

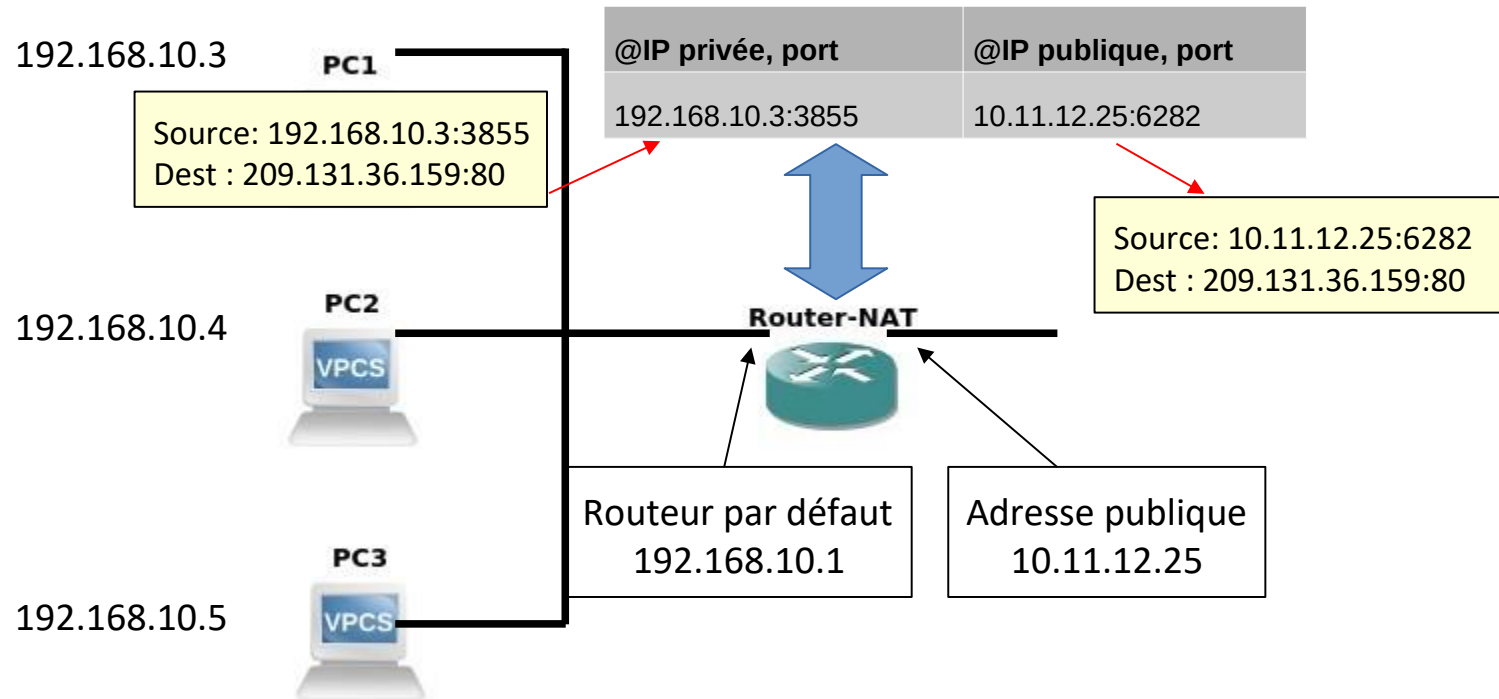
Plan

- Translation adresses et ports (NAT/PAT)
- Fonctionnement pare-feu
- Principe iptables
- Commandes iptables
- Exemple autre pare-feu (ufw)

Translation d'adresses et ports (NAT/PAT)

NAT : Objectifs

- IP Network Address Translator (NAT) : RFC 1631, 3022, 7857
- NAT permet plusieurs machines hôtes de partager une même adresse IP
 - Machines réseau local utilisant adresse(s) publique(s)
- Serveur NAT placé en périphérie de réseau, en général associé à un routeur



