CFGM: Sistemes Microinformàtics i Xarxes

M6: Seguretat Informàtica

UF5 Tallafocs: Part 2 (IPtables)

Prof. Manel Orós. Darrera actualització 1/2019

IPTables - Index

0. Introducció

Tallafocs de capa de xarxa o de filtrat de paquets.

Inspeccionen els **paquets rebuts/enviats** i comproven si encaixen en les regles, aquestes són aplicades per a cada paquet i després el reenvia o descarta. Encamina tràfic entre la Xarxa externa i la interna.

Normalment se sol configurar perquè tingui en compte:

Adreça d'origen / destí de les dades.

Protocol de sessió de les dades. TCP, UDP o ICMP.

Si el paquet és l'inici d'una petició de connexió.

El port d'aplicació d'origen/destinació del servei desitjat.

Aquest tipus de filtrat és molt ràpid i gairebé totalment transparent per als usuaris.

http://www.spi1.nisu.org/recop/al02/chaman/index.html

1. Iptables

Netfilter és un entorn de treball **associat al nucli de Linux** que permet interceptar i manipular paquets de xarxa.

El component de Netfilter més important i més popular és **IPTABLES**. **Iptables** és una eina de tallafocs que permet no només filtrar paquets sinó també fa la traducció d'adreces de xarxa (**NAT**) per a **Ipv4** i mantenir registres de **log**.

Iptables ens permet configurar regles. Per aplicar aquestes regles utilitzem la comanda iptables (mitjançant CLI) amb la que podem afegir, esborrar o modificar regles directament a la **línia de comandes**, però el més còmode és fer un **script que s'executi a l'inicialitzar el firewall, o fer servir altres eines (ufw** → Uncomplicated Firewall, Shorewall, **fwbuilder**, **NuFW o webmin** http://www.trbailey.net/tech/iptables.html**).**

http://rlworkman.net/howtos/iptables/spanish/iptables-tutorial.html
http://blog.desdelinux.net/como-iniciar-reglas-de-iptables-automaticamente/

1. Iptables

Com hem vist abans podem escriure aquest script a mà, però també pot ser interessant preparar-lo amb una eina d'alt nivell com **fwbuilder**.

Podem crear les regles simplement arrossegant i deixant anar accions en els objectes. Uns quants menús contextuals poden canviar la condició (negar-la, per exemple). A continuació, haurà de triar l'acció i configurar-la.

En quant a IPv6, **pot crear dos conjunts de regles diferents per IPv4 i IPv6, o crear només una i deixar que fwbuilder tradueixi les regles segons les adreces assignades als objectes.**

aptitude install fwbuilder

Fwbuilder pot generar un script de configuració del tallafocs segons les regles que s'han definit.

http://debian-handbook.info/browse/es-ES/stable/sect.firewall-packet-filtering.html

1. Iptables

Les regles s'agrupen en cadenes i cada cadena és una llista ordenada de regles.

Les cadenes s'agrupen en taules i cada taula està associada a un tipus diferent de processament de paquets.

Un paquet va passant per cada regla de la llista fins que compleix alguna regla, llavors la regla s'activa i s'executa l'acció indicada a la regla (**L'ordre és determinant, ja que no executarà la resta de regles**).

Si arriba al final de la cadena de regles sense que es verifiqui cap, s'aplica la política general de la cadena (no és més que una regla que es correspon a una de les dues polítiques).
iptables -P cadena objectiu

#iptables -P INPUT DROP → política restrictiva. Indiquem a iptables, que la política per defecte (-P) per a tot el que vulgui entrar al nostre ordinador (INPUT) és ignorar, no fer-li cas (DROP)

#iptables -P INPUT ACCEPT → política permissiva

1. Iptables

Política permissiva.

Facilita molt la gestió del firewall. Hem de preocupar-nos de protegir aquells ports o adreces que ens interessa; la resta no importa tant i es deixa passar.

L'únic problema que podem tenir és que no controlem què és el que està obert i si instal·lem qualsevol software que ha d'obrir un port potser ens despistem i no actualitzem el firewall.

http://www.adminso.es/index.php/Iptables_-_gu%C3%ADa_r%C3%A1pida

1. Iptables

Política Restrictiva.

Si la política per defecte és DENEGAR, a no ser que ho permetem explícitament, el firewall es converteix en un autèntic mur infranquejable. Ara, hi ha més feina per preparar el firewall i cal tenir molt clar com funciona el sistema (sigui amb iptables o sigui el que sigui) i què és el que cal obrir.

http://www.adminso.es/index.php/Iptables_-_gu%C3%ADa_r%C3%A1pida

2. Taules

En IPtables tenim 4 taules incorporades: **raw, nat, mangle i filter.** Generalment n'hi ha prou amb les taules **filter i nat**. Però es poden crear taules especialitzades i emmagatzemar-les al directori /lib/modules/<versió-kernel>/kernel/net/ipv4/netfilter

Filter→**taula per defecte**, decideix si un paquet continua.

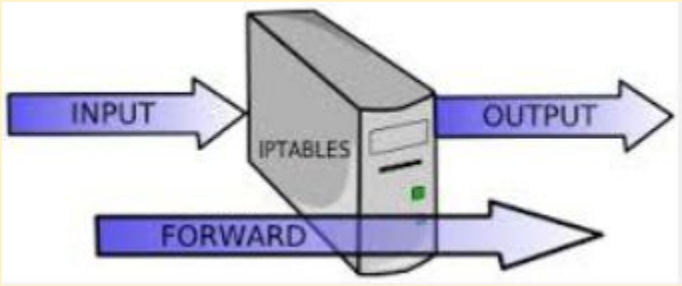
Nat→traducció d'adreces **(NetWork Address Translation)** per IPv4.

Mangle → dissenyada per **manipular paquets** o **marcar-los**. Cadenes predefinides: PREROUTING, POSTROUTING, OUTPUT, INPUT i FORWARD.

Raw → Aquesta taula **permet marcar paquets que no seran controlats** amb estat de connexió. Per a això farem servir l'objectiu NOTRACK i ho farem en les cadenes PREROUTING o OUTPUT.

http://debian-handbook.info/browse/es-ES/stable/sect.firewall-packet-filtering.html
https://www.frozentux.net/iptables-tutorial/spanish/iptables-tutorial.html

3. Cadenes



Cadascuna d'aquestes **taules** té un grup de **cadenes internes incorporades** que corresponen a les accions que es porten a terme en el filtratge de la xarxa.

Les **cadenes** internes per a la **taula filter** generalment són les següents :

* **INPUT** (entrada): s'aplica als paquets rebuts a través de la interface de xarxa. Analitza paquets rebuts per una interfície.

* **OUTPUT** (sortida): serveix per paquets enviats mitjançant la mateixa NIC que ha rebut els paquets. Analitza paquets que seran enviats per una NIC.

* **FORWARD** (redirigir): analitza paquets que el travessessin per una de les seves interfícies per enviar-les per una altra (paquets de pas).

Que normalment són aplicades per exemple amb els **objectius ACCEPT, DROP** (ignorar, el descarta sense efectuar cap acció més) i **REJECT** (rebutjar), funciona com Drop, però retorna un missatge d'error a la màquina que havia enviat el paquet).

http://www.ite.educacion.es/formacion/materiales/85/cd/linux/m6/cortafuegos_iptables.html

3. Cadenes

Les **cadenes** internes per a la **taula nat** generalment són les següents :

* **PREROUTING**: altera els paquets quan aquests arriben a la NIC. Modifica els paquets rebuts per una interfície traduint adreces de destinació DNAT

* **POSTROUTING**: altera paquets quan aquests són enviats per la NIC. Modifica els paquets abans d'enviar-se a una NIC traduint adreces d'origen SNAT

* **OUTPUT** Explicat abans. Paquets generats localment.

Objectius vàlids en aquesta taula son **SNAT** (SourceNAT traducció d'IP d'origen i només és vàlid per la cadena de POSTROUTING a la taula nat), **DNAT** (com SNAT però per Ips Destí, vàlid per les cadenes POSTROUTING i OUTPUT ala taula nat), **MASQUERADE** (similar a SNAT però per a IP dinàmica DHCP) o **REDIRECT** (serveix per redirigir paquets, molt recomanable per a proxies transparents funciona per les cadenes PREROUTING i OUTPUT, a la taula nat.

Exemple → iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT –to-ports 8080).

http://sw-libre.blogspot.com.es/2011/10/hola-basandome-en-la-documentacion-de.html

4. Regles

Exemple funcionament → un paquet que s'origina en la nostra xarxa local i va destinat a una màquina d'Internet:

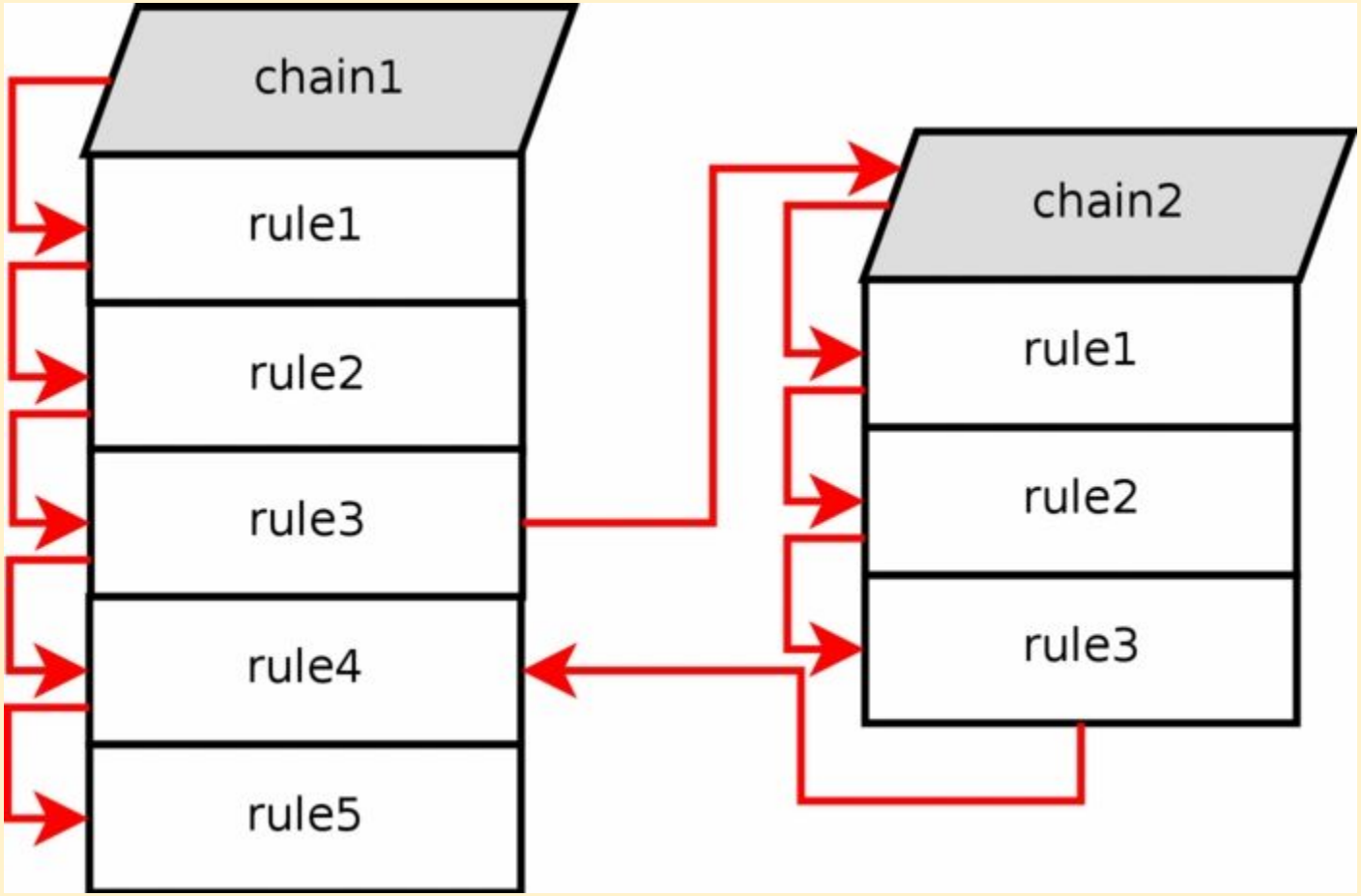
- El paquet entra al tallafocs per una interfície de xarxa, per tant, primer es comprovarien les regles de la **cadena PREROUTING**.

