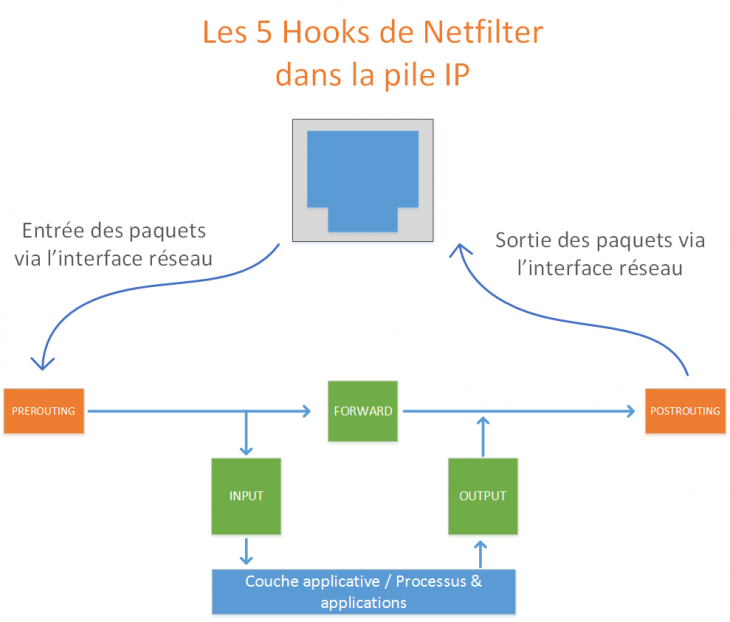
|  |  |
| --- | --- |
| No alt text provided for this image | **Étude de cas** |

## Points d’ancrage nftables

Un point d’ancrage est un point précis dans les flux réseau qui arrivent et sortent du système.   
Les points d’ancrages sont les éléments sur lesquels s'appliquent les règles et qui permettent donc de modifier le comportement des paquets en entrée ou sortie de l'interface en fonction des règles configurées.

Lorsque le flux de trafic entre dans un hôte local, il fait d'abord face au point d’ancrage de pré-routage (*prerouting*), puis au point d’ancrage d'entrée (*input*).   
Ensuite, le trafic généré par les processus de l’hôte local suit le point d’ancrage de sortie (*output*), puis le point d’ancrage de post‑routage (*postrouting*).

Les paquets destinés à son réseau mais qui ne sont pas adressés au nœud local feront face au point d’ancrage de transfert (*forward*) après avoir suivi le chemin de pré-routage puis de post-routage.   
Le point d’ancrage d'entrée (*input*), cependant, qui est un nouveau point d’ancrage dans nftables, est un point d’ancrage qui est placé avant tous les points d’ancrage (tout de suite après le point d’ancrage de pré-routage). Il peut ainsi filtrer le trafic de la couche 2 du modèle OSI.   
Avec ce point d’ancrage, donc, des politiques de filtrage précoces peuvent être définies.



**Pré-routage** (*prerouting*)  
Tous les paquets entrant dans un nœud sont traités par ce point d’ancrage.   
Il est invoqué avant le processus de routage et est utilisé pour le filtrage précoce ou la modification des attributs de paquet qui affectent le routage ;

**Entrée** (*input*)  
Ce point d’ancrage est exécuté après la décision de routage.   
Les paquets livrés à un système local sont traités par ce point d’ancrage ;

**Transfert** (*forward*)  
Ce point d’ancrage se produit également après la décision de routage.   
Les paquets qui ne sont pas dirigés vers l’hôte local sont traités par ce point d’ancrage ;

**Sortie** (*output*)  
Ce point d’ancrage contrôle les paquets provenant des processus d'une machine locale ;

**Post-routage** (*Postrouting*)  
Ce point d’ancrage est utilisé pour les paquets quittant un système local après la décision de routage ;

# Installation

Pour installer le paquet de nftables, l’activer et le rendre actif au démarrage, on exécute les commandes suivantes …  
**>> sudo apt install nftables**

Pour vérifier le statut du service …  
**>> sudo systemctl status nftables.service  
>> sudo systemctl start nftables.service  
>> sudo systemctl stop nftables.service**

Commandes complémentaires … **>> sudo systemctl start nftables.service  
>> sudo systemctl enable nftables.service**

# Configuration initiale nftables

## Vérification de la configuration actuelle

Afin de configurer nftables, il est d'abord nécessaire de vérifier le contenu actuel du fichier de configuration nftables.

