (not enabled)
Processing triggers for man-db (2.12.0-4build2) ...
(Reading database ... 152890 files and directories currently installed.)
Purging configuration files for ufw (0.36.2-6) ...
Processing triggers for rsyslog (8.2312.0-3ubuntu9) ...
```

sudo iptables –L

La commande **sudo iptables -L** permet d'afficher les règles de pare-feu actuellement actives sur ton système. Voici une petite explication :

- **sudo :** Exécute la commande avec des privilèges administrateur.
- **iptables :** Outil utilisé pour configurer le pare-feu sur Linux.
- **-L :** Option qui liste toutes les règles de pare-feu actuelles dans les tables d'iptables.

Donc, **sudo iptables -L** montre les règles de filtrage des paquets actuellement appliquées par iptables.



4. Building (construction du pare-feu)

4.1.Définir la politique par défaut pour le trafic entrant avec iptables:

- **Explications des commandes du building :**

sudo iptables --policy INPUT ACCEPT

Cette commande définit la politique par défaut pour la chaîne INPUT d'iptables. Cela signifie que toutes les connexions entrantes (c'est-à-dire, les paquets entrants) sont acceptées par défaut, sans aucune restriction.

- **INPUT :** C'est la chaîne qui gère le trafic entrant (les paquets venant vers ton serveur).
- **ACCEPT :** Cela signifie que tous les paquets entrants sont autorisés.
- Pour refuser tout le trafic entrant, tu remplaces **ACCEPT** par **DROP :**
  - **DROP :** Cela rejette tous les paquets entrants sans donner de réponse.

4.2. Installer et activer le protocole SSH

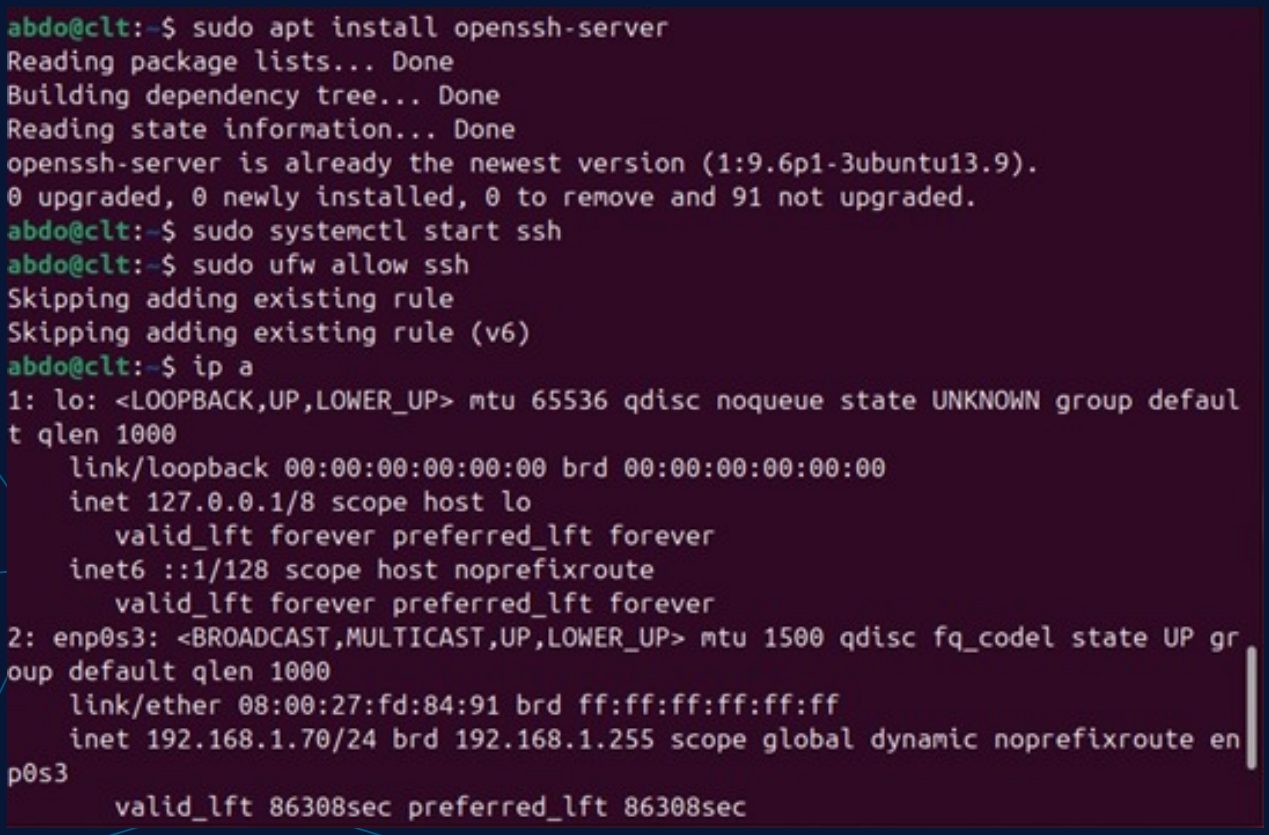
Sudo apt install openssh-server

Sudo apt systemctl start ssh

Sudo ufw allow ssh

Ces commandes servent à installer et activer le serveur SSH pour permettre l'accès à distance à ta machine via SSH.

- **sudo apt install openssh-server :** Installe le serveur OpenSSH, qui permet de se connecter à ton serveur via SSH.
- **sudo systemctl start ssh :** Démarre le service SSH pour permettre les connexions à distance.
- **sudo ufw allow ssh :** Si UFW est activé, cette commande permet d'autoriser les connexions SSH (port 22) à travers le pare-feu. Si tu utilises iptables directement, il n'est pas nécessaire de passer par cette commande.



sudo iptables –policy INPUT ACCEPT

La commande **sudo iptables --policy INPUT ACCEPT** permet de définir la politique par défaut pour la chaîne INPUT d'iptables dans la VM, ce qui signifie que toutes les connexions entrantes seront acceptées sans aucune restriction.
Si on remplace "**Linode**" par une autre VM dans la configuration, cela signifie que la VM remplaçante permettra également toutes les connexions entrantes par défaut, tant que cette politique reste en place. Cela ouvre le serveur à toute connexion sans filtrage préalable.

ssh user_name@address_ip

La commande **ssh user_name@address_ip permet** de se connecter à une machine distante (dans ce cas, une autre VM) via le protocole SSH. Voici une explication détaillée de cette commande :

- **ssh :** C'est le protocole sécurisé pour se connecter à une machine distante.
- **user_name :** Il s'agit du nom d'utilisateur sur la machine distante (par exemple, "root" ou un autre utilisateur).
- **address_ip :** C'est l'adresse IP de la machine distante (la VM avec laquelle tu veux te connecter).