NAT : différents modes

- Plusieurs façon de faire du NAT : statique, dynamique
- NAT statique
 - Traduction ensemble adresses internes (privées) vers autre ensemble adresses externes de même taille (type 1 pour 1)

- Modification champ @IP dans en-tête IP
- Table de correspondance prédéfinie des adresses IP internes/externes
- NAT dynamique (*masquerading*)
 - Attribution adresse IP externe **lors de la requête**
 - Utile pour réduire nombre adresses externes (@externe non utilisée par chaque client) → **avoir assez d'adresses externes pour connexions simultanées**
- NAPT (Network Address and Port Translation)
 - Problème NAT dynamique résolue avec association @IP et n°port
 - Translation de port (*port forwarding*)
 - Besoin identifier chaque flux (contexte utilisateur) :
- IP src in, TCP/UDP, port src in ↔ IP sortie out, TCP/UDP, port src out

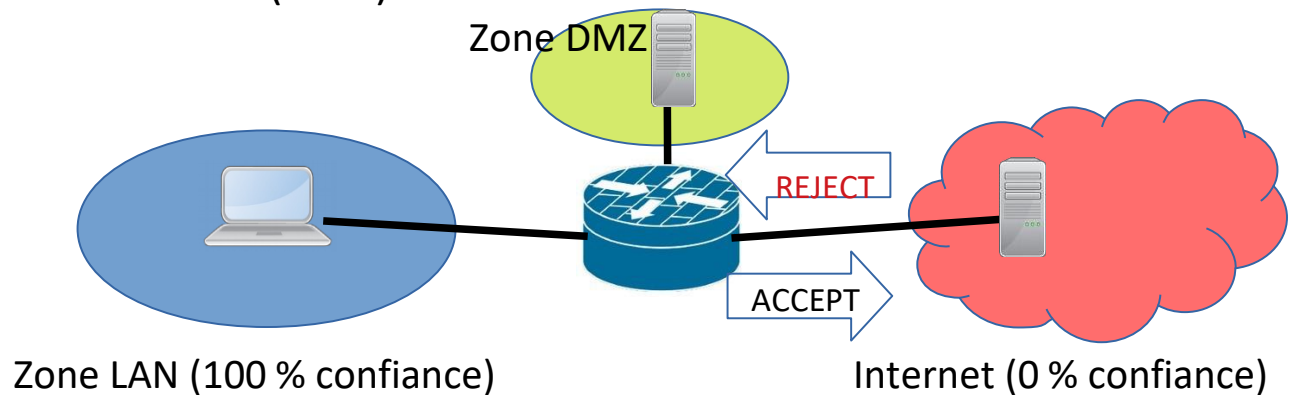
NAT : différents modes

- Bi-directionnal NAT
 - NAT dynamique ne rend pas accessible machine interne depuis extérieur
 - Connexion initiée à partir réseau privé (pas de l'extérieur)
 - Permet accès direct machines du réseau privé depuis extérieur

- Utilisation du DNS pour interpréter requêtes
- **Combiner NAT statique et dynamique**
 - NAT statique permet accès machine interne de l'extérieur
 - NAT dynamique permet réduire nb adresse publique pour sortie vers Internet
- **Twice-NAT**
 - Transformation adresses/ports source et destination
 - NAT cache @ internes vis-à-vis réseau externe – NAT cache @ externes vis-à-vis réseau interne
 - Eviter conflits adresses avec plusieurs réseaux privés internes

Fonctionnement d'un pare-feu

- Permet le filtrage et le contrôle du trafic TCP/IP
 - Accès non-autorisés (interface)
 - Filtrage fin des flux à chaque niveau couches TCP/IP
 - Empêche attaques de déni de service
- Ne protège pas des menaces internes (analyse trafic traversant pare-feu)
- Contrôle trafic entre zones de confiance
 - Internet (confiance nulle)
 - 1 ou plusieurs réseaux internes (confiance plus importante)
 - Zone démilitarisées (DMZ) avec niveaux confiance différents