Par défaut, les configurations nftables se retrouvent dans le fichier **/etc/nftables.conf** …  
**>> sudo nft list ruleset**

Chaque règle possède un certain nombre de poignée (*handle*).   
Pour les afficher, on utilise le commutateur **-a** ou **–handle** …  
**>> sudo nft --handle list ruleset**ou **>> sudo nft -a list ruleset**Pour afficher les tables …  
**>> sudo nft list tables**

Pour lister les tables de la famille IP …  
**>> sudo nft list tables ip**

Pour afficher les tables IPv6 …  
**>> sudo nft list tables ip6**

# Modules nftables

## Tables

Une table est au sommet de l'ensemble de règles en tant que conteneur dans lequel les chaînes qui sont les conteneurs des règles sont conservées.

Compte tenu de l'importance des tables, la présentation hiérarchique de la structure de nftables est la suivante …  
**Tables > Chaînes > Règles**

Il est possible de manipuler une table, une chaîne ou une règle à l'aide des outils de commande nft ou encore en les écrivant directement dans le fichier de configuration nftablrs.

### Familles d'adresses

Les familles d'adresses déterminent le type de paquets entrants et sortants traités par nftables.

Les principales familles d'adresses sont les suivantes …

* **ip**  
  Famille d'adresses IPv4.
* **ip6** ;  
  Famille d'adresses IPv6 ;
* **inet**  
  Familles d'adresses IPv4 et IPv6 ;

## Chaînes

Une chaîne est un conteneur de règles et se situe à l'intérieur d'une table existante.

Il existe trois types de chaine …

* **Filter**   
  Il s'agit d'un type de chaîne standard et prend en charge toutes les familles d'adresses ;
* **Route**  
  Cette chaine ne prend en charge que les familles d'adresses ip et ip6 et uniquement le point d’ancrage de sortie ;
* **NAT**  
  Cette chaine peut effectuer la traduction d'adresses réseau et ne prend en charge que les familles d'adresses ip et ip6.

### Stratégies (*Policies*)

Les chaînes doivent avoir leurs politiques selon lesquelles les paquets sont traités pour être rejetés ou acceptés par défaut.   
Ces valeurs de stratégie peuvent être **Accepter**, qui est la stratégie par défaut, ou **Abandon**.   
Par défaut, la stratégie d’une chaine est **acccept**.

On peut définir une stratégie (*policy*) lors de l’ajout d’une chaîne …  
**>> sudo nft add chain <Famille> <Table> <Chaine> { type <Type de chane> hook <point d’ancrage> …  
 … priority <Niveau de priorité> \; policy <Déclaration> \; }**  
**>> sudo nft add chain inet filter INPUT { type filter hook input priority 0 \; policy drop \; }  
>> sudo nft add chain inet filter FORWARD { type filter hook input priority 0 \; policy drop \; }  
>> sudo nft add chain inet filter OUTPUT { type filter hook input priority 0 \; policy drop \; }**

**Attention** …  
Une stratégie avec la déclaration drop peut fermer la connexion SSH actuelle.

## Règles (*Rules*)

Les règles sont les actions qui contrôlent les paquets entrants et sortants en fonction des points d’ancrage définis dans une chaîne. Si une règle à l'intérieur d'une chaîne correspond à un paquet basé sur l'étape dérivée de leurs points d’ancrage, le paquet est abandonné ou accepté.

Une règle est évaluée de gauche à droite de telle sorte que lorsque la première instruction correspond, elle continue avec les parties suivantes d'une règle.  
Sinon, la règle suivante sera évaluée.

La structure d'une règle comprend des correspondances et des instructions qui sont les suivantes …  
**<Correspondance> <Déclarations>**

### Correspondance

Les correspondances sont ces filtres qui permettent à une règle de filtrer certains paquets.

Voici brièvement quelques correspondances importantes avec leurs formats possibles …

* **ip saddr** <**Adresse IP source**>
* **Ip daddr** <**Adresse IP de destination**>
* **tcp/udp sport** <**Port source**>
* **tcp/udp dport** <**Port de destination**>
* **tcp flags** <**Drapeau**>
* **icmp type** <**Type**>
* **iifname** <**Nom de l’interface d’entrée**>
* **oifname** <**Nom** **de l’interface de sortie**>
* **protocol** <**Protocole**>

### Déclarations

Une instruction est l'action définie effectuée une fois qu'un paquet correspond à une ou plusieurs correspondances définies par une règle.   
Les déclarations comprennent des déclarations de verdict, de journal et de compteur.

