Traversée des tables et des chaînes

Ce chapitre décrit la façon dont les paquets traversent les différentes chaînes, et dans quel ordre. De même, il explique l'ordre dans lequel les tables sont traversées. Vous percevrez l'importance de ce fonctionnement plus loin, lors de l'écriture de vos propres règles. D'autres points seront examinés, liés à des éléments dépendants du noyau, car ils entrent également en considération dans ce chapitre. Entre autres, les différentes décisions de routage. C'est particulièrement utile si vous voulez écrire des règles pour **iptables** qui peuvent modifier les consignes/règles de routage des paquets, c-à-d. pourquoi et comment les paquets sont routés; le **DNAT** et le **SNAT** sont des exemples caractéristiques. Bien sûr, il ne faut pas oublier les bits de TOS.

# Généralités

Quand un paquet arrive pour la première fois dans un pare-feu, il rencontre le niveau matériel, puis il est recueilli par le pilote de périphérique approprié au sein du noyau. Ensuite, le paquet enchaîne une succession d'étapes dans le noyau, avant d'être envoyé à l'application adéquate (localement), ou expédié à un autre hôte - ou quoi que ce soit d'autre.

D'abord, analysons un paquet destiné à la machine locale. Il enchaîne les étapes suivantes avant d'être réellement délivré à l'application qui le reçoit :

**Tableau 6.1. Hôte local destinataire (votre propre machine)**

| **Étape** | **Table** | **Chaîne** | **Commentaire** |
| --- | --- | --- | --- |
| 1 |  |  | Sur le câble (ex. Internet) |
| 2 |  |  | Arrive sur l'interface (ex. eth0) |
| 3 | raw | PREROUTING | Cette chaîne sert normalement à modifier les paquets, i.e. changer les bits de TOS, etc. |
| 4 |  |  | Lors du contrôle de code de connexion comme indiqué dans le chapitre [La machine d'état](http://www.inetdoc.net/guides/iptables-tutorial/statemachine.html). |
| 5 | mangle | PREROUTING | Chaîne principalement utilisée pour modifier les paquets, i-e, changement de TOS, etc. |
| 6 | nat | PREROUTING | Cette chaîne sert principalement au DNAT. Évitez de filtrer dans cette chaîne puisqu'elle est court-circuitée dans certains cas. |
| 7 |  |  | Décision de routage, i.e. le paquet est-il destiné à notre hôte local, doit-il être réexpédié et où ? |
| 8 | mangle | INPUT | Ici, il atteint la chaîne INPUT de la table mangle. Cette chaîne permet de modifier les paquets, après leur routage, mais avant qu'ils soient réellement envoyés au processus de la machine. |
| 9 | filter | INPUT | C'est l'endroit où est effectué le filtrage du trafic entrant à destination de la machine locale. Notez bien que tous les paquets entrants et destinés à votre hôte passent par cette chaîne, et ceci quelle que soit leur interface ou leur provenance d'origine. |
| 10 |  |  | Processus/application local (i.e. programme client/serveur) |

Remarquez que cette fois, le paquet est transmis à travers la chaîne INPUT au lieu de la chaîne FORWARD. C'est parfaitement logique. Et c'est certainement la seule chose logique à vos yeux dans le parcours des tables et des chaînes pour le moment, mais si vous continuez d'y réfléchir, vous trouverez ceci de plus en plus clair au fur et à mesure.

À présent, analysons les paquets sortant de notre hôte local et les étapes qu'ils enchaînent.

**Tableau 6.2. Hôte local source (votre propre machine)**

| **Étape** | **Table** | **Chaîne** | **Commentaire** |
| --- | --- | --- | --- |
| 1 |  |  | Processus/application local (i.e. programme client/serveur) |
| 2 |  |  | Décision de routage. Quelle adresse source doit être utilisée, quelle interface de sortie, et d'autres informations nécessaires qui doivent être réunies. |
| 3 | raw | OUTPUT | C'est l'endroit où le traçage de connexion prend place pour les paquets générés localement. Vous pouvez marquer les connexions pour qu'elles ne soient pas tracées par exemple. |
| 4 |  |  | C'est ici que le traçage de connexion se situe pour les paquets générés localement, par exemple les changements d'état, etc. Voir le chapitre [La machine d'état](http://www.inetdoc.net/guides/iptables-tutorial/statemachine.html) pour plus d'information. |
| 5 | mangle | OUTPUT | C'est là où les paquets sont modifiés. Il est conseillé de ne pas filtrer dans cette chaîne, à cause de certains effets de bord. |
| 6 | nat | OUTPUT | Cette chaîne permet de faire du NAT sur des paquets sortant du pare-feu. |
| 7 |  |  | Décision de routage, comment les modifications des mangle et nat précédents peuvent avoir changé la façon dont les paquets seront routés. |
| 8 | filter | OUTPUT | C'est de là que les paquets sortent de l'hôte local. |
| 9 | mangle | POSTROUTING | La chaîne POSTROUTING de la table mangle est principalement utilisée lorsqu'on souhaite modifier des paquets avant qu'ils quittent la machine mais après les décisions de routage. Cette chaîne est rencontrée d'une part par les paquets qui ne font que transiter par le pare-feu, d'autre part par les paquets créés par le pare-feu lui-même. |
| 10 | nat | POSTROUTING | C'est ici qu'est effectué le SNAT. Il est conseillé de ne pas filtrer à cet endroit à cause des effets de bord, certains paquets peuvent se faufiler même si un comportement par défaut a été défini pour la cible **DROP**. |
| 11 |  |  | Sort par une certaine interface (ex. eth0) |
| 12 |  |  | Sur le câble (ex. Internet) |

Dans cet exemple, on suppose que le paquet est destiné à un autre hôte sur un autre réseau. Le paquet parcourt les différentes étapes de la façon suivante :

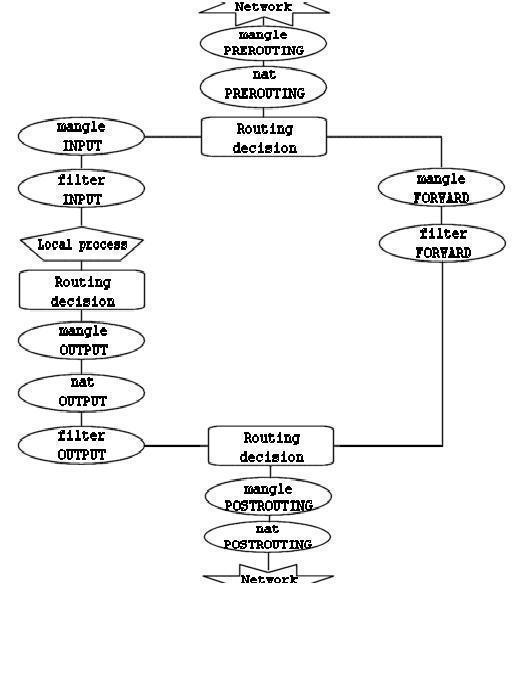
**Tableau 6.3. Paquets redirigés**

| **Étape** | **Table** | **Chaîne** | **Commentaire** |
| --- | --- | --- | --- |
| 1 |  |  | Sur le câble (ex. Internet) |
| 2 |  |  | Arrive sur l'interface (ex. eth0) |
| 3 | raw | PREROUTING | Ici vous pouvez placer une connexion qui ne sera pas interprétée par le système de traçage de connexion. |
| 4 |  |  | C'est ici que le traçage de connexion généré non-localement prend place, nous verrons celà dans le chapitre [La machine d'état](http://www.inetdoc.net/guides/iptables-tutorial/statemachine.html). |
| 5 | mangle | PREROUTING | Cette chaîne est typiquement utilisée pour modifier les paquets, i.e. changer les bits de TOS, etc. |
| 6 | nat | PREROUTING | Cette chaîne sert principalement à réaliser du DNAT. Le SNAT est effectué plus loin. Evitez de filtrer dans cette chaîne car elle peut être court-circuitée dans certains cas. |
| 7 |  |  | Décision de routage, c-à-d. le paquet est-il destiné à votre hôte local, doit-il être redirigé et où ? |
| 8 | mangle | FORWARD | Le paquet est alors envoyé à la chaîne FORWARD de la table mangle. C'est utile pour des besoins très spécifiques, lorsque l'on souhaite modifier des paquets après la décision de routage initiale, mais avant la décision de routage finale effectuée juste avant l'envoi du paquet. |
| 9 | filter | FORWARD | Le paquet est routé vers la chaîne FORWARD. Seuls les paquets réexpédiés arrivent ici, et c'est ici également que tout le filtrage est effectué. Notez bien que tout trafic redirigé passe par ici (et pas seulement dans un sens), donc vous devez y réfléchir en rédigeant vos règles. |
| 10 | mangle | POSTROUTING | Cette chaîne est employé pour des formes particulières de modification de paquets, que l'on veut appliquer postérieurement à toutes les décisions de routage, mais toujours sur cette machine. |
| 11 | nat | POSTROUTING | Cette chaîne est employée principalement pour le SNAT. Évitez de faire du filtrage ici, car certains paquets peuvent passer cette chaîne sans être vérifiés. C'est aussi l'endroit où l'on fait du masquerading (masquage d'adresse). |
| 12 |  |  | Sort par l'interface de sortie (ex. eth1). |
| 13 |  |  | Sort de nouveau par le câble (ex. LAN). |

Comme vous pouvez le constater, il y a de nombreuses étapes à franchir. Un paquet peut être arrêté dans n'importe quelle chaîne d'**iptables**, et même ailleurs s'il est malformé. Pourtant, il est intéressant de se pencher sur le sort du paquet vu par **iptables**. Remarquez qu'aucune chaîne ou table spécifique n'est définie pour des interfaces différentes, ou quoi que ce soit de semblable. La chaîne FORWARD est systématiquement parcourue par les paquets qui sont redirigés par l'intermédiaire de ce pare-feu/routeur.

|  |  |
| --- | --- |
| [Attention] | **Attention** |
| N'utilisez pas la chaîne INPUT pour filtrer dans le scénario précédent ! INPUT n'a de sens que pour des paquets destinés à votre hôte local, autrement dit qui ne seront routés vers aucune autre destination. |

Maintenant, vous avez découvert comment les différentes chaînes sont traversées selon trois scénarios distincts. On peut en donner une représentation graphique :



Pour être plus clair, ce dessin mérite quelques explications. Si un paquet atteignant la première décision de routage n'est pas destiné à la machine locale, il sera orienté vers la chaîne FORWARD. En revanche, s'il est destiné à une adresse IP que la machine écoute, ce paquet sera envoyé vers la chaîne INPUT, et donc à la machine locale.

Il est important de remarquer que même si des paquets sont destinés à la machine locale, leur adresse de destination peut être modifiée à l'intérieur de la chaîne PREROUTING par une opération de NAT. En effet, puisque ceci a lieu avant la première décision de routage, le paquet ne sera examiné qu'après un éventuel changement. A cause de cette particularité, le routage peut être altéré avant que la décision de routage ne soit prise. Notez bien que *tous* les paquets transiteront par l'un ou l'autre des chemins de ce dessin. Si vous réalisez du DNAT sur un paquet pour le renvoyer sur le réseau duquel il provient, il continuera malgré tout sa route à travers les chaînes restantes jusqu'à ce qu'il retourne sur le réseau externe.

|  |  |
| --- | --- |
| [Astuce] | **Astuce** |
| Si vous pensez avoir besoin d'informations supplémentaires, vous pouvez utiliser le script [rc.test-iptables.txt](http://www.inetdoc.net/guides/iptables-tutorial/rctest-iptablestxt.html). Ce script de test devrait vous procurer des règles suffisantes pour expérimenter et comprendre de quelle façon les tables et les chaînes sont traversées. |

# La table Mangle

Comme il a déjà été précisé, le rôle principal de cette table devrait être de modifier des paquets. En d'autres termes, vous pouvez utiliser en toute liberté les correspondances de la table mangle, qui permettent de changer les champs de TOS (type de service), et d'autres.

|  |  |
| --- | --- |
| [Attention] | **Attention** |
| Vous avez été suffisamment prévenus de ne pas utiliser cette table pour effectuer du filtrage; de même, les opérations de **DNAT**, **SNAT** ou de **masquerading** ne fonctionnent pas dans cette table. |

Les cibles suivantes sont valides uniquement dans la table mangle. Elles ne doivent pas être utilisées en dehors de cette table.

* TOS
* TTL
* MARK
* SECMARK
* CONNSECMARK

La cible **TOS** permet de définir et/ou modifier le champ de Type de Service d'un paquet. C'est utile pour définir des stratégies réseau concernant le choix de routage des paquets. Sachez que, d'une part ceci n'a pas été perfectionné, d'autre part ce n'est pas vraiment implémenté sur Internet car la majorité des routeurs ne se préoccupent pas de ce champ, et quelquefois même, ils adoptent un comportement erroné. Bref, ne configurez pas ce champ sur les paquets qui naviguent sur Internet, sauf si vous souhaitez leur appliquer des décisions de routage, avec iproute2.

La cible **TTL** permet de modifier le champ durée de vie ou TTL (Time To Live) d'un paquet. Il est possible par exemple de spécifier aux paquets d'avoir un champ TTL spécifique. Ceci peut se justifier lorsque vous ne souhaitez pas être rejeté par certains Fournisseurs d'Accès à Internet (FAI) trop indiscrets. En effet, il existe des FAI qui désapprouvent les utilisateurs branchant plusieurs ordinateurs sur une même connexion, et de fait, quelques-uns de ces FAI sont connus pour vérifier si un même hôte génère différentes valeurs TTL, supposant ainsi que plusieurs machines sont branchées sur la même connexion.

La cible **MARK** permet d'associer des valeurs de marquage particulières aux paquets. Elles peuvent ensuite être identifiées par les programmes **iproute2** pour appliquer un routage différent en fonction de l'existence ou de l'absence de telle ou telle marque. On peut ainsi réaliser de la restriction de bande passante et de la gestion de priorité (Class Based Queuing).

La cible **SECMARK** peut être utilisée pour placer des marques dans un contexte de sécurité sur des paquets dans SELinux ou tout autre système de sécurité capable de gérer ces marques.

**CONNSECMARK** sert à copier un contexte de sécurité vers ou depuis un simple paquet ou vers une connexion complète. Elle est utilisée par SELinux ou autre système de sécurité pour affiner cette sécurité au niveau connexion.

# La table Nat

Cette table devrait être utilisée seulement pour effectuer de la traduction d'adresse réseau (NAT) sur différents paquets. Autrement dit, elle ne devrait servir qu'à traduire le champ de l'adresse source d'un paquet ou celui de l'adresse destination. Précisons à nouveau que seul le premier paquet d'un flux rencontrera cette chaîne. Ensuite, les autres paquets subiront automatiquement le même sort que le premier. Voici les cibles actuelles capables d'accomplir ce genre de choses :

* DNAT
* SNAT
* MASQUERADE
* REDIRECT

La cible **DNAT** est généralement utile dans le cas où vous détenez une adresse IP publique et que vous désirez rediriger les accès vers un pare-feu localisé sur un autre hôte (par exemple, dans une zone démilitarisée ou DMZ). Concrètement, on change l'adresse de destination du paquet avant de le router à nouveau vers l'hôte désigné.

La cible **SNAT** est quant à elle employée pour changer l'adresse de source des paquets. La plupart du temps, vous dissimulerez votre réseau local ou votre DMZ, etc. Un très bon exemple serait donné par un pare-feu pour lequel l'adresse externe est connue, mais qui nécessite de substituer les adresses IP du réseau local avec celle du pare-feu. Avec cette cible, le pare-feu effectuera automatiquement sur les paquets du **SNAT** dans un sens et du **SNAT inverse** dans l'autre, rendant possible les connexions d'un réseau local sur Internet. A titre d'exemple, si votre réseau utilise la famille d'adresses 192.168.0.0/masque\_réseau, les paquets envoyés sur Internet ne reviendront jamais, parce que l'IANA (institut de régulation des adresses) a considéré ce réseau (avec d'autres) comme privé, et a restreint son usage à des LANs isolés d'Internet.

La cible **MASQUERADE** s'utilise exactement de la même façon que la cible **SNAT**, mais la cible **MASQUERADE** demande un peu plus de ressources pour s'exécuter. L'explication vient du fait que chaque fois qu'un paquet atteint la cible **MASQUERADE**, il vérifie automatiquement l'adresse IP à utiliser, au lieu de se comporter comme la cible **SNAT** qui se réfère simplement à l'unique adresse IP configurée. Par conséquent, la cible **MASQUERADE** permet de faire fonctionner un système d'adressage IP dynamique sous DHCP, que votre FAI devrait vous procurer pour des connexions à Internet de type PPP, PPPoE ou SLIP.

# La table Raw

La table Raw est principalement utilisée pour placer des marques sur les paquets qui ne doivent pas être vérifiés par le système de traçage de connexion. Ceci est effectué en utilisant la cible NOTRACK sur le paquet. Si une connexion rencontre une cible NOTRACK, conntrack ne tracera pas cette connexion. Ceci était impossible à résoudre sans l'ajout d'une nouvelle table, car aucune des autres tables n'est appelée jusqu'à ce que conntrack ait été lancé sur les paquets, et ait été ajouté aux tables conntrack, ou vérifié sur une connexion existante. À ce sujet, voir le chapitre [La machine d'état](http://www.inetdoc.net/guides/iptables-tutorial/statemachine.html).

Cette table ne supporte que les chaînes PREROUTING et OUTPUT. Aucune autre chaîne n'est nécessaire, car c'est le seul endroit où vous pouvez opérer sur les paquets avant qu'ils soient vérifiés par le traçage de connexion.

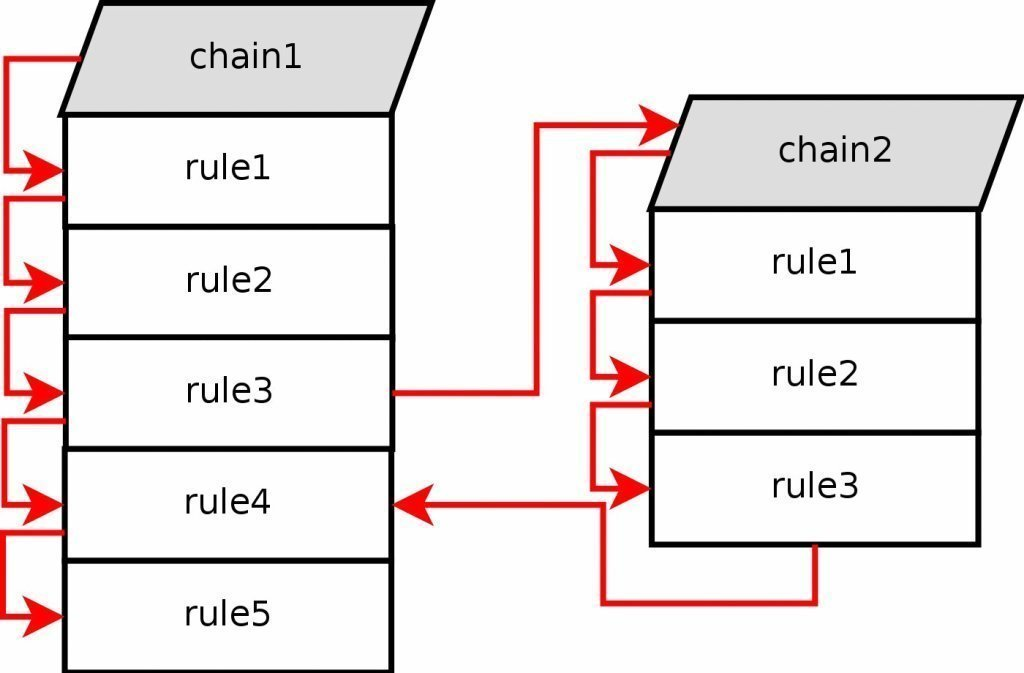
|  |  |
| --- | --- |
| [Note] | **Note** |
| Pour que cette table fonctionne, le module iptable\_raw doit être chargé. Il sera chargé automatiquement si iptables est lancé avec l'option -t raw, et si le module est disponible. |
| [Note] | **Note** |
| La table raw est relativement récente dans iptables et le noyau. Elle peut ne pas être disponible dans les premiers noyaux 2.6 et les 2.4 en dehors d'un patch. |

# La table Filter

La table filter sert principalement à filtrer les paquets. On peut établir une correspondance avec des paquets et les filtrer comme on le désire. C'est l'endroit prévu pour intervenir sur les paquets et analyser leur contenu, c'est-à-dire les détruire (avec la cible **DROP**) ou les accepter (avec **ACCEPT**) suivant leur contenu. Bien entendu, il est possible de réaliser préalablement du filtrage ; malgré tout, cette table a été spécialement conçue pour ça. Presque toutes les cibles sont utilisables dans celle-ci. D'autres informations seront données sur la table filter, cependant vous savez maintenant que c'est l'emplacement idéal pour effectuer votre filtrage principal.

# Chaînes utilisateurs spécifiques

Si un paquet pénètre dans une chaîne comme la chaîne INPUT de la table Filter, vous pouvez spécifier une règle (saut) vers une chaîne différente dans la même table. La nouvelle chaîne doit être spécifique utilisateur, elle ne peut pas être générique comme les chaînes INPUT et FORWARD par exemple. Si nous considérons un pointeur vers une règle d'une chaîne à exécuter, ce pointeur passera de règle en règle, du sommet à la base, jusqu'à ce que la traversée de la chaîne soit close par une cible ou la même chaîne (i-e. FORWARD, INPUT, etc.). Une fois ceci fait, la stratégie par défaut de la chaîne générique sera appliquée.



Si une des ces chaînes pointe vers une autre chaîne spécifique utilisateur, le pointeur sautera cette chaîne et démarrera la traversée des chaînes depuis le sommet jusqu'à la base. Par exemple, voir comment la règle saute de l'étape 3 vers la chaîne 2 dans l'image ci-dessus. Le paquet sélectionne les correspondances contenues dans la règle 3, et cible/saute vers la chaîne 2.

|  |  |
| --- | --- |
| [Note] | **Note** |
| Les chaînes spécifiques utilisateur ne peuvent pas avoir de stratégie par défaut. Seules les chaînes génériques le peuvent. Ceci peut être contourné en ajoutant une simple règle à la fin de la chaîne, ainsi elle aura une stratégie par défaut. Si aucune règle n'est sélectionnée dans une chaîne spécifique utilisateur, le comportement par défaut sera celui de la chaîne d'origine. Comme vu dans l'image ci-dessus, la règle saute de la chaîne 2 et retourne vers la chaîne 1 règle 4. |

Chaque règle dans une chaîne spécifique utilisateur est traversée jusqu'à ce que, soit une des règles corresponde -- alors la cible spécifie si la traversée se termine ou continue -- soit la fin de la chaîne est atteinte. Si la fin de la chaîne spécifique utilisateur est atteinte, le paquet est envoyé en retour vers la chaîne qui l'invoque. Cette chaîne peut être, soit une chaîne spécifique utilisateur soit une chaîne générique.