```
taha@taha:~$ ssh abdo@192.168.1.70
The authenticity of host '192.168.1.70 (192.168.1.70)' can't be established.
ED25519 key fingerprint is SHA256:Wz02eZ6g1IHuEAdG8f4OVyg+isKcOwEDDuuPNjBmiUM.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.1.70' (ED25519) to the list of known hosts.
abdo@192.168.1.70's password:
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.11.0-21-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

Expanded Security Maintenance for Applications is not enabled.

80 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
```

4.3. Installer Apache2 (serveur web):

sudo apt install apache2

Cette commande installe le serveur web Apache2 sur ta machine. Apache2 est un des serveurs web les plus utilisés pour héberger des sites et des applications web. Après l'installation, Apache sera démarré automatiquement.

4.4. Vérifier l'adresse IP de la machine ::

ip a

Cette commande affiche toutes les interfaces réseau de sa machine, y compris leurs adresses IP. Tu trouveras l'adresse IP de la machine dans la sortie sous la forme inet X.X.X.X . Cette adresse IP sera utilisée pour tester la connexion au serveur web.

4.5. Accéder au serveur via un navigateur :

Une fois que tu as l'adresse IP de ton serveur, ouvre Mozilla Firefox (ou un autre navigateur) et entre l'adresse IP dans la barre d'adresse, comme suit

http://adresse_ip

```
abdo@clt:~$ sudo apt install apache2
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  apache2-bin apache2-data apache2-utils libapr1t64 libaprutil1-dbd-sqlite3
  libaprutil1-ldap libaprutil1t64
Suggested packages:
  apache2-doc apache2-suexec-pristine | apache2-suexec-custom
The following NEW packages will be installed:
  apache2 apache2-bin apache2-data apache2-utils libapr1t64
  libaprutil1-dbd-sqlite3 libaprutil1-ldap libaprutil1t64
0 upgraded, 8 newly installed, 0 to remove and 91 not upgraded.
Need to get 1,900 kB of archives.
After this operation, 7,455 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://ma.archive.ubuntu.com/ubuntu noble-updates/main amd64 libapr1t64 am
d64 1.7.2-3.1ubuntu0.1 [108 kB]
Get:2 http://ma.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1t64 amd64
1.6.3-1.1ubuntu7 [91.9 kB]
Get:3 http://ma.archive.ubuntu.com/ubuntu noble/main amd64 libaprutil1-dbd-sqlit
e3 amd64 1.6.3-1.1ubuntu7 [11.2 kB]
```

5. Bloquer ou autoriser une adresse IP :

5.1.Rejeter une adresse IP (bloquer l'accès) :

