HowTo ASIX Firewalls

iptables firewalld

Curs 2017 - 2018

Documentació

Aquest document ha estat elaborant utilitzant com a eina de treball un sistema GNU/Linux Fedora 20.

- Documentació de les pàgines man de les ordres.
- Fedora Documentation, Fedora 14. Security Guide: 2.8 firewalls, 2.9 iptables.
- Manual de iptables de Pello: http://www.pello.info/filez/firewall/iptables.html

Ordres:

iptables iptables-restore iptables-save iptables-xml

Xuleta de treball Curs 2017-2018

---

1) Regles INPUT / OUTPUT

Practicar al propi host de l'alumne.
El host utilitza els serveis: sshd (22) , httpd (80), daytime-stream (13), echo-stream (7).
Es poden ampliar els serveis tripicant-los amb xinetd aquests serveis usant els ports 2007, 2013, 2080 i 3007, 3013 i 3080.
Usar dos hosts de l'aula (i28 i i29) des d'on tots els alumnes poden atacar remotament els serveis del seu host.

Per practicar les regles INPUT, OUTPUT i Port Forwarding (llocal) usar el propi host de l'aula i establir regles locals.

Exemples de regles:

- Tallar toc el tràfic entrant.

- Tallar trafic entrant a un port.
- Tallar tràfic entrant si prové de una xarxa concreta.
- Tallar tràfic entrant si prové de un host concret.
- Combinacions de port, xarxa i host.

- Exemples de frases a saber implementar:
  - permetre tots.
  - tallar a tots.
  - tallar a una xarxa.
  - tots no però una xarxa si.
  - una xarxa si/no però un host no/si.
  - tots si/no, una xarxa no/si, un host concret si/no

- Idem amb tràfic sortint atacantserveis de els hosts remots i28 i i29 (22, 80, 7 i 13).

2) Regles Port Forwarding (local) PREROUTING.

Els primers exemples senzills de port forwarding és atacar ports 4001, 4002, 4003… i redirigir-ho a altres ports locals o remots (--to a.b.c.d:port).

Observar que amb els ports locals funciona (el --to :port), però a remots no funciona si el host no fa forwarding.

Fer forwarding amb el echo 1 i FORWARD ACCEPT per defecte. Observar que si es treu el echo1 o el forward no accedeix als remots

IMPORTANT:

Sempre després del prerouting ve el routing o bé de input o bé de forward. Verificar que si el INPUT fa DROP no fa el port forwarding local.

3) NAT: POSTROUTING

Implementar amb containers Docker l'estructura següent:
xarxaA 172.31.0.0./16: host a1(.2) ----- host a2(.3) ---- (.1) br.1 Host-alumne
Host-> eth0--publica
xarxaB 172.32.0.0./16: host b1(.2) ----- host b2(.3) ---- (.1) br-2 Host-alumne

Crear amb docker network dues xarxes i dos hosts per xarxa.
Al host alumne desar la configuració iptables actual amb iptables-save.

Practicar:

- ❏ esborrar les regles iptables i observar que ara els containers no fan nat però si que es poden fer ping els uns als altres.
- ❏ Posar regles NAT per a les dues xarxes i observar que ara tenen connectivitat a l'exterior.

4) FORWARDING

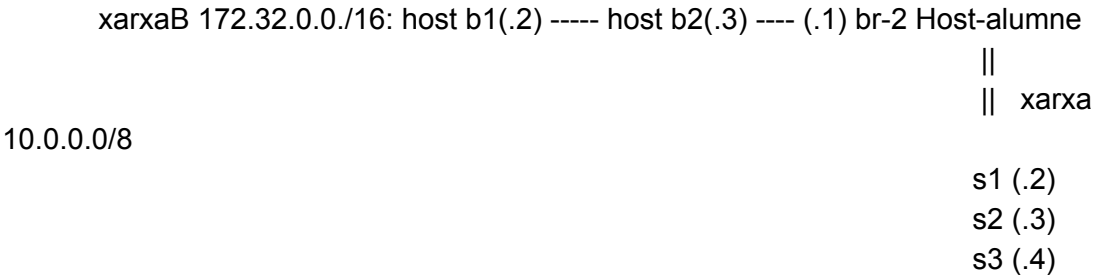
Usar l'esquema anterior de NAT i Containers Docker per fer forwarding de tipus:

a) Tràfic xarxes locals / exterior (el que passa pel NAT).
b) Tràfic de xarxaA a xarxaB.
c) Tràfic que es genera amb el Port Forwarding, quan les peticions exteriors al host són reenviades als containers interiors.

IMPORTANT: Si hi ha port forwarding un cop fet el PREROUTING s'aplica el FORWARD (o INPUT).

5) DMZ: Demilitarized Zone

xarxaA 172.31.0.0./16: host a1(.2) ----- host a2(.3) ---- (.1) br.1 Host-alumne
Host-> eth0--publica



Ampliar la topologia de containers docker amb una nova xarxa que representa la DMZ amb servidors de s1(web), s2(smtp/imap/pop) i s3(daytime/echo) anomenats s1, s2 i s3. És una xarxa clarament diferenciada.

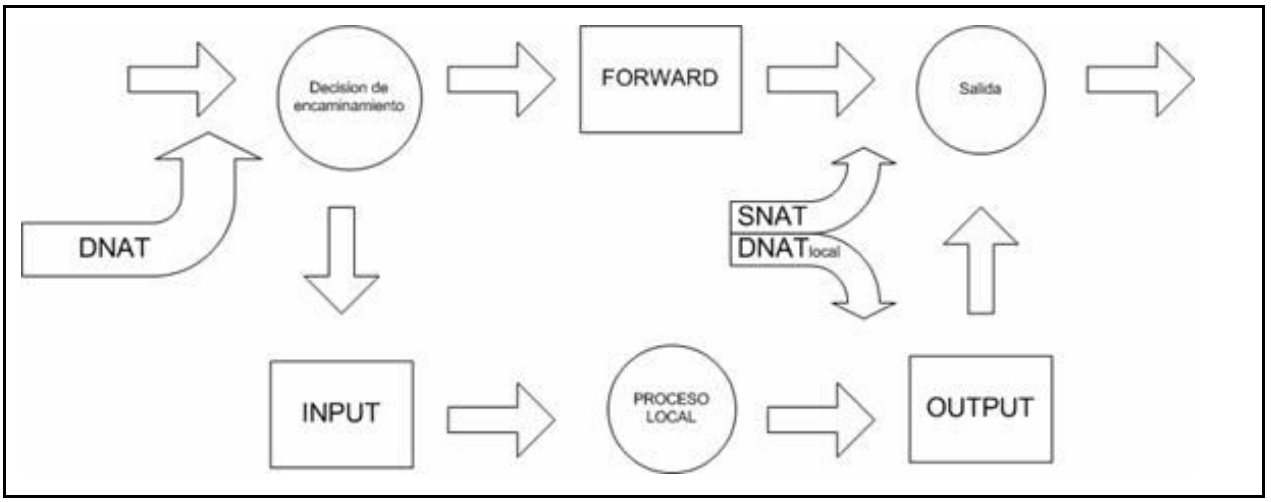
Exemples de regles:

- de l'exterior es té acces als servidors de la dMZ però no a les xarxes interiors.
- les xarxes interiors poden accedir als servidors.
- de la xarxaA hi ha accés total a la xarxaB
- de la xarxaB no es pot accedir a la xarxa A.
- hi ha un port 4001 que permet accedir per ssh a a1. 4002 per ssh a b1.
- el port 4002 permet acces per ssh al servidor s1.

Iptables: descripció general

Documentació basada en el manual de Pello: www.pello.info

Regles de decisió del kernel / iptables d'un paquet rebut:
gràfic extret de la documentació de pello www.pello.info



Regles iptables:
- MANGLE
- NAT: reglas PREROUTING, POSTROUTING
- FILTER: reglas INPUT, OUTPUT, FORWARD

Observar els ports que hi ha oberts en un sistema:

```
[root@portatil ~]# netstat -tupa | grep LISTEN
tcp        0      0 localhost:37803         *:*                     LISTEN      1778/GoogleTalkPlug
tcp        0      0 *:sunrpc                *:*                     LISTEN      851/rpcbind
tcp        0      0 portatil.localdo:domain *:*                     LISTEN      937/dnsmasq
tcp        0      0 *:51286                 *:*                     LISTEN      858/rpc.statd
tcp        0      0 *:ssh                   *:*                     LISTEN      847/sshd
tcp        0      0 localhost:ipp           *:*                     LISTEN      566/cupsd
```

Exemple 01: política per defecte

Exemple de configuració iptables que esborra les regles actuals, estableix una política per defecte de tot obert, permet tot el tràfic IN/OUT del loopback i de la pròpia adreça IP. De fet és una mica absurd perquè la política per defecte ACCEPT ja el permet.

```
#!/bin/bash
# ASIX M11-Seguretat i alta disponibilitat
# @edt 2015
```

```
# ============================================

# Activar si el host ha de fer de router
#echo 1 > /proc/sys/net/ipv4/ip_forward

# Regles Flush: buidar les regles actuals
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# Establir la política per defecte (ACCEPT o DROP)
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

# Filtrar ports (personalitzar)  - - - - - - - - - - - - - - - - - - - - - - -
# ------------------------------------------------------------------------

# Permetre totes les pròpies connexions via localhost
iptables -A INPUT  -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# Permetre tot el trafic de la pròpia ip(192.168.1.34))
iptables -A INPUT  -s 192.168.1.34 -j ACCEPT
iptables -A OUTPUT -d 192.168.1.34 -j ACCEPT

# Aplicar altres regles per obrir i tancar ports
# ....
# Aplicar altres regles per permetre o no tipus de trafic
# ....
# ------------------------------------------------------------------------

# Mostrar les regles generades
iptables -L -t nat
```

```
[root@host ~]# ./iptables-01-regles_basiques.sh
Chain PREROUTING (policy ACCEPT)
target   prot opt source          destination

Chain OUTPUT (policy ACCEPT)
target   prot opt source          destination

Chain POSTROUTING (policy ACCEPT)
target   prot opt source          destination
```

Exemple 02: establir algunes regles bàsiques

Els següent és un exemple que permet observar com permetre tràfic entrant als ports del postgres i ftp a hosts concrets i al servei web a tothom. Es denega l'accés a aquests serveis (FTP, postgres, ssh, etc) a qualsevol altre host entrant.

```
# Aplicar altres regles per obrir i tancar ports
# Permetre al 231.45.134.23 accedir al postgres (port 5432)
iptables -A INPUT -s 231.45.134.23 -p tcp --dport 5432 -j ACCEPT

# Permetre accedir al FTP (ports 20,21) al host 80.37.45.194
iptables -A INPUT -s 80.37.45.194 -p tcp -dport 20:21 -j ACCEPT

# Permetre l'accés al servidor web (port 80) de la xarxa 80.37.0.0/16
iptables -A INPUT -s 80.37.0.0/16 -p tcp --dport 80 -j ACCEPT

# Tancar aquests serveis a la resta de hosts
iptables -A INPUT   -p tcp --dport 20:21 -j DROP
iptables -A INPUT   -p tcp --dport 5432  -j DROP
iptables -A INPUT   -p tcp --dport 80      -j DROP

# Tancar l'accés extern al servei ssh i daytime del host
iptables -A INPUT   -p tcp --dport 22    -j DROP
iptables -A INPUT   -p tcp --dport 13      -j DROP
```

Exemple 03: política drop per defecte

```
<under construction>
```

Regles i Accions

Les regles s'avaluen una a una seqüencialment de la primera cap avall. Si una regla fa *match* (coincideix) es deixa d'examinar les regles, s'aplica i surt.

Hi ha vàries cadenes o *chains* de regles diferents:

- ❏ INPUT
- ❏ OUTPUT
- ❏ FORWARD

- ❏ MANGLE
- ❏ NAT: PREROUTING i POSTROUTING

Les accions a fer són:

- ACCEPT
- DROP
- REJECT

- DNAT
- SNAT

Les regles "*noves*" es poden afegir a:
- Al final de les regles ja existents usant -A de *append*.
- A l'inici de les regles ja existents usant -I de *insert*.

Activar el filtrat de paquets amb iptables: és un servei?

És actualment un servei com a tal o regles que s'apliquen al Kernel? Són regles directes que s'apliquen al kernel. En canvi firewalld es un servei.

Funcionen independentment de firewalld?

```
$ systemctl start iptables
$ systemctl start ip6tables
$ systemctl status firewalld
```

Esborrar / Desar / Carregar regles

L'exemple següent descriu com eliminar les regles actuals deixant en blanc el conjunt de regles, és a dir, fer un flush.

```
# Regles Flush: buidar les regles actuals
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
```

Desar les regles que s'estan aplicant actualment al kernel a un fitxer (tipus script) que podrà ser cagat a iptables quan es consideri oportú:

```
# iptables-save
```

Carregar de nou una configuració iptables basada en una configuració anterior generada per iptables-save:

```
# iptables-restore
```

Generar una configuració pròpia, personalitzada del firewall. Escriure un fitxer script (o varis per a cada situació) que estableixin les regles iptables apropiades a cada cas.
mv i

```
# iptables-regles01.sh
```

Polítiques per defecte

Es poden establir dues polítiques per defecte ACCEPT i DROP. Generalment en l'inici dels scripts s'estableix quina és la política per defecte que seguirà aquell script, tot permès per defecte o tot prohibit per defecte.

ACCEPT: Tot està permès per defecte excepte allò que explícitament es denegui. En aquest cas les regles del firewall són el llistat de tot allò que explícitament es prohibeix i la resta està tot permès.

```
# Establir la política per defecte ACCEPT
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
```

DROP: Tot està prohibit excepte allò que explícitament es permet. En aquest cas l'script del firewall és el conjunt de regles que permeten tràfic, el que no hi sigui contemplat està prohibit per defecte.

Generar firewalls amb política per defecte drop és molt més difícil perquè cal assegurar-se de que no es tanca res que és imprescindible que estigui obert. Però també és més segur, perquè tot el que no és imprescindible està tancat.

```
# Establir la política per defecte DROP
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP
iptables -t nat -P PREROUTING DROP
iptables -t nat -P POSTROUTING DROP
```

Obrir tot el tràfic per a la pròpia adreça ip del host

Sovint es desitja permetre tot tipus de tràfic a una determinada adreça IP local o a un device local, per exemple al loopback o a la propia adreça IP (una, alguna o tetes elles) per exemple de eth0.

Per dir a la meva interfície tal del host/firewall li permeto tot el tràfic d'entrada i sortida s'escriu:

```
# Permetre totes les pròpies connexions via localhost
iptables -A INPUT       -i lo -j ACCEPT
iptables -A OUTPUT  -o lo -j ACCEPT

# Permetre tot el trafic de la pròpia adreça ip (192.168.1.34)
iptables -A INPUT       -s 192.168.1.34   -j ACCEPT
iptables -A OUTPUT  -d 192.168.1.34   -j ACCEPT
```

Casos pràctics amb iptables

Política ACCEPT: regles per 'tancar' serveis

En una configuració iptables amb política per defecte ACCEPT tot està permès i el que cal és afegir regles per tancar el tràfic que no es permet.

El cas més fàcil de provar és en el propi host aplicar una política per defecte ACCEPT i considerar aquests casos:

- INPUT: tancar serveis que ofereix el propi host/firewall.
- OUTPUT: tancar serveis externs, és a dir, no permetre que el propi host/firewall es connecti a determinats serveis externs.
- FORWARD: tancar l'accés a serveis que creuen el firewall.

Cas 1: de fora no es pot accedir al servei 'servei' del host/firewall

Suposem un host/firewall que ofereix un servei de cara a l'exterior, a altres hosts, siguin d'on siguin (d'altres xarxes o de la pròpia xarxa) no poden accedir al servei. Observar els casos:

- tancar l'accés d'entrada a un servei (indicant el port). Sigui quin sigui l'origen.
- tancar l'accés a un servei si l'origen és un host concret.
- tancar l'accés a un servei si l'origen és una xarxa concreta.
- tancar un servei a tots els hosts menys un de concret.
- combinacions dels casos anteriors: al servei tal tots els de la xarxa de segon no hi poden entrar però si l'alumne tal. O a l'inrevés, a tal servei està denegat per tothom, als de segon si que poden accedir-hi però no l'alumne tal.

Exemples de tancar tràfic entrant al host/firewall, **INPUT**:

```
# Es denega l'accés al servei local del host/firewall
```

```
# de web, telnet i ftp (tràfic tcp) i tftp (tràfic udp)

iptables -A INPUT  -p tcp   --dport 80       -j DROP
iptables -A INPUT  -p tcp   --dport 23       -j DROP
iptables -A INPUT  -p tcp   --dport 20:21  -j DROP
iptables -A INPUT  -p udp  --dport 69       -j DROP
```

```
# Es denega l'accés al servei "servei" local del host/firewall
# si l'origen és un host concret
# host 231.45.134.23 no pot accedir a smtp ni tftp
# host 192.168.1.55   no pot accedir a ftp ni web

iptables -A INPUT -s 231.45.134.23   -p tcp    --dport 25       -j DROP
iptables -A INPUT -s 231.45.134.23   -p udp   --dport 69      -j DROP
iptables -A INPUT -s 192.168.1.55     -p tcp   --dport 20:21  -j DROP
iptables -A INPUT -s 192.168.1.55     -p tcp   --dport 80       -j DROP
```

```
# Es denega l'accés al servei "servei" local del host/firewall
# si l'origen és una xarxa concreta
# xarxa 192.168.1.0/24 no accés al servei postgresql
# xarxa 80.37.0.0/16 no accés al serve web
# qualsevol xarxa/host no accés al servei telnet

iptables -A INPUT -s 192.168.1.0/24    -p tcp   --dport 5432   -j DROP
iptables -A INPUT -s 80.37.0.0/16        -p tcp   --dport 80       -j DROP
iptables -A INPUT -s 0.0.0.0/0             -p tcp   --dport 23       -j DROP
```

```
# Tancar un servei a tots els hosts excepte els indicats.
# host 192.168.1.55 accés als serveis echo, daytime. Altres host denegat
# xarxa 192.168.1.0/24 accés al servei ssh, altres hosts/xarxes denegat

iptables -A INPUT -s 192.168.1.55     -p tcp   --dport 7   -j ACCEPT
iptables -A INPUT                            -p tcp    --dport 7  -j DROP

iptables -A INPUT -s 192.168.1.55     -p tcp   --dport 13    -j ACCEPT
iptables -A INPUT                            -p tcp    --dport 13   -j DROP

iptables -A INPUT -s 192.168.1.0/24   -p tcp   --dport 22    -j ACCEPT
iptables -A INPUT                             -p tcp  --dport 22     -j DROP
```

Tot i que la política per defecte sigui accept sempre es poden posar regles que denegin un tipus de tràfic a tots els hosts, tancar un servei, o regles que el deneguin per a tots menys per a uns casos concrets.

Atenció: "**l'ordre de les regles és determinant**", per tant, per dir tot el tràfic entrant ssh al host/firewall està denegat excepte a tal i tal host o xarxa cal:

- **primer**: escriure les regles que de manera explícita ***permeten*** el tràfic a tal i tal host o xarxa.
- **segon**: a continuació escriure les regles que ***denegen*** el tràfic a la resta.

Es pot observar de l'últim exemple l'aplicació d'aquesta manera de fer les regles iptables.

Cas 2: tancar ports d'entrada que no siguin necessaris

En un firewall amb política per defecte accept abans de finalitzar podem tancar tots aquells ports que no siguin necessaris i que ens volem assegurar que estan tancats. Atenció, però de no tancar ports que fan falta.

Primerament es poden observar quins són els ports que usualment tenim oberts i analitzar si realment fan falta o no (mirar nmap o netstat o ss). Si hi ha ports que explícitament volem tancar o simplement volem assegurar-nos de tancar rangs de ports perillosos podem fer-ho.

Això no és cap contradicció amb la política per defecte accept, podem fer-ho de les dues maneres (amb accept i amb drop). La idea és:

- usem política per defecte accept, tot el tràfic és permès menys el prohibit.
- establim regles indicant el tràfic prohibit.
- podem establir regles denegant tràfic excepte per a casos concrets.
- finalment tanquem tots aquells ports que volem assegurar-nos que estan tancats.

Recordar: al final de les regles per defecte accept és com si hi ha un "**accept all**", permetre tot el que queda!.

```
# Tancar els ports privilegiats 0-1023
# atenció: és molt perillós si no ens hem assegurat d'obrir serveis necessaris
iptables -A INPUT -p tcp --dport 1:1024 -j DROP
iptables -A INPUT -p udp --dport 1:1024 -j DROP

# Tancar explícitament ports o rangs de ports
iptables -A INPUT -p tcp   --dport 10000       -j DROP
iptables -A INPUT -p udp   --dport 10000       -j DROP
iptables -A INPUT -p tcp   --dport 2000:3000 -j DROP
```

Cas 3: denegar accés a un servei extern

En els exemples anteriors s'han vist regles que denegen l'accés a un servei que ofereix el host/firewall, és a dir regles de INPUT al iptables, regles sobre el tràfic que va dirigit al host/firewall.

Anem a veure regles per denegar l'accés del host/firewall a serveis externs. És a dir, regles sobre el tràfic que es genera en el propi host/firewall dirigit cap a fora, regles OUTPUT del iptables.

Observar els casos de:

- tancar l'accés a un servei extern concret (indicant el port). Sigui on sigui aquest servidor extern.
- tancar l'accés a un servei d'un servidor concret. És a dir, a tal servidor no t'hi pots connectar per aquest servei.
- