Pare-feu

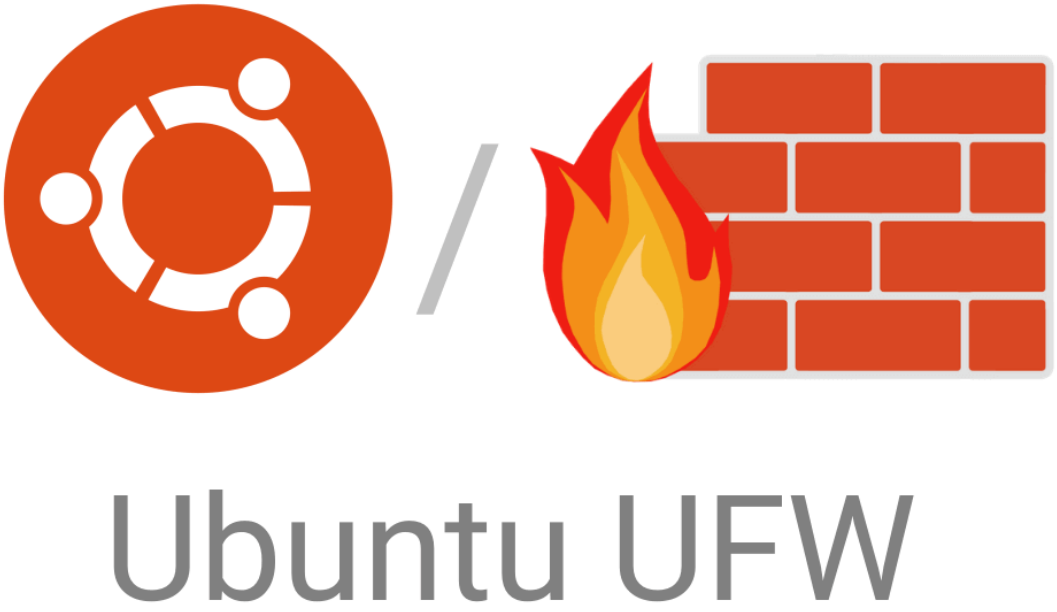
Il ouvre et ferme les porcs, c'est pas très halal.

- [Pare-feu] UFW
- [Pare-feu] IPTables

[Pare-feu] UFW

Introduction

Le pare-feu **UFW** pour **Uncomplicated Firewall** est une surcouche à iptables qui permet de mettre en place des règles de pare-feu sur un système Linux en ligne de commande. Une version GUI de cette application a été implémentée (GUFW).



Source

- Documentation officielle Ubutnu sur UFW

Installation

```
sudo apt install -y ufw
```

Manuel

Afficher les règles en place

```
ufw status [verbose]
```

Définir une politique par défaut

- **Autoriser le trafic entrant** suivant les règles par défaut :

```
ufw default allow
```

- **Refuser le trafic entrant** suivant les règles par défaut :

```
ufw default deny
```

- **Autoriser le trafic sortant** suivant les règles par défaut :

```
ufw default allow outgoing
```

- **Refuser le trafic sortant** suivant les règles par défaut :

```
ufw default deny outgoing
```

Ajout de règle de trafic entrant

- Syntaxe de règle des flux entrants :

```
ufw [allow|deny] [PORT]/[TCP|UDP]
```

Exemple :

```
ufw allow 22/tcp
```

Ajout de règle de trafic sortant

- Syntaxe de règle des flux sortants :

```
ufw [allow|deny] outgoing [PORT]/[TCP|UDP]
```

Exemple :

```
ufw allow outgoing 80/tcp
```

Activer le pare-feu

```
ufw enable
```

Désactiver le pare-feu

```
ufw disable
```

Politique ICMP

Par défaut, UFW ne prend pas en charge la gestion de flux ICMP. Cependant, il est possible d'ajouter des règles iptables dans le fichier de configuration **/etc/ufw/before.rule** pour gérer ces flux.

Autoriser le ping entrant (actif par défaut)

```
# ok icmp codes for INPUT
-A ufw-before-input -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-input -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-input -p icmp --icmp-type parameter-problem -j ACCEPT
-A ufw-before-input -p icmp --icmp-type echo-request -j ACCEPT
```

Autoriser le ping sortant (inactif par défaut)

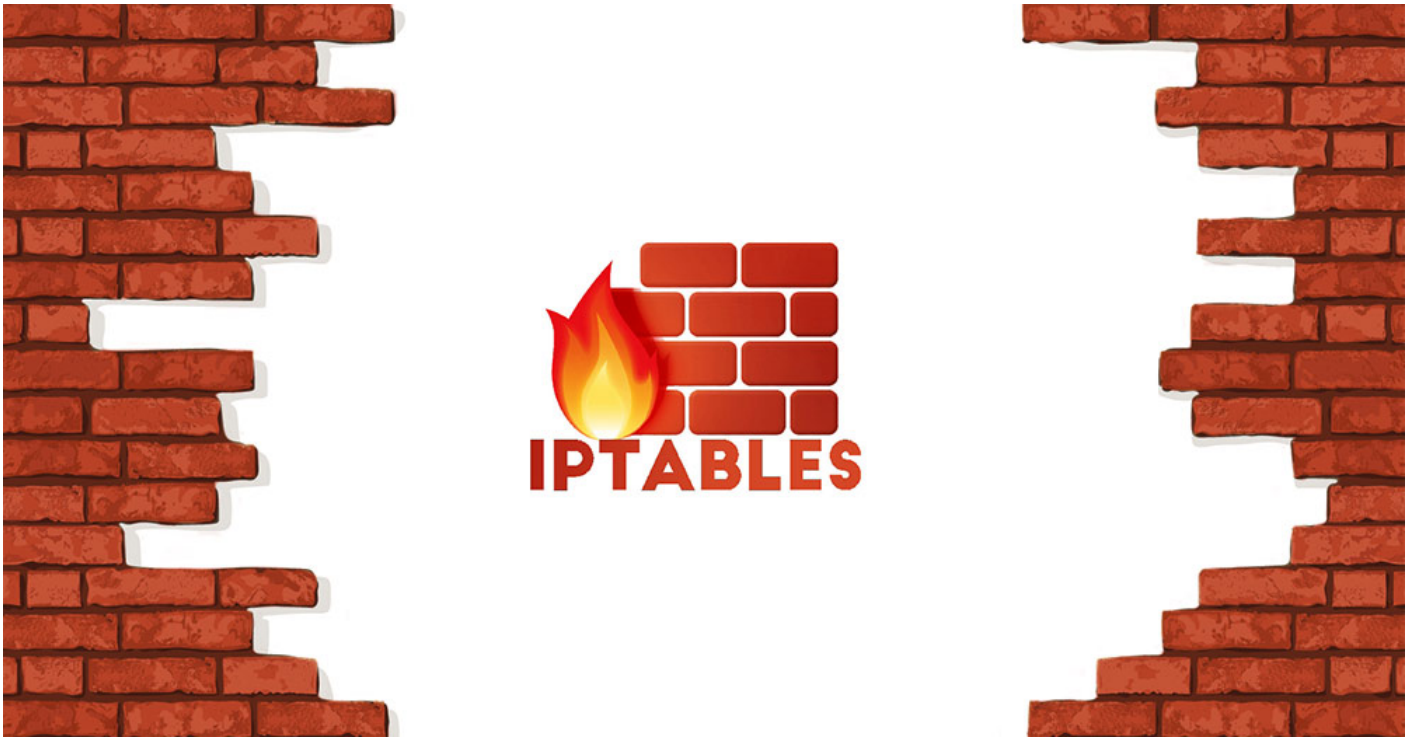
```
# ok icmp codes for OUTPUT
-A ufw-before-output -p icmp --icmp-type destination-unreachable -j ACCEPT
-A ufw-before-output -p icmp --icmp-type time-exceeded -j ACCEPT
-A ufw-before-output -p icmp --icmp-type parameter-problem -j ACCEPT
-A ufw-before-output -p icmp --icmp-type echo-request -j ACCEPT
```

[Pare-feu] IPTables

Introduction

Certainement l'outil le plus complet et le plus fiable sur Linux pour créer vos règles de pare-feu et de routage, **iptables** remplira ses missions sans broncher.

Toutefois, sa multitude d'options fait qu'il est assez rugueux à prendre en main.



Sources

- Documentation Ubuntu - IPTables

Installation

Debian

```
apt install -y iptables
```

Configuration pare-feu

Afficher les règles actives

```
iptables -L
```

> N'affiche que la table "filter". Ajoutez l'option -t suivie de "nat", "mangle" ou "raw" pour voir les tables correspondantes.