```
iptables -I INPUT -s adresse_ip -j DROP
```

- **-I INPUT :** Insère une règle en tête de la chaîne INPUT (trafic entrant).
- **-s adresse_ip :** Spécifie l'adresse IP source à bloquer.
- **-j DROP :** Indique de rejeter silencieusement les paquets venant de cette IP (sans réponse).

5.2. Accepter une adresse IP (autoriser l'accès) :

```
iptables -I INPUT -s adresse_ip -j ACCEPT
```

Même logique que la commande précédente, mais ici tu autorises l'adresse IP à accéder à ton serveur.

```
abdo@clt:~$ sudo iptables -I INPUT -s 10.0.0.2 -j DROP
abdo@clt:~$ sudo iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       all  --  10.0.0.2             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
abdo@clt:~$ sudo iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    DROP       all  --  10.0.0.2             anywhere

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
```

6. Supprimer une règle iptables ajoutée précédemment :

6.1.Commande de suppression :

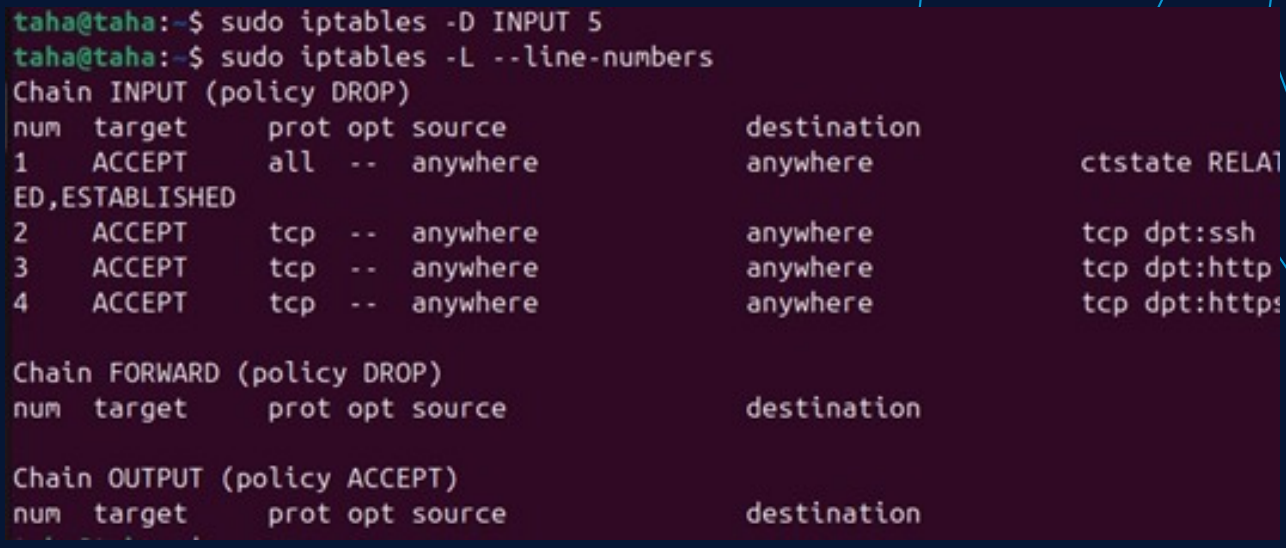
iptables -D INPUT numéro_de_la_règle

- **-D :** Supprime une règle.
- **INPUT :** Spécifie la chaîne (ici, les règles du trafic entrant).
- **numéro_de_la_règle :** C'est le numéro de la règle à supprimer.

6.2.Trouver le numéro de la règle :

iptables -L --line-numbers

- Cette commande affiche toutes les règles actives avec leurs numéros, ce qui te permet de savoir laquelle supprimer.

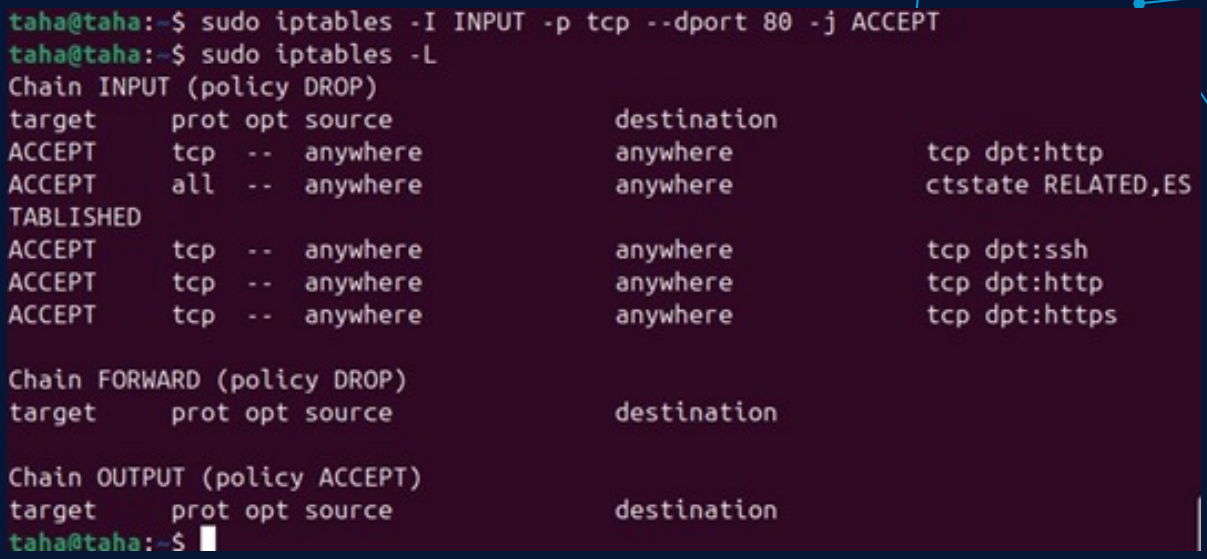


7. Gestion des connexions aux ports:

7.1.Bloquer ou autoriser tout le trafic vers le serveur (web/SSH):

iptables -I nom_de_chaine -p tcp --dport port -j DROP/ACCEPT

- **-I nom_de_chaine :** Insère une règle dans la chaîne spécifiée (**INPUT** pour les connexions entrantes, **OUTPUT** pour sortantes, **FORWARD** pour le routage).
- **-p tcp :** Précise que la règle s'applique au protocole TCP (utilisé par le web et SSH).
- **--dport port :** Spécifie le port de destination (ex. : 22 pour SSH, 80 pour HTTP/web).
- **-j DROP/ACCEPT : DROP** pour bloquer, **ACCEPT** pour autoriser.



7.2. Bloquer / Autoriser une adresse IP sur un port spécifique (comme HTTP ou SSH)

iptables -I nom_de_chaine -p tcp --dport port -s adresse_ip -j DROP/ACCEPT

- **Détail de chaque partie :**

1. **-I nom_de_chaine :** Insère la règle dans la chaîne choisie :
- **INPUT :** pour bloquer/autoriser les connexions entrantes.
- **OUTPUT** : pour bloquer/autoriser les connexions sortantes.
- **FORWARD :** pour le trafic redirigé/routé.
2. **-p tcp :** Spécifie le protocole TCP (utilisé par les connexions web ou SSH).
3. **--dport :** Le port de destination :
- **80** → pour le web (HTTP).
- **22** → pour le SSH.
- En général, on utilise 80 pour un serveur web.
4. **-s adresse_ip :** Spécifie l'adresse IP source (celle qu'on veut bloquer ou autoriser).
5. **-j DROP/ACCEPT :**
- **DROP** → pour bloquer.
- **ACCEPT** → pour autoriser.

```
taha@taha:~$ sudo iptables -I INPUT -p tcp --dport 80 -s 192.168.1.2 -j ACCEPT
taha@taha:~$ sudo iptables -L
```

8.Sauvegarder les règles iptables :

sudo /sbin/iptables-save

Cette commande permet de sauvegarder toutes les règles iptables actuelles dans un format lisible, généralement utilisé pour les restaurer au redémarrage du système.

Conclusion:

Ce projet nous a offert une immersion concrète dans la gestion avancée du trafic réseau à travers la table MANGLE d'IPTables. Nous avons appris à identifier, marquer et contrôler différents types de paquets selon leur origine ou leur destination, en mettant en place un filtrage fin du trafic. L'installation d'IPTables, la création des règles, le contrôle d'accès aux ports et la persistance des configurations nous ont permis de maîtriser les bases essentielles du pare-feu sous Linux. Ce travail nous a également sensibilisés à l'importance de la sécurité réseau et à la puissance de Linux dans l'administration système.