Pare-feu

- Configuration pare-feu consiste ensemble de règles déterminant action de rejet ou d'autorisation du trafic qui passe les interfaces du pare-feu suivant
 - Source/destination trafic
 - Protocole de couche 3 (IPv4 ou v6, ...)
 - Protocole de couche 4 (TCP, UDP, ...)
 - Protocole applicatif (HTTP, SMTP, DNS, ...)
- Application règle suivant
 - sens trafic (entrant/sortant) d'une interface
 - avant/après processus routage paquets

Netfilter sous Linux

- Netfilter est le module du noyau Linux implémentant un pare-feu (filtrage + manipulation paquets)
 - Filtrage niveau 2 : interface, @MAC
 - Filtrage niveau 3 : @source, @dest, ToS (Type of Service), TTL, protocole
 - Filtrage niveau 4 : ports, flags TCP
 - Filtrage suivant taux d'arrivée
- Netfilter permet
 - Rejet paquet (en informant émetteur)
 - Destruction paquet (sans informer émetteur)
 - Réécriture paquet (@IP, ports, TTL ...)
 - Notification dans journal
 - Acceptation et traitement dans espace utilisateur
- Comportement de netfilter est défini par des règles appartenant à l'une des 3 trois tables (mangle, nat, filter)

Netfilter : Tables

- Chaque table regroupe ensemble de règles avec même utilisation
 - Mangle : règles modifiant des paquets (modification TTL, application QoS, ...)
 - Nat : règles permettant les translations d'adresses
 - Filter : règles pour filtrage des paquets (acceptation, destruction, rejet ...)
- Table filter utilisée par défaut si non précision dans une règle
- Chaque règle est composée
 - d'une chaîne
 - de conditions
 - de cibles

Netfilter : Chaînes

- Chaînes définissent des points traversés par les paquets lors du processus de filtrage
 - INPUT : paquets entrant à destination de la machine locale
 - OUTPUT : paquets sortant émis par machine locale
 - FORWARD : paquets traversant la machines par une interface et sortant par une autre
 - PREROUTING : paquets reçu du réseau
- DNAT : modification @destination
- POSTROUTING : paquets émis sur le réseau après décision de routage
 - SNAT : modification @source
- Autre chaîne utilisateur : possibilité de créer sa propre chaîne dans une table

Netfilter : table filter

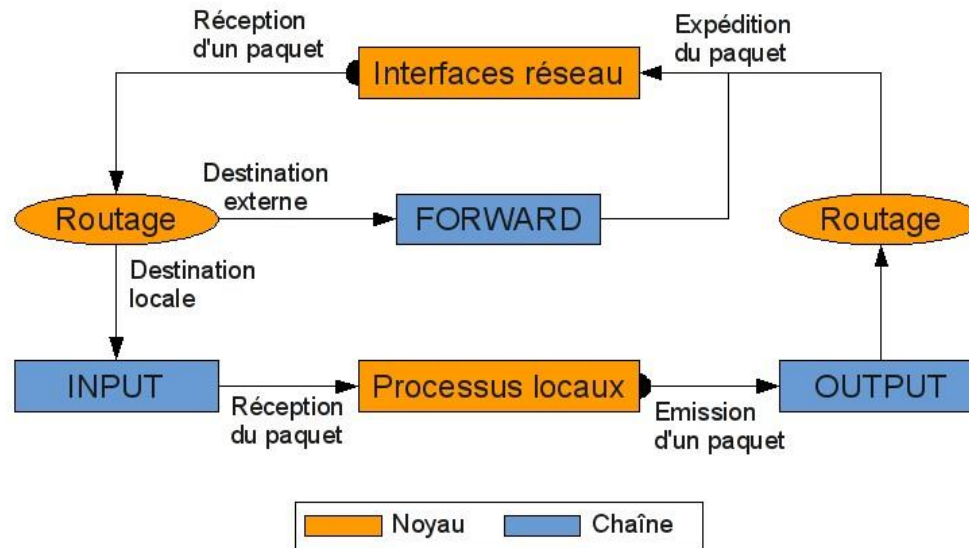


Schéma de fonctionnement de la table filter de Netfilter
Source : Michael Witrant

Netfilter : table nat

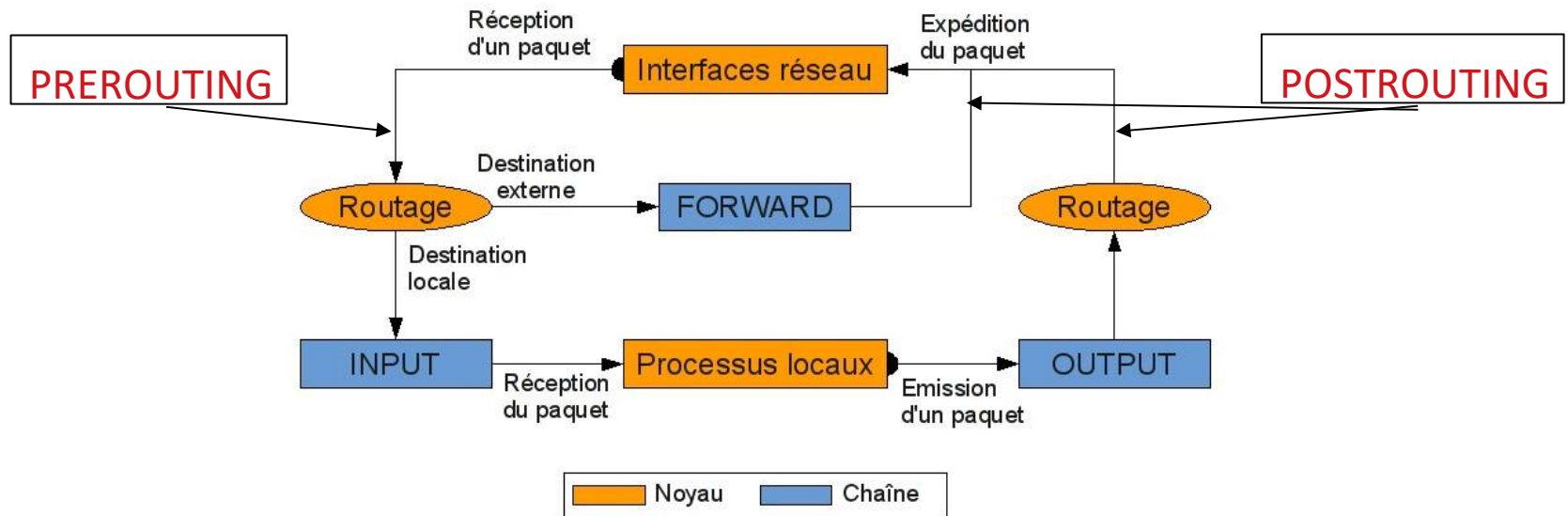


Schéma de fonctionnement de la table nat de Netfilter
Source : Michael Witrant

Netfilter : cibles

- Une cible spécifie l'action à appliquer au paquet
- 2 types de cibles : terminales ou non-terminales
- Cibles terminales
 - ACCEPT : acceptation du paquet
 - DROP : destruction du paquet (silencieusement, source non avertie)
 - REJECT : rejet du paquet et envoie message à l'expéditeur (avec code ICMP ou TCP)
 - SNAT : translation adresse source
 - MIRROR : renvoie à l'expéditeur
- Cibles non-terminales
 - MARK : marquage paquet pour action ultérieure
 - LOG : ajout entrée dans le journal
 - QUEUE : traitement paquet dans espace utilisateur
 - ...

Netfilter : politiques

- Politique par défaut pour chaque chaîne
 - Utilisation politique par défaut si aucune règle applicable
- Politiques possibles
 - ACCEPT : acceptation du paquet
 - DROP : rejet silencieux du paquet
- Conseillé d'utiliser DROP pour politique par défaut
- Fermer tous les ports
- Ouvrir uniquement ports nécessaires

Netfilter : iptables

- iptables est l'outil pour gérer les règles utilisées par netfilter
- Syntaxe iptables

Commandes :

iptables -h

iptables -L

iptables -N *<chaine>*

iptables -X *<chaine>*

iptables -F [*chaine*]

iptables -P *<chaine>* *<cible>*

iptables -A *<chaine>* *règle*

iptables -I *<chaine>* [*numéro*] *règle*

iptables -D *<chaine>* [*numéro*] [*règle*]

iptables -R *<chaine>* [*numéro*] [*règle*]

iptables -C *<chaine>*

iptables -Z [*chaine*]

Fonctions :

Aide en ligne

Lister les chaînes et règles actives

Créer une nouvelle chaîne utilisateur

Supprimer chaîne utilisateur

Vider une chaîne (ou toutes)

Traitement par défaut pour une règle

Ajouter une *règle* à la chaîne

Insérer une chaîne en position *numéro*

Effacer une *règle*

Remplacer une *règle*

Tester présence d'une règle

Remettre à zéro les compteurs

Netfilter : iptables

Principaux paramètres de la partie règle

— le ! signifie la négation (cf. exemple)	<u>Paramètres :</u>
[!] -i <i>interface</i>	Appliquer la règle sur cette interface d'entrée
[!] -o <i>interface</i>	Appliquer la règle sur cette interface de sortie
-t <i>table</i>	Table concernée (filter par défaut)
-n	Affichage valeurs numériques
[!] -s <i>X.X.X.X[/masque]</i>	Adresse source, longueur du masque,
[!] -d <i>X.X.X.X[/masque]</i>	Adresse dest (et longueur du masque)
[!] -p <i>protocole</i>	tcp / udp / icmp / all
[!] --sport [<i>port[:port]</i>]	Port source (ou intervalle de ports)
[!] --dport [<i>port[:port]</i>]	Port destination (ou intervalle de ports)
-j <i>cible</i>	ACCEPT/DROP/QUEUE/RETURN/REDIRECT/ MASQUERADE / DNAT/ SNAT/ LOG

Netfilter : iptables

Exemples de règles iptables

→ Politique par défaut (rejet total et silencieux) : `iptables -F INPUT && iptables -P INPUT DROP`

→ Acceptation connexions TCP sur port 22 (SSH) venant du réseau 10.1.1.0/24 :

```
iptables -A INPUT -p tcp -m tcp -s 10.1.1.0/24 -- dport 22 -j ACCEPT
```

→ Suppression règle n°1 chaîne INPUT : `iptables -D INPUT 1`

→ Acceptation connexions UDP sur port 53 (DNS) ne venant pas de 10.1.1.23 :

```
iptables -A INPUT -p udp -m udp -s ! 10.1.1.23 -- dport 53 -j ACCEPT
```

→ Acceptation connexions TCP vers port 3306 (MySQL) uniquement via interface loopback (local)

```
iptables -A INPUT -p tcp -m tcp -i lo -- dport 3306 -j ACCEPT
```

→ le module multiport permet de 'matcher' plusieurs ports en une règle

```
iptables -A INPUT -m multiport -p tcp -s 10.1.1.1 -- dports  
smtp,imap,pop3 -j ACCEPT
```

→ Enregistrement dans journal système (syslog) connexion extérieure sur serveur MySQL

```
iptables -A INPUT -p tcp -m tcp ! -i lo --dport 3306 -j LOG --log-  
prefix "MSG : "
```

Netfilter : iptables

- Paramètres spécifiques SNAT, DNAT et redirection (locale)
 - Translation port (PAT) est incluse dans SNAT et DNAT

Paramètres :

<code>--to-source @ip[-@ip][:port-port]</code>	Changement adresse source statique (SNAT)
<code>--to-destination @ip[-@ip][:port-port]</code>	Renvoie vers autre hôte (DNAT)
<code>--to-ports port[-port]</code>	redirection port local (DNAT, proxy transparent)

Netfilter : iptables

- Exemples mise en place DNAT
 - eth0 : interface vers réseau local
 - eth1 : interface vers Internet

→ DNAT (redirection ssh entrant réseau externe vers adresse réseau interne) :

```
echo 0 > /proc/sys/net/ipv4/ip_forward
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
iptables -A FORWARD -i eth0 -o eth1 -s 192.168.1.0/24 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -p tcp -dport 22 -j ACCEPT
iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 22 -j DNAT --to-destination
192.168.1.99 echo 1 >
/proc/sys/net/ipv4/ip_forward
```

Netfilter : iptables

- Exemples mise en place SNAT
 - eth0 : interface vers réseau local
 - eth1 : interface vers Internet

→ SNAT (accès de toutes machines sur réseau local via eth0 vers site web (port 80) sur Internet via eth1) :

```
echo 0 > /proc/sys/net/ipv4/ip_forward
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
iptables -A FORWARD -i eth0 -o eth1 -s 192.168.1.0/24 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -d 192.168.1.0/24 -p tcp --sport 80 -j ACCEPT
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.1.0/24 -j SNAT --to-source
```

```
11.12.13.14 echo 1 >
/proc/sys/net/ipv4/ip_forward
```

– Cas avec adresses publiques multiples :

```
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.1.0/24 -j SNAT --to-source  
11.12.13.14-11.12.13.20
```

Netfilter : iptables

- Exemples mise en place SNAT
 - eth0 : interface vers réseau local
 - eth1 : interface vers Internet

→ SNAT (accès de toutes machines sur réseau local via eth0 vers site web (port 80) sur Internet via eth1) :

```
echo 0 > /proc/sys/net/ipv4/ip_forward
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
iptables -A FORWARD -i eth0 -o eth1 -s 192.168.1.0/24 -p tcp --dport 80 -j ACCEPT
iptables -A FORWARD -i eth1 -o eth0 -d 192.168.1.0/24 -p tcp --sport 80 -j ACCEPT
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.1.0/24 -j SNAT --to-source
11.12.13.14 echo 1 >
/proc/sys/net/ipv4/ip_forward
```

- Cas adresse dynamique interface vers Internet :

```
iptables -t nat -A POSTROUTING -o eth1 -s 192.168.1.0/24 -j MASQUERADE
```

Netfilter : iptables

- **Changement de port**

- **Redirection port web (80) vers autre port local (8080)**

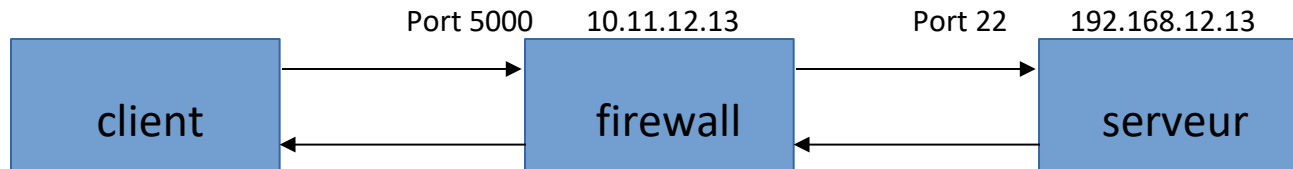
```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j  
REDIRECT --to-ports 8080
```

- **Redirection flux ssh port 5000 vers port local 22**

```
iptables -t nat -A PREROUTING -p tcp --dport 5000 -j  
REDIRECT --to-ports 22
```

- **Redirection flux ssh port 5000 vers autre machine port 22**

```
iptables -t nat -A PREROUTING -p tcp --dport 5000 -j DNAT  
--to-destination 192.168.12.13:22 iptables -t nat -A  
POSTROUTING -p tcp --dport 22 -j  
MASQUERADE
```



Netfilter : iptables

- Règles non persistantes, besoin de les charger au prochaine redémarrage
 - Fichier avec règles chargée au démarrage (debian) :
`/etc/iptables.up.rules`
- Sauvegarde des règles de iptables courantes (debian)
 - `iptables-save > /etc/iptables.up.rules`
- Restauration règles sauvegardées au démarrage
 - Création fichier script chargement règles
 - `editor /etc/network/if-pre-up.d/iptables`
 - Ajout lignes suivantes :

```
#!/bin/sh
/sbin/iptables-restore < /etc/iptables.up.rules
```
 - `chmod +x /etc/network/if-pre-up.d/iptables`

Netfilter : nftables

- Nftables est un projet de Netfilter qui remplace iptables
 - Code iptables lourd à maintenir
 - Ne passe pas à l'échelle (rechargement ensemble règles à chaque modification) → plusieurs minutes pour milliers chaînes
 - Impossibilité de gérer plusieurs adresses dans une règle
- Nouveau firewall par défaut des noyaux Linux 2.6 • Compatibilité avec iptables
 - Transformation en ligne des règles pour nftables
 - iptables-legacy : est iptables, iptables-nft : est nftables
- Possibilité de transformer règles iptables et obtenir règle nftables

```
iptables-translate -A INPUT -p tcp --dport 22 -j ACCEPT
nft add rule ip filter INPUT tcp dport 22 counter accept
```

Uncomplicated firewall (ufw)

- Pare-feu surcouche de iptables permettant de configurer des règles simples

- Etat et mise en route de ufw (non activé par défaut)

```
ufw enable    # Activation pare-feu ufw
ufw disable   # Desactivation pare-feu ufw
ufw status    # Affiche etat pare-feu
```

- Information sur les règles en application

- Paquets entrant et sortant acceptés

```
ufw status verbose          # Affiche etat pare-feu et regles
Status: active             # firewall actif
Logging: on (low)          # journal actif /var/log/ufw.log
Default: allow (incoming), allow (outgoing)
New profiles: skip
```

Uncomplicated firewall (ufw)

- Gérer la politique par défaut

```
ufw default allow
```

```
# Autorise trafic entrant et  
# sortant sur toutes interfaces
```

```
ufw default deny
```

```
# Interdit tout trafic sur  
# toutes interfaces
```

```
ufw default allow incoming
```

```
# Autorise trafic entrant
```

```
ufw default allow outgoing
```

```
# Autorise trafic sortant
```

```
ufw default deny incoming
```

```
# Interdit trafic entrant
```

```
ufw default deny outgoing
```

```
# Interdit trafic sortant
```

Contrôler un port ufw allow

```
[port|service] ufw deny
```

```
[port|service] ufw deny
```

```
out service
```

```
# Autoriser trafic entrant  
# et sortant sur port
```

```
# Interdit tout trafic
```

```
# entrée port
```

```
# Interdit trafic sortant
```

```
# service
```

Supprimer une règle

- ufw status numbered # Affiche numero des règles ufw
- delete <num_regle> # Suppression regle

Uncomplicated firewall (ufw)

- Liste port(s) réservé(s) pour services : /etc/services • Exemple service

```
# Autoriser port ssh dns (53) en UPD et TCP
ufw allow 53
# Autoriser en sortie en TCP à
Internet ufw allow out 80/tcp ufw
allow out 443/tcp
```

- Création de règles plus complexes

```
# Interdire protocole TCP à tout le monde sur port 80
ufw deny proto tcp to any port 80
# Interdire données protocole TCP provenant de 1.2.3.4
# sur port 80
ufw deny proto tcp from 1.2.3.4 to any port 80
# Interdire données sortie port 80 destinée à
192.168.0.5 ufw deny out from 192.168.0.5 to any port 80
```