Politiques par défaut

- Bloquer le trafic entrant :

```
iptables -P INPUT DROP
```

- Bloquer le trafic sortant :

```
iptables -P OUTPUT DROP
```

- Bloquer le forwarding :

```
iptables -P FORWARD DROP
```

- Autoriser le trafic entrant :

```
iptables -P INPUT ACCEPT
```

- Autoriser le trafic sortant :

```
iptables -P OUTPUT ACCEPT
```

- Autoriser le forwarding :

```
iptables -P FORWARD ACCEPT
```

Autoriser un flux entrant

Tout d'abord, il faut autoriser le trafic déjà établit en sortant :

```
iptables -A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Permettre les connexions entrantes sur un port spécifique, lancez la commande suivante :

```
iptables -A INPUT -p [tcp|udp] -i <IFACE> --dport <PORT> -j ACCEPT
```

Autoriser un flux sortant

Tout d'abord, il faut autoriser le trafic déjà établit en entrant :

```
iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Et bloquer les paquets invalides (en-têtes malveillantes etc) :

```
iptables -A INPUT --m conntrack --ctstate INVALID -j DROP
```

Permettre les connexions sortantes sur un port spécifique, lancez la commande suivante :

```
iptables -A OUTPUT -p [tcp|udp] --dport <PORT> -j ACCEPT
```

Politique ICMP

- Autoriser le ping sortant :

```
iptables -A OUTPUT -p icmp -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j ACCEPT
```

- Autoriser le ping entrant :

```
iptables -A INPUT -p icmp -j ACCEPT
```

- Bloquer le ping sortant :

```
iptables -A OUTPUT -p icmp -m conntrack --ctstate NEW,ESTABLISHED,RELATED -j DROP
```

- Bloquer le ping entrant :

```
iptables -A INPUT -p icmp -j DROP
```

Supprimer une règle

Tout d'abord affichez la table avec les numéros de ligne :

```
iptables -L --line-numbers
```

Admettons la table ci-dessous :

```
Chain INPUT (policy DROP)
num  target     prot opt source               destination
1    DROP       icmp --  anywhere             anywhere
2    ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ssh
3    ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:www
4    ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:webmin

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

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     tcp  --  anywhere             anywhere            tcp spt:www
2    ACCEPT     tcp  --  anywhere             anywhere            tcp spt:12345
```

Dans le cas où on souhaite supprimer la deuxième ligne de la chaîne OUTPUT, on devrait taper cette commande :

```
iptables -D OUTPUT 2
```

Persistence des règles après redémarrage

Exportez vos règles IPv4 dans le fichier **/etc/iptables/rules.v4** et vos règles IPv6 dans le fichier **/etc/iptables/rules.v6** si vous en avez :

```
iptables-save > /etc/iptables/rules.v4
```

```
ip6tables-save > /etc/iptables/rules.v6
```

Installez le paquet **iptables-persistent** :

```
apt install -y iptables-persistent
```

Exemple configuration

Voici un exemple de fichier de configuration qui autorise les connexions sortants sur les ports **80/TCP**, **443/TCP**, **53/UDP** et **80/TCP**, **22/TCP** en entrant :

```
# Generated by iptables-save v1.8.9 (nf_tables) on Sat Jan 20 13:17:34 2024
*filter
:INPUT DROP [30:2646]
:FORWARD DROP [0:0]
:OUTPUT DROP [371:23832]
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -i lo -j ACCEPT
-A INPUT -p icmp -j ACCEPT
-A INPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
-A OUTPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A OUTPUT -o lo -j ACCEPT
-A OUTPUT -p icmp -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 80 -j ACCEPT
-A OUTPUT -p tcp -m tcp --dport 443 -j ACCEPT
-A OUTPUT -p udp -m udp --dport 53 -j ACCEPT
COMMIT
# Completed on Sat Jan 20 13:17:34 2024
```

Configuration NAT

Tout d'abord, activez la fonctionnalité de **port forwarding** dans le fichier **/etc/sysctl.conf** en décommentant la ligne suivante :

```
net.ipv4.ip_forward=1
```

Puis créez les règles de routage suivantes :

```
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

iptables -A INPUT -i eth1 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -j ACCEPT
```

> Remplacez **eth0** par le nom de l'interface de votre premier réseau et faites de même pour **eth1** avec le deuxième réseau.