- A continuació es comprovarien les regles de la **cadena FORWARD**, ja que el paquet no va destinat a un procés del tallafocs, sinó que va a travessar, sortint per la interfície que el connecta amb la xarxa externa.

- Finalment, si el paquet no ha estat filtrat i segueix endavant, abans de sortir del tallafocs per l'altra interfície de xarxa, es comproven les regles de la **cadena POSTROUTING**

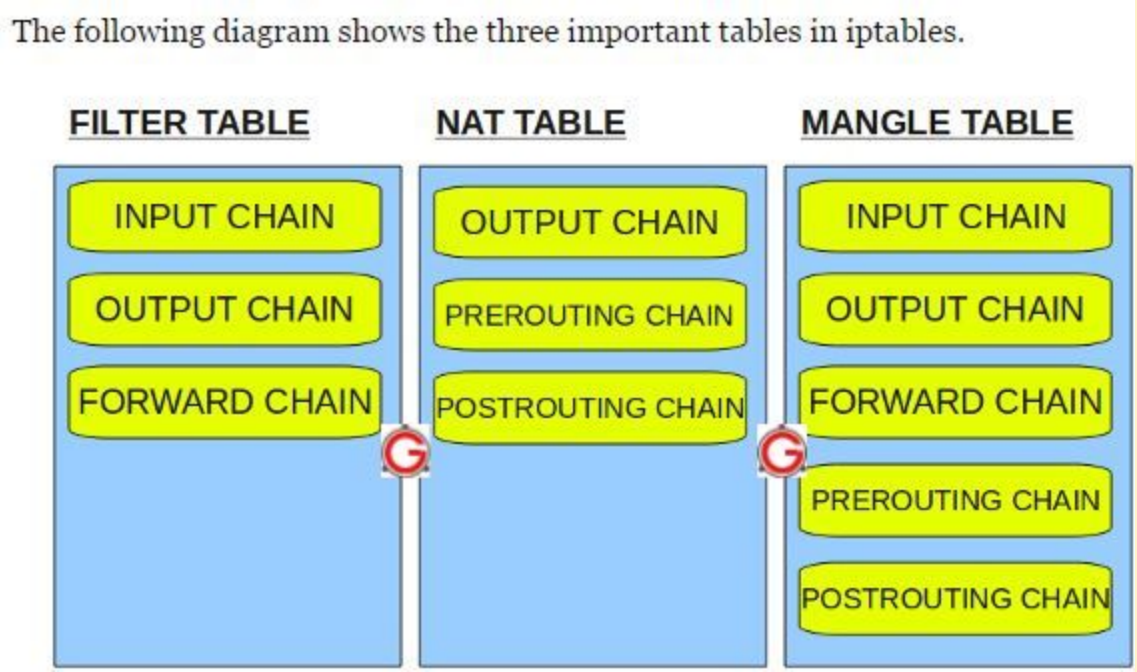
Nota important: al definir les regles de iptables cal utilitzar només adreces IP. No és correcte utilitzar noms fqdn's (noms amb el domini complet), ja que iptables comença abans que qualsevol servei DNS, de manera que es poden donar situacions errònies.

4. Regles i cadenes



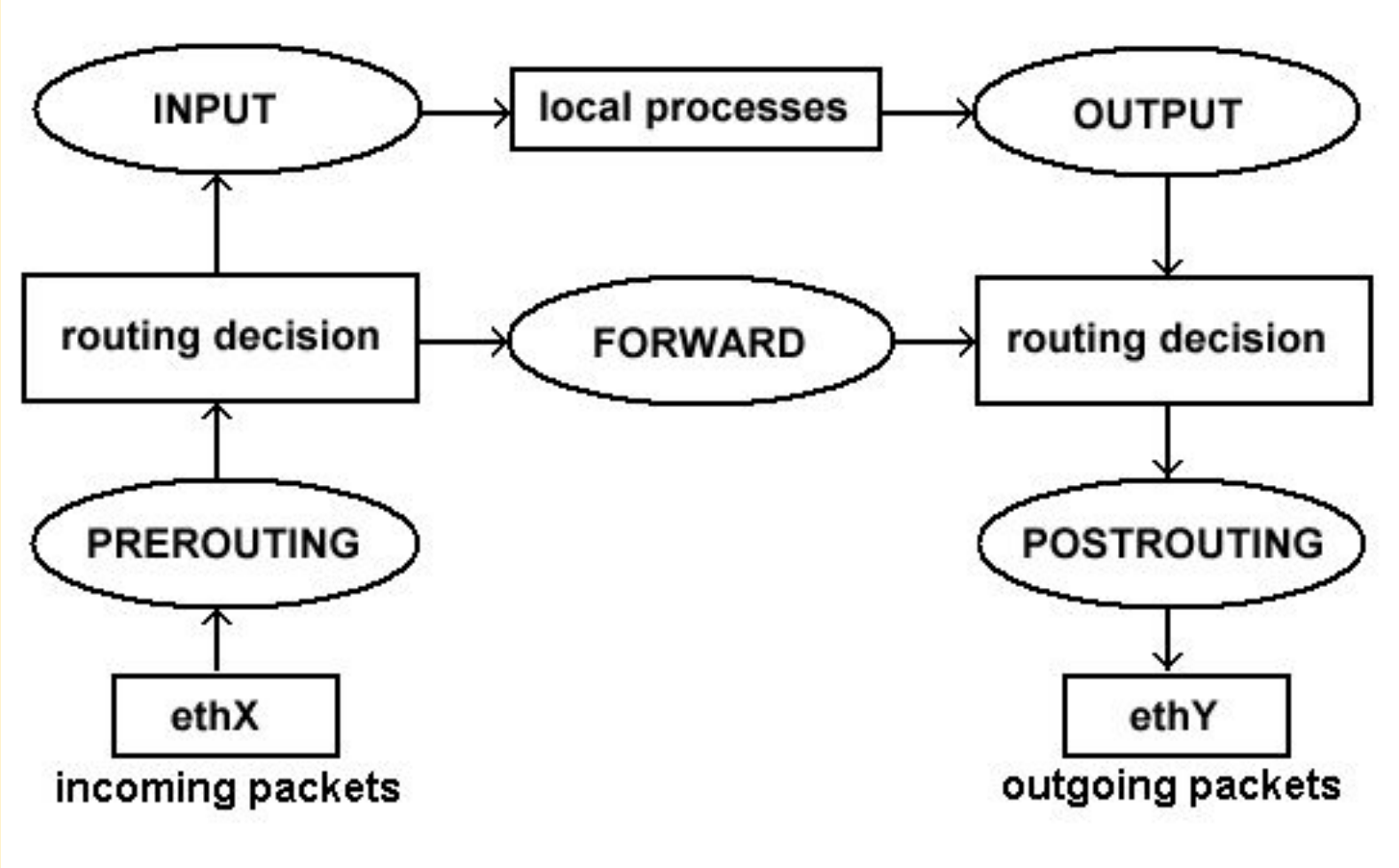
Font: http://kenfat-tech.blogspot.com/2016/07/iptables.html

4. Regles i cadenes



Font: http://kenfat-tech.blogspot.com/2016/07/iptables.html

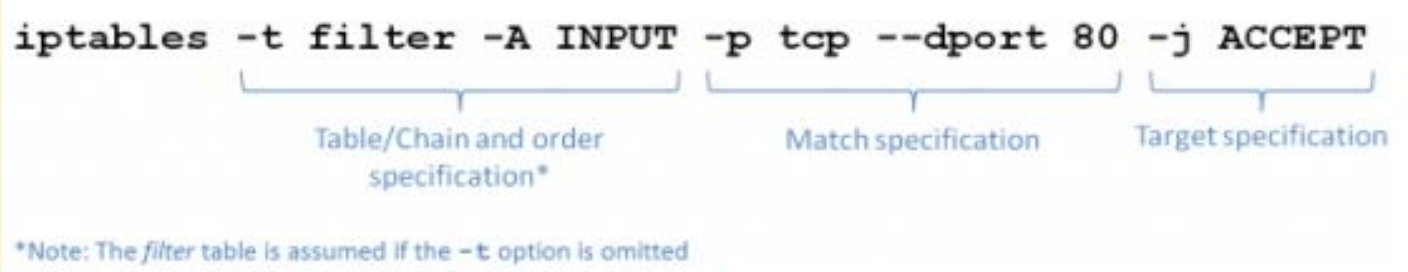
4. Processament



Font: http://cs312.osuosl.org/slides/23_security.html#5

4.1 Comandes

Definint regles



Les comandes especifiquen que farem, disposem de totes aquestes:
-L, --list [CADENA]→llista les regles de la cadena especificada . Si no s'especifica cadena, mostra totes les cadenes de la taula especificada (per defecte **filter**) Ex. iptables -L --line-numbers (també iptables -S les llista sense agrupar)

-P, --policy CADENA → es defineix una política per defecte (Restrictiva, DROP o Permissiva, ACCEPT) per a la cadena (INPUT ...).

-A, --append CADENA → afegeix una regla al final de la cadena.
Ex: iptables -A FORWARD -j ACCEPT (permet tot el tràfic)

-D, --delete CADENA esborra una regla de la cadena. Es pot determinar la cadena pel nº de regla o introduint la regla completa a comparar.
Ex: iptables -D FORWARD -j ACCEPT (esborra la regla anterior)

https://netvulcano.wordpress.com/2008/12/02/netfilter-iptables-ii/

4.1 Comandes

-R, --replace → substitueix una regla existent al nº de la CADENA especificada. Funciona com -D però substituint enlloc d'eliminar. Normalment es fa per experimentar i mirar el comportament de la regla.

iptables -R INPUT 1 -s 192.168.0.1 -j DROP

-I, --insert → Inserir una nova regla en alguna posició de la cadena. A l'exemple s'inserirà a la posició nº 1 en la cadena INPUT, pel que a partir de llavors serà la primera regla en aquesta cadena. Sense número, insereix la regla per sobre de la resta de regles de la cadena.

iptables -I INPUT 1 --dport 80 -j ACCEPT

-F, --flush →Aquesta comanda elimina totes les regles de la cadena especificada. És equivalent a esborrar cada regla una a una, però bastant més ràpid. Es pot emprar sense opcions, de manera que esborrarà totes les regles de totes les cadenes a la taula especificada. iptables -F INPUT

https://netvulcano.wordpress.com/2008/12/02/netfilter-iptables-ii/

4.1 Comandes

-Z, --zero → exactament com flush però pels comptadors d'una cadena. Més ben dit els reinicialitza a zero. iptables -Z INPUT
-N, --new-chain → fa que el nucli crei una cadena nova (no ha d'existir) iptables -N permesos (crea la cadena permesos, no existeix per defecte)
-X, --delete-chain → elimina la cadena especificada. Si es fa sense opcions les cadenes creades per l'usuari s'esborraran iptables -X permesos

-E, --rename-chain → canvia el nom de la cadena especificada no el seu comportament. iptables -E permesos NoPermesos

4.2 Comparacions

La comparació és la part de la regla que **determina a quins paquets afecta**. Pot ser per adreça **IP , port** , etc ...
Si la comparació amb el paquet es compleix, es realitza l'acció objectiu o un salt a una regla definida per l'usuari.

Es poden agrupar de moltes maneres, nosaltres el farem amb 5 grups:
1.- **Comparacions genèriques** → es podem fer servir a totes les regles.
2.- **Comparacions TCP**→ només es poden aplicar als paquets TCP.
3.- **Comparacions UDP**→ només es poden aplicar als paquets UDP.
4.- **Comparacions ICMP**→ només es poden aplicar als paquets ICMP.
5.- **Comparacions especials o explícites**→com les d'estat, les de propietari, comparacions límit, etc ...
També es podrien classificar en **implícites (genèriques, TCP, UDP, ICMP) i explicites**, que es diferencien en si **s'han d'activar de forma específica o venen activades per defecte.**

4.2 Comparacions

Comparacions genèriques. → sempre estan disponibles.

-p, --protocol → protocol del paquet a comprovar pot ser TCP, UDP i ICMP o ALL (si no especifiquem el paràmetre -p agafa tots). Accepta diversos protocols delimitats per comes (exemple tpc, udp) Es pot invertir la comparació fent servir ! (Ex --protocol **! tcp** → compara tots els protocols excepte tcp) #iptables -A INPUT -p tcp

-s, --src, --source → es basa en la IP d'origen dels paquets. També accepta màscares de xarxa (Ex 192.168.0.0/24 o 192.168.0.0/255.255.255.0). També es pot fer servir <<!>> per invertir la comparació. #iptables -A INPUT -s 192.168.1.1

-d, --dst, --destination → es fa servir en funció de l'IP de destí dels paquets. Funciona com --source. #iptables -A INPUT -d 192.168.1.1

4.2 Comparacions

Comparacions genèriques.

-i, --in-interface → La farem servir per a reconèixer a través de quina interfície prové un paquet entrant. Aquesta opció només és vàlida a les cadenes **INPUT, FORWARD i PREROUTING**, retornant un error si es fa servir en qualsevol altre lloc. També es pot fer servir **<<!>>** per invertir la comparació. Si no especifiquem una interfície concreta compararà tots els paquets sense importar de quina interfície vinguin. #iptables -A INPUT -i eth0

-o, --out-interface → És l'oposada a –in-interface. Admet inversió **<<!>>**. Funciona sobre la interfície de sortida i en aquest cas sobre les cadenes **OUTPUT, FORWARD i POSTROUTING**. Si no s'especifica cap comparació, es comparen totes les targes, independentment d'on hi vagi el paquet. #iptables -A FORWARD -o eth0

-f, --fragment → comprova la 2na i 3ra part d'un paquet fragmentat per saber destí/origen ... a més els paquets fragmentats es poden fer servir per fer atacs. Admet inversió però en aquest cas va davant de la comparació (**Ex. ! -f**) #iptables -A INPUT -f

4.2 Comparacions

Comparacions TCP → Per fer-les servir hem d'indicar --protocol tcp a la línia de comandes abans de fer-les servir.

--sport, --source-port → comprova paquets basat en el port d'origen. Es poden especificar tant un port (**80**), un rang de ports (<<**20:23**>>, es pot fer des de 0 al port que volem <<**:25**>> o des de el que volem fins al 65535 <<**80:**>>) o amb el nom del servei (ftp ha d'existir a /etc/services és una mica més lent que per nº). Si no s'indica res es comparen tots els ports d'origen iptables -A INPUT -p tcp --sport 22

--dport, --destination-port → fa servir la mateixa sintaxis que --sport però fa la comprovació al paquet basant-se als ports de destinació. iptables -A INPUT -p tcp --dport 22

--tcp-flags → comprova que la llista d'etiquetes a comprovar tenen només actius els inclosos en la llista de valors (flags activats a 1) pot reconèixer les següents etiquetes SYN, ACK, FIN, RST, URG i PSH, a més d'interpretar ALL i NONE. --tcp-flags ALL NONE significa que s'han de comprovar totes les banderes TCP i comprovar que cap d'elles està activada. Admet inversió **<<!>>** i les comes no han d'incloure espais (ex **! SYN,! FIN,! ACK SYN**) iptables -p tcp --tcp-flags SYN,FIN,ACK SYN

4.2 Comparacions

Comparacions UDP → Per fer-les servir hem d'indicar --protocol udp a la línia de comandes abans de fer-les servir. **Els paquets UDP no estan orientats a la connexió i per això no hi ha banderes** per establir el seu valor en el paquet de manera **que indiquin quina és la utilitat del datagrama** (com obrir o tancar una connexió) **o si únicament serveixen per enviar dades**. Així mateix, **tampoc requereixen cap tipus de reconeixement:** si es perden, simplement s'han perdut amb la qual cosa tenim menys opcions que per TCP.

--sport, --source-port, --dport, --destination-port→ Funcionen exactament igual que per TCP. Recordem que no s'accepten ports múltiples ni rangs múltiples separats per comes. Per a més informació sobre aquest tema llegeix-te l'extensió de comparacions multiport.

iptables -A INPUT -p udp --sport 53
iptables -A INPUT -p udp --dport 53

4.2 Comparacions

Comparacions ICMP → Per fer-les servir hem d'indicar --protocol icmp a la línia de comandes abans de fer-les servir. Els paquets **ICMP no** estan **orientats a la connexió**. El protocol **ICMP s'usa principalment per a comunicació d'errors, control de connexions i tasques similars**. ICMP **ajuda a gestionar els errors del protocol IP**. Les capçaleres dels paquets ICMP són molt similars a les capçaleres IP amb diverses diferències. La principal característica d'aquest protocol és la **capçalera de "tipus", la qual ens indica per a què és el paquet**. Per exemple , si intentem accedir a una adreça IP inaccessible, normalment rebrem un ICMP *host unreachable* com a resposta. Llista completa dels "tipus" ICMP a l'apèndix Tipus ICMP. o executant #iptables --protocol icmp --help Només hi ha una comparació específica per paquets ICMP. Recorda que totes les comparacions genèriques es poden usar amb ella, pel que ens podem centrar en les adreces d'origen i destinació.

--icmp-type →Es poden especificar per nº o nom. Aquesta comparació pot ser invertida amb **<<!>>** (--icmp-type ! 8) iptables -A INPUT -p icmp --icmp-type 8

4.3 Objectius

- -j ACCEPT. Acepta el tráfico.
- -j DROP. Elimina el tráfico.
- -j REJECT. Rechaza el tráfico e informa al equipo de origen.
- -j LOG –log-prefix "IPTABLES_L". Registra el tráfico que cumple los criterios en /var/log.

Objectius (targets)

Quan la comparació d'una regla troba un paquet que coincideix amb les condicions que imposa (tal com hem vist fins ara), es recorre a l'**objectiu/salt** on se li **indica a la regla què ha de fer amb aquest paquet**
Els objectius van precedits de l'opció -j, excepte les polítiques per defecte.

Objectius per la taula FILTER.

ACCEPT → la regla accepta el transit i no passa per cap més regla.
iptables -A FORWARD -j ACCEPT (permet tot el tansit)

DROP → la regla descarta el paquet i no passa per cap regla més.
iptables -A FORWARD -s 192.168.1.0/24 -j DROP (denegem accés a aquesta subxarxa)

REJECT → funciona bàsicament com DROP i disposa de moltes variables mitjançant **--reject-with** per enviar la resposta associada.
icmp-net-unreachable, icmp-host-unreachable, icmp-port-unreachable, icmp-proto-unreachable, icmp-net-prohibited i icmp-host-prohibited. Retornen el missatge d'error por defecte **port-unreachable**. TCP Reset es fa servir per tancar de manera elegant connexions TCP obertes
iptables -A FORWARD -p TCP --dport 22 -j REJECT --reject-with tcp-reset

4.3 Objectius

- -j ACCEPT. Acepta el tráfico.
- -j DROP. Elimina el tráfico.
- -j REJECT. Rechaza el tráfico e informa al equipo de origen.
- -j LOG –log-prefix “IPTABLES_L”. Registra el tráfico que cumple los criterios en /var/log.

Objectius per la taula FILTER.

LOG → Dissenyat per enregistrar informació detallada dels paquets per buscar defectes o errades. També es fa servir abans de fer servir per exemple **DROP** per testejar la regla.

Les opcions que poden acompanyar a aquesta regla són:

--log-level → Determina quin nivell de registre utilitzar (debug, info, notice, warning, err, crit, emerg i alert). iptables -A FORWARD -p tcp -j LOG --log-level debug

--log-prefix → Afegeix un prefix als missatges. Molt útil per seguir problemes iptables -A INPUT -p tcp -j LOG --log-prefix "INPUT packets"

--log-tcp-sequence → Afegeix al missatge el nº de seqüència tcp.
iptables -A INPUT -p tcp -j LOG --log-tcp-sequence

--log-tcp-options → Registra les opcions de la capçalera tcp.
iptables -A FORWARD -p tcp -j LOG --log-tcp-options

--log-ip-options → Registra les opcions de la capçalera IP útil per perseguir delincuents.. iptables -A FORWARD -p tcp -j LOG --log-ip-options

4.3 Objectius

Objectius per la taula NAT

DNAT: Modifica la destinació del paquet (IP i opcionalment el port). Només s'utilitza a les cadenes OUTPUT i POSTROUTE i dins de la taula de nat. La decisió presa per al primer paquet d'una connexió es recorda per a la resta de paquets de la mateixa connexió.

SNAT: Modifica l'origen del paquet (IP i opcionalment el port). Només s'utilitza a les cadenes PREROUTE i dins de la taula nat. La decisió presa per al primer paquet d'una connexió es recorda per a la resta de paquets de la mateixa connexió.

MASQUERADE: És una forma especial i restringida de SNAT per adreces IP dinàmiques.

4.4 Registre

Hi ha multiples maneres d'enregistrar els esdeveniments de IPTables. La més senzilla és afegint una regla que afegeixi una entrada al log del sistema, ubucat a /var/log/kern.log, cada cop que succeeixiun cert esdeveniment:

```
$ iptables -A INPUT -j LOG --log-prefix "Detectat paquet d'entrada:  "
```

Si volem guardar la configuració del nostre firewall:

```
iptables-save > $HOME/my.active.firewall.rules
```

I per restaurar-les: `iptables-restore < $HOME/my.active.firewall.rules`

Mes informació a: man iptables

5. Iptables Exemples

Aquesta configuració normalment s'aplica a l'inici de l'script:

#!/bin/sh o /bin/bash, que és la que fem servir nosaltres
SCRIPT de IPTABLES - exemple del manual d'iptables
Exemple de script per protegir la pròpia màquina
iptables -F → Aquesta comanda iptables esborra totes les regles una a una de la taula FILTER.
iptables -X→ Esborra les regles definides per l'usuari de la taula FILTER. Aquesta regla s'ha d'aplicar després de la F per al seu correcte funcionament.
iptables -Z→ Neteja tots els comptadors de les cadenes, els posa a 0, a la taula FILTER.
iptables -t nat -F→ esborra les regles de la taula NAT.

Un cop fet això hem d'aplicar les polítiques per defecte (ja sigui permissiva o restrictiva)

5. Iptables Exemples

Cas Permissiu

Establim política per defecte → (fixat que no porta l'opció -j, això fa que sigui la política per defecte)
iptables -P INPUT ACCEPT → Carrega la política de deixar passar tot el que entra per la interfície de xarxa, a la taula FILTER
iptables -P OUTPUT ACCEPT→ Carrega la política de deixar sortir tot el que s'envia per la interfície de Xarxa, a la taula FILTER
iptables -P FORWARD ACCEPT → Carrega la política de acceptar que es pugui passar transit d'una interfície de xarxa a una altra, a la taula FILTER.
iptables -t nat -P PREROUTING ACCEPT→ Carrega la política a la taula NAT de acceptar que es puguin alterar els paquets quan arriben a la interfície de Xarxa.
iptables -t nat -P POSTROUTING ACCEPT → Carrega la política a la taula NAT de acceptar que es puguin alterar els paquets quan s'envien per la interfície de Xarxa.

5. Iptables Exemples

##Després de la política per defecte Comencem a filtrar. Farem el cas **Permissiu** → Teòricament al ser una política d'acceptar per defecte, només farien falta les regles de **Drop**.

Per evitar errors en el sistema, hem d'acceptar totes les comunicacions per lo (localhost):

/sbin/iptables -A INPUT -i lo -j ACCEPT → Append, Afegeix regla (-A) si la interficie d'entrada (-i) és el nostre localhost (lo) accepta el tràfic (-j ACCEPT). A la nostra IP li deixem fer tot.
iptables -A INPUT -s 195.65.34.234 -j ACCEPT → si la IP font és xxx.xxx.xxx.xxx accepta el tràfic, li deixem fer tot.

A un company li deixem entrar al mysql per a que mantingui la BBDD →
iptables -A INPUT -s 231.45.134.23 -p tcp --dport 3306 -j ACCEPT → Com que més tard deneguem aquest port hem d'habilitar a quina adreça font deixem accedir (-s) el protocol que accedirà (-p) i el port destí al que li permetem accedir (--dport)

5. Iptables Exemples

A un dissenyador li deixem emprar el FTP
iptables -A INPUT -s 80.37.45.194 -p tcp --dport 20:21 -j ACCEPT → Com que més tard deneguem aquest port hem d'habilitar a quina adreça font deixem accedir (-s) el protocol que accedirà (-p) i el port destí al que li permetem accedir (--dport) en aquest cas des del port 20 al 21.
El port 80 de www ha d'estar obert, és un servidor web.
iptables -A INPUT -p tcp --dport 80 -j ACCEPT → teòricament aquest no faria falta perquè la política es d'acceptar, de totes maneres el que fa és que a tothom el deixa accedir al port 80 i no el deneguem.

I la resta, ho tanquem.
iptables -A INPUT -p tcp --dport 20:21 -j DROP (ftp)
iptables -A INPUT -p tcp --dport 3306 -j DROP (mysql)
iptables -A INPUT -p tcp --dport 22 -j DROP (ssh)

Tanquem un port de gestió: webmin
iptables -A INPUT -p tcp --dport 10000 -j DROP

Fi del script

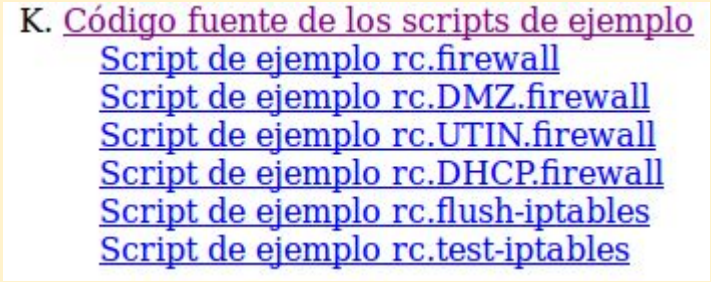
5. Iptables Exemples

Com és un script no ens hem d'oblidar de donar-li permisos d'execució.
chmod +x firewall1.sh o **chmod 750 firewall1.sh**

Cas Restrictiu:

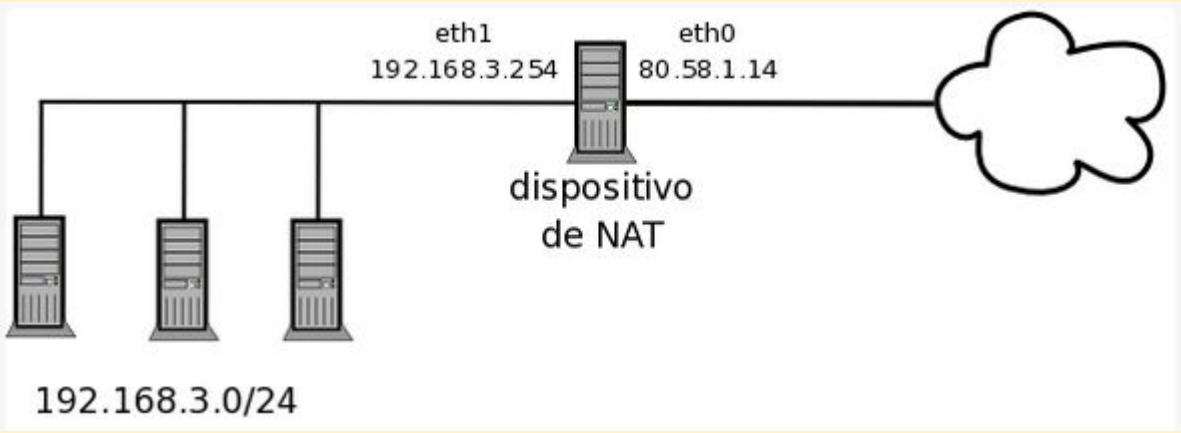
Establim política per defecte → (fixat que no porta l'opció -j, això fa que sigui la política per defecte)
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

A partir d'aquí començaríem a filtrar.


K. Código fuente de los scripts de ejemplo
Script de ejemplo rc.firewall
Script de ejemplo rc.DMZ.firewall
Script de ejemplo rc.UTIN.firewall
Script de ejemplo rc.DHCP.firewall
Script de ejemplo rc.flush-iptables
Script de ejemplo rc.test-iptables

Aquí pots mirar exemples de diferents configuracions:
https://www.frozentux.net/iptables-tutorial/spanish/iptables-tutorial.html#EXAMPLECODE
http://sw-libre.blogspot.com.es/2011/10/hola-basandome-en-la-documentacion-de.html
http://elbauldelprogramador.com/20-ejemplos-de-iptables-para-sysadmins/#more-314
Exemple interessant per restringir ip's durant una estona i evitar atacs
http://backup-cosas.blogspot.com.es/2013/01/iptablesnetfilter-recent-module.html

5. (NAT)



Source NAT (estàtic)

iptables -t nat -A POSTROUTING -s 192.168.3.0/24 -o eth0 -j SNAT --to 80.58.1.14

Source NAT (dinàmic)

iptables -t nat -A POSTROUTING -s 192.168.3.0/24 -o eth0 -j MASQUERADE

L'equip amb dues NIC haurà de fer NAT (source NAT), acceptant paquets provinents de la xarxa 192.168.3.0/24 que entrin per eth1 amb destinació a qualsevol equip d'Internet. El dispositiu de NAT ha de canviar l'adreça IP origen, però això es fa just abans d'enviar el paquet a Internet i per tant caldrà definir-lo en la cadena POSTROUTING (mirar exemple de funcionament SNAT de la web).

http://blog.desdelinux.net/nat-iptables-asegurando-red-parte2/
https://albertomolina.wordpress.com/2009/01/09/nat-con-iptables/

6. Guardar configuració

Cada cop que reinicies l'ordinador, les regles o canvis que hagis fet en iptables es perden. Per evitar això, hi ha **diverses solucions**:

Opció A:

1.- Sabent quines **regles** usar, les posem **en un arxiu** (/etc/iptables-script per exemple).

2.-Li donem **permisos d'execució** (chmod + x / etc / iptables-script)

3.- Diem al sistema que executi aquest script quan s'iniciï, per això el **posem la ruta de l'arxiu a l'arxiu /etc/rc.local**

Opció B:

1.- **Escric les regles** que vull fer servir **a la shell.**

2.- **Un cop provades** i que tot funciona com vull **executo la comanda** #iptables-save > /ruta/Fitxer

3.- **Per restaurar la configuració** → #iptables-restore < /ruta/Fitxer

4.- **Per què aquestes regles s'inicien al reiniciar, hem de crear un altre arxiu a:** #geany /etc/network/if-pre-up.d/iptables

I Afegir les següents línies:

```
#!/bin/bash
 /sbin/iptables-restore < /etc/iptables.up.rules
```

I fer l'arxiu executable #chmod + x /etc/network/if-pre-up.d/iptables

https://wiki.debian.org/es/iptables

http://blog.desdelinux.net/como-iniciar-reglas-de-iptables-automaticamente/

6. Guardar configuració

Per Red Hat funciona una mica diferent, el podem trobar a: http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/s1-iptables-saving.html
Executem com a root**#/sbin/service iptables save**

Això executa l'script d'inici iptables, el qual executa el programa /sbin/iptables-save i escriu la configuració actual de iptables a /etc/sysconfig/iptables. L'arxiu /etc/sysconfig/iptables existent és guardat com /etc/sysconfig/iptables.save.

La propera vegada que s'iniciï el sistema, l'script d'inici d'iptables tornarà a aplicar les regles guardades en /etc/sysconfig/iptables fent servir la comanda /sbin/iptables-restore.

http://web.mit.edu/rhel-doc/4/RH-DOCS/rhel-rg-es-4/index.html