# M2 e-secure

# Réseaux - Filtrage

# Netfilter et Iptables

Bureau S3-354
Jean.Saquet@unicaen.fr
http : saquet.users.greyc.fr/M2/rezo

# Filtrage - Introduction

Internet a été conçu pour que tout paquet puisse transiter, sauf incident technique, d'une adresse IP à une autre.
Au début, tout fantaisiste qui essayait d'abuser de cette possibilité se faisait remettre à l'ordre par la communauté des utilisateurs.
Avec la généralisation de l'Internet, ceci n'a plus été possible, d'où la nécessité de système automatiques permettant de se protéger des attaques ou même mauvaises manipulations involontaires.

# Attaques - motivations

- Vol de données
- Usurpation d'identité
- Nuire au fonctionnement (ex : Deni de service)
- Modification des données d'un serveur
- Introduction dans un système pour rebond
- Jeu
- ...

# Attaques - moyens

- Accès physique aux machines
- Ecoute du réseau (ex : Wi-Fi)
- Failles des systèmes ou logiciels
- Buffer overflow
- Inondation
- Scan de ports pour repérer un logiciel attaquable
- ...

# Attaques - parades

- Mise à jour des systèmes et logiciels
- Accès sécurisés aux locaux
- Ne pas s'absenter avec session ouverte
- Filtrage pour éliminer la plupart des attaques

Le filtrage nécessite l'analyse des éléments de protocole. Il peut agir au niveau transport ou au niveau application. Pour le niveau application, le filtrage doit être associé aux applications concernées (exemple : analyse du contenu d'un mail pour détecter les virus connus).

# Filtrage niveau transport

Le filtrage porte sur les couches transport, réseau, éventuellement physique, et simultanément.
Pour une question d'efficacité, on analyse les trois couches en même temps.

Les critères de filtrage portent sur les différents paramètres des en-têtes des couches : adresses, ports, paramètres propres à TCP, …
Le filtrage porte alors séparément sur chaque paquet arrivant à, partant de ou traversant la machine surveillée.

# Filtrage avec suivi, ou surveillant les répétitions

Il peut être intéressant de surveiller les successions de paquets plutôt que les paquets isolés pour par exemple :

- Repérer les tentatives d'inondation
- Ne laisser passer que des flots de données relatifs à une connexion dont l'ouverture a été autorisée.
- Laisser passer les réponses aux requêtes qui ont été autorisées

Un bon système de filtrage doit permettre ceci.

# Translation d'adresses

En IPv4, on utilise beaucoup les adresses privées et les translations d'adresses pour cause de manque d'adresses IP.
Ces adresses sont invisibles de l'extérieur, mais pas de l'intérieur de l'entreprise.

Il est donc fortement conseillé de ne pas compter que sur les adresses privées comme moyen de protection.

# Où filtrer ?

Une passerelle entre l'Internet et le réseau interne de l'entreprise est évidemment un lieu privilégié pour y placer le filtre, car toutes les communications entre l'entreprise et l'extérieur y passent.

Mais ça ne suffit pas, car il existe souvent des risques venant de l'intérieur. Voir quelques exemples.

D'où l'intérêt des « auto-firewalls » qui surveillent l'entrée des machines sensibles (serveurs de données confidentielles par exemple)

# Netfilter - Introduction

Netfilter est le système de filtrage des éléments de protocole incorporé au noyau Linux depuis la version 2.4.

Il est très efficace et rapide, plus complet que la version précédente. Son mécanisme et sa configuration sont un peu moins évidents qu'avec l'ancien ipchains, mais, après une phase d'adaptation, il s'avère plus commode.

Il rivalise aisément avec de nombreuses réalisations commerciales. Parfois, elles l'utilisent !

# Netfilter – filtrage et NAT

Classiquement, Netfilter peut filtrer le paquets comme les transformer (translation d'adresse ou/et de port).

Les deux fonctions sont différentes, les tables qui contiennent les règles ne sont pas les mêmes dans les deux cas.

Deux "HOWTO" fournissent les notices pour ces deux fonctions.

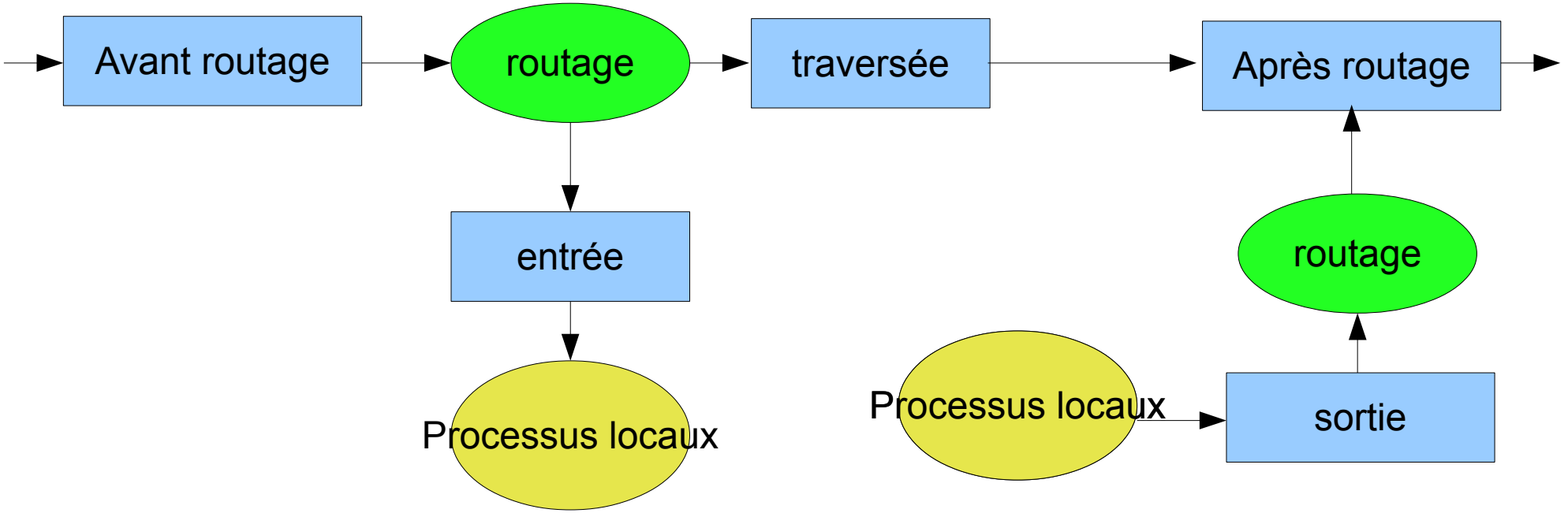
# Netfilter - principe

Les datagrammes IP peuvent être analysés à différents stades de leur transit dans une machine :

- à l'arrivée, avant toute opération de routage
- après décision de routage, et avant remise à la couche supérieure si le dg est arrivé à destination, ou avant envoi à la passerelle suivante si le dg doit être forwardé
- après la création du dg si ce dernier a été généré par la machine (à partir d'une couche supérieure)
- après toutes les opérations de routage si le dg sort de la machine

# Netfilter - schéma

En bleu : points de filtrage possibles d'un dg et/ou d'action sur ce dernier




Ceci ne dépend pas de l'interface d'entrée du dg ni de celle de sortie

# Netfilter – tables et chaînes

Netfilter utilise des tables, notamment :

- filter qui gère les entrées, sorties, traversées
- nat, qui gère les transformations d'adresse (et de port)

Ces tables comportent des chaînes de règles, se positionnant à un endroit du schéma précédent

# Netfilter – tables et chaînes

De manière plus précise :

filter utilise les chaînes INPUT, OUTPUT et FORWARD
nat utilise PREROUTING, POSTROUTING et OUTPUT.
Il y a aussi d'autres tables (voir plus loin)
Ces cinq chaînes se positionnent aux endroits du schéma précédent (position évidente d'après leurs noms)

# Comparaison avec ipchains

Dans la version précédente (ipchains), il n'y avait que les chaînes INPUT, OUTPUT, FORWARD, systématiquement invoquées lors de l'arrivée, du départ ou du transit d'un datagramme.

La configuration de Netfilter par iptables est donc sensiblement différente de celles d'ipchains.

# Iptables, utilisation

Si on se limite à la table "filter", on devra définir des règles dans les chaînes :

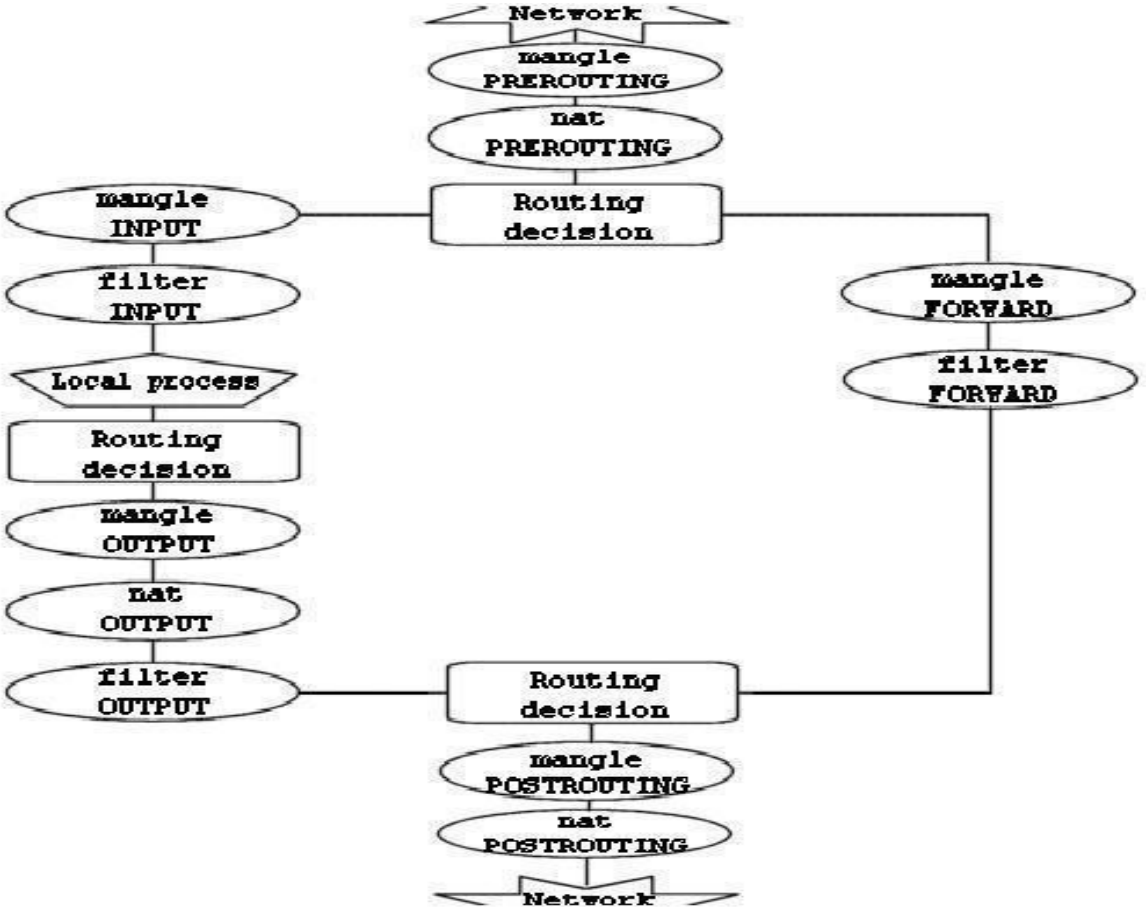
- INPUT pour les paquets destinés à la machine
- OUTPUT pour les paquets issus de la machine
- FORWARD pour les paquets traversant la machine.

Pour la table nat, on utilisera les chaînes PREROUTING, POSTROUTING, éventuellement OUTPUT

# Autres tables

Mangle : permet de modifier certains champs des paquets (TTL, type of service). Peut être utilisée en conjonction avec des règles de routage spécifiques.
Raw : marquage spécifique des paquets pour éviter leur prise en compte par conntrack (voir plus loin)

# Le vrai schéma !



# Iptables, syntaxe

Iptables prend en paramètre une opération :

-N ou -X ou -L ou -F pour créer, détruire, lister les règles de, éliminer les règles de, une chaîne.
-A ou -D pour ajouter ou supprimer une règle dans une chaîne
-P pour définir la politique par défaut d'une chaîne

# Iptables, syntaxe

Pour ajouter une règle à une chaîne :

Iptables [-t table] -A chaine [-p protocole]
[-s source] [-d destination]
[--sport portsource] [--dport portdestination]
[-i interfsource] [-o interfdest] [-j cible]
par défaut, table = filter
cible définit l'opération à appliquer au paquet :
ACCEPT, DROP, REJECT
SNAT, DNAT, MASQUERADE
...

# Iptables, exemples

Iptables -A INPUT -p TCP -s 192.168.1.0/24 --dport 80 -j ACCEPT
accepte les connexions sur le port 80 (http) de la machine, venant d'une machine d'adresse commençant par 192.168.1
Iptables -t nat -A postrouting -s 192.168.0.0/24 -o eth1 -j MASQUERADE
Opère la translation d'adresse pour tout paquet émis par une machine 192.168.0.xx et sortant par l'interface eth1

# Concordance d'état (conntrack)

Suivi de connexion -m state
NEW paquet engendrant une nouvelle connexion
ESTABLISHED : le contraire
RELATED : relatif à une connexion (ex err icmp, connexion de données ftp, ...)
INVALID : paquets non identifiés

Exemple : suivi d'état de la connexion TCP
Mais vaut aussi pour UDP ou ICMP

# Conntrack, suite

Le suivi de connexion permet l'analyse plus fine des paquets afin, par exemple, de s'adapter à des protocoles ouvrant d'autres connexions (FTP, SIP, …)

Conntrack est toutefois un peu coûteux. On peut parfois utiliser des marquages avec la table raw pour s'en passer (cf. exemples en TD).

# Iptables, réponses

Avec iptables, une seule règle pour que les réponses aux requêtes acceptées puissent passer en sens inverse :
iptables -A input –match state –state established, related -j accept
(autorise les paquets entrants en réponse aux requêtes qu'on a laissées sortir – fonctionne pour TCP et UDP)

# Iptables, filtrage fin

-syn : pour spécifier uniquement les demandes de connexion TCP
-icmp-type : filtrage fin des paquets ICMP
-m pour "match"
    mac : adresse mac
    limit : nb max de concordances /sec
(par ex pour n'agir que sur quelques paquets/h (log) ou bien pour éviter les attaques répétées)
...

# Iptables, cibles

Chaînes utilisateur
-j <nom chaine>
renvoie l'analyse aux règles de la chaîne indiquée

LOG log des paquets

REJECT renvoie une erreur icmp :
"port unreachable" (par défaut)

# Iptables, nat

On peut translater la source, ou la destination (SNAT, DNAT)
MASQUERADE translate la source pour des paquets avec adresses assignées dynamiquement

# Ip6tables

Même chose, moins tout ce qui concerne les translations, devenues inutiles...
… bien que, dans certains cas de figure particuliers …