**Déclarations de verdict**Une déclaration de verdict modifie le flux de contrôle dans l'ensemble de règles et émet des décisions de politique pour les paquets.

Les deux principales déclarations de verdict valides sont …

* **Accept**  
  Acceptation du paquet et arrête l'évaluation des règles restantes ;
* **Drop**  
  Suppression du paquet et arrêt de l'évaluation des règles restantes ;

# Manipulation des objets

**Préalable** …  
On peut **effacer toutes les données actuelles** du fichier de configuration et de créer ses propres modules nftables.  
Pour nettoyer …  
**>> sudo nft flush ruleset**

## Module d’aide

Pour **afficher le module d’aide** de nftables …  
**>> sudo nft --help**

**Remarque** …  
nftables est sensible à la casse … **INPUT** est différent d’**input**.

## Ajout d’un objet

### Ajout d’une table

Pour **créer une table** …  
**>> sudo nft add table <Famille> <Table>**  
**>> sudo nft add table ip filter**

### Ajout d’une chaine

Pour **ajouter une chaine de base** …  
**>> sudo nft add chain <Chaine> <Point d’ancrage> { type type> hook <Point d’ancrage> priority <Priorité> \; }  
>> sudo nft add chain ip filter INPUT { type filter hook input priority 0 \; }  
>> sudo nft add chain ip filter OUTPUT { type filter hook output priority 0 \; }**

La priorité la plus basse prend ainsi le pas sur les autres priorités.

**Remarque** …  
La priorité la plus basse prend ainsi le pas sur les autres priorités.  
Il est possible d'avoir des priorités négatives (Exemple : -100) qui sont alors prioritaires sur une priorité 0.

**Vérification de la création d’une table**Pour vérifier la création de la table filter et des chaines INPUT et OUTPUT …  
**>> sudo nft --handle list ruleset  
ou  
>> sudo nft -a list ruleset**

### Ajout de règles

Pour **abandonner silencieusement les paquets** **vers une adresse externe** …  
**>> sudo nft add rule <Famille> <Table> <Chaine> <Famille> daddr <Adresse IP> <Déclaration>  
>> sudo nft add rule ip filter OUTPUT ip daddr 192.168.1.100 drop  
>> sudo nft add --handle list ruleset**

Les **compteurs de règles**[[1]](#footnote-1) sont facultatifs avec nftables et le mot-clé counter doit être utilisé pour l'activer …  
**>> sudo nft add rule <Famille> <Table> <Chaine> <Famille> daddr <Adresse IP> counter <Déclaration>  
>> sudo nft add rule ip filter OUTPUT ip daddr 192.168.1.200 counter drop  
>> sudo nft --handle list ruleset**Pour **ajouter une règle pour un réseau** …  
**>> sudo nft add rule <Famille> <Table> <chaine> <Famille> daddr <Adresse du reseau/CIDR> counter <Déclaration>  
>> sudo nft add rule ip filter OUTPUT ip daddr 192.168.1.0/24 counter accept  
>> sudo nft --handle list ruleset**

Pour **abandonner silencieusement les paquets vers le port 80** …  
**>> sudo nft add rule <Famille> <Table> <Chaine> <Protocole> dport <Numéro de port> <Déclaration>  
>> sudo nft add rule ip filter INPUT tcp dport 80 drop  
>> sudo nft --handle list ruleset**

Pour **accepter une demande d'écho ICMP** …  
**>> sudo nft add rule <Famille> <Table> Chaine> icmp type echo-request <Déclaration>  
>> sudo nft add rule filter OUTPUT icmp type echo-request accept  
>> sudo nft --handle list ruleset**

Pour **combiner le filtrage**, il suffit de spécifier plusieurs fois la syntaxe ip …  
**>> sudo nft add rule <Famille> <Table> <Chaine> ip protocol icmp ip daddr <Adresse IP> counter <Déclaration>  
>> sudo nft add rule ip filter OUTPUT ip protocol icmp ip daddr 192.168.1.100 counter drop  
>> sudo nft --handle list ruleset**

Pour **insérer une règle à la suite d’une position précise** …  
**>> sudo nft add rule <Table> <chaine> position <Numéro de poignée> <Famille>daddr <Adresse IP> <Déclaration>  
>> sudo nft add rule filter INPUT position 8 ip daddr 192.168.1.111 drop**

Si on **souhaite uniquement faire correspondre un protocole** **(ajout au début de la chaine)** …  
**>> sudo nft insert rule <Table> <chaine> <Famille>protocole <Protocol> counter  
>> sudo nft insert rule filter OUTPUT ip protocol tcp counter**

### Suivi de connexion

Pour **accepter tous les paquets entrants d'une connexion établie** …  
**>> sudo nft insert rule <Table> input ct state established <Déclaration>  
>> sudo nft insert rule filter INPUT ct state established accept**

## Suppression d’objet

### Vidange d’une table

La vidange d'une table supprime toutes les chaînes et règles de celle-ci, mais laisse la table elle-même en place.

Pour **vider une table** …  
**>> sudo nft flush table <Table>  
>> sudo nft flush table filter**

### Suppression d’une table

La suppression d'un tableau supprime la table et tout son contenu.   
Pour **supprimer une table** …  
**>> sudo nft delete table <Table>  
>> sudo nft delete table filter**

**Remarque** …  
Si la **table n'est pas vide**, c’est-à-dire s'il y a des chaines et des tables de configurées à l'intérieur, nftables va **refuser de** **la supprimer** et **retourner un message d’erreur**.

### Vidange d’une chaine

Le vidage d'une chaîne en supprime toutes les règles mais ne supprime pas la chaîne elle-même, y compris ses propriétés.

Pour **vider une chaine** …  
La commande est sous la forme :nft flush chain <table name> <chain name>  
**>> sudo nft flush chain <Table> <Chaine>   
>> sudo nft flush chain filter OUTPUT**

### Suppression d’une chaine

Pour supprimer une chaine …  
**>> sudo nft delete chain <Table> <Chaine>   
>> sudo nft delete chain filter OUTPUT**

**Remarque** …  
On ne peut **supprimer qu'une chaîne vide**.  
Il faut donc vider la chaîne avant d'entrer la commande de suppression.

### Suppression de règles

**Remarque** …  
Afin de supprimer une règle spécifique, on doit identifier au préalable son numéro de de poignée (*handle*).

Pour supprimer une règle …  
**>> sudo nft delete rule <Table> <Chaine> handle <Numéro de poignée>  
>> sudo nft delete rule filter OUTPUT handle 7**

Il est également possible de supprimer une règle sans désigner sa poignée …  
**>> sudo nft delete rule <Table> <Chaine> <Famille><Adresse IP> counter  
>> sudo nft delete rule filter OUTPUT ip saddr 192.168.1.100 counter**

## IPv6

La manipulation des objets **IPv6 suit en toute les règles d’iPv4**.

Pour **ajouter une règle** …  
**>> sudo nft add rule <Famille> <Table> <Chaine> <Famille> daddr <Adresse IP> <Déclaration>  
>> sudo nft add rule ip6 filter OUTPUT ip daddr 192.168.1.100 drop**

L’**affichage** **des règles** peut se faire avec …  
**>> sudo nft list table <Famille> <Table>  
>> sudo nft list table ip6 filter**

Pour **accepter la configuration IPv6 dynamique et la découverte de voisins** …  
**>> sudo rule ip6 <Table> <Chaine> icmpv6 type nd-neighbor-solicit <Déclaration>  
>> sudo nft add rule ip6 filter INPUT icmpv6 type nd-neighbor-solicit accept**

**>> sudo rule ip6 <Table> <Chaine> icmpv6 type nd-router-solicit <Déclaration>  
>> sudo nft add rule ip6 filter INPUT icmpv6 type nd-router-advert accept**

# Étude de cas

## Description du contexte

On dispose d'un serveur web hébergé chez un hébergeur.   
Le site reçoit plusieurs milliers de visiteurs par jour et aussi quelques pirates qui tentent des choses afin de contourner la sécurité.

Voici donc les règles iptables mises en place …

* Le service web est accessible à l’aide des protocoles HTTP et HTTPS ;
* Les sites web ont besoin, pour contacter d'autres services web, d'effectuer des requêtes en HTTPS seulement (en tant que client) – Ne pas oublier les requêtes DNS (port UDP 53) ;
* Le serveur est accessible à l’aide de SSH sur le port TCP 1022 ;
* Le serveur de courriel doit pouvoir contacter des serveurs NTP externes à et envoyer des courriels   
  sur le port TCP 587 ;
* Les paquets TCP de types NULL ou XMAS doivent être comptabilisés et bloqués ;
* Tout ce qui n'est pas explicitement autorisé doit être refusé ;
* Le serveur pourra envoyer des requêtes ping mais ne pas en recevoir.

Voici les règles iptables existantes …

**# On autorise la sortie et l'entrée sur le port HTTP (TCP 80)  
iptables -A INPUT -m tcp -p tcp --dport 80 -j ACCEPT**

**# On autorise la sortie du protocole HTTPS (TCP 443)  
iptables -A INPUT -m tcp -p tcp --dport 443 -j ACCEPT  
iptables -A OUTPUT -m tcp -p tcp --dport 443 -j ACCEPT**

**# On autorise la sortie et l'entrée pour le protocole SSH (TCP 1022)  
iptables -t filter -A INPUT -p tcp --dport 1022 -j ACCEPT  
iptables -t filter -A OUTPUT -p tcp --dport 1022 -j ACCEPT**

**# On autorise la sortie et l'entrée pour le protocole DNS (UDP 53)  
iptables -t filter -A OUTPUT -p udp --dport 53 -j ACCEPT  
iptables -t filter -A INPUT -p udp --dport 53 -j ACCEPT**

**# On autorise la sortie des requêtes pour le protocole NTP (TCP 123)  
iptables -t filter -A OUTPUT -p udp --dport 123 -j ACCEPT**

**# On autorise la sortie des mails vers le serveur de courriels (TCP 587)  
iptables -t filter -A OUTPUT -p tcp --dport 587 -j ACCEPT**

**# Refus des requêtes TCP XMAS pour éviter les balayages de ports  
iptables -A INPUT -p tcp --tcp-flags ALL ALL -j DROP**

**# Refus des requêtes TCP NULL pour éviter les balayages de ports  
iptables -A INPUT -p tcp --tcp-flags ALL NONE -j DROP**

**# On refuse tout le reste  
iptables -A OUTPUT -j DROP  
iptables -A INPUT -j DROP**

**Fichier de configuration /etc/nftables.conf**

## Sauvegarde des règles nftables

Pour sauver la configuration complète …  
**>> sudo nft list rulseset > nftables.rules**

Pour sauvegarder le contenu d’une table …  
**>> sudo nft list table <Table> > <Fichier>**  
**>> sudo nft list table filter > nftables.filter.rules**

**Remarque** …  
Le fichier peut avoir le suffixe que l’on souhaite.   
Pour rappel, le système d'extension n'existe pas sous Linux, ils sont totalement optionnels.

## Restauration des règles nftables

Pour restaurer les règles nftables, on utlise la commande nft avec le commutateur -f.  
Cette commande permet d'importer des règles depuis un fichier.  
Pour importer le contenu d’une table …  
**>> sudo nft list table <Table> < <Fichier>**  
**>> sudo nft -f nftables.rules  
>> sudo nft -f nftables.filter.rules**

## Application des règles au démarrage

Il faut sauvegarder la configuration nftables dans le fichier /root/nftables/iptables.rules.

Par la suite, il faut éditer le fichier d l’interface réseau …  
**>> sudo nano /etc/network/interfaces  
auto ens33  
iface ens33 inet static  
 address 192.168.1.100  
 netmask 255.255.255.0  
 pre-up nft -f /root/nftables/iptables.rules**

La dernière ligne débutant par pre-up … va exécuter la commande qui suit une fois que l'interface sera montée, assurant ainsi que les règles nftables soient restaurées au démarrage.

# Règles nftables

**table ip filter {  
 chain input {  
 type filter hook input priority 0;  
 ct state established,related accept   
 tcp dport ssh accept  
 tcp dport http accept   
 tcp dport https accept   
 tcp dport 1022 accept   
 tcp flags & (fin | syn | rst | psh | ack | urg) > urg counter packets 0 bytes 0**

**tcp flags & (fin | syn | rst | psh | ack | urg) < fin counter packets 0 bytes 0 # handle 9  
 icmp type echo-reply accept # handle 10  
 drop # handle 11**

**}**

**chain output {  
 type filter hook output priority 0;  
 ct state established,related accept # handle 13  
 tcp dport ssh accept  
 tcp dport https accept # handle 14  
 udp dport domain accept # handle 15  
 udp dport ntp accept # handle 16  
 tcp dport 587 accept # handle 17  
 icmp type echo-request accept # handle 18  
 drop # handle 19  
 }  
}**

# Référence

Document adapté du site …  
IT-connect

Publié sous licence Common Creative

1. Un compteur permet de comptabiliser à la fois le nombre total de paquets et le nombre total d'octets que l’interface a reçus ou envoyés depuis sa dernière réinitialisation.   
   Avec nftables, on doit spécifier explicitement un compteur pour chaque règle que l’on souhaite compter. [↑](#footnote-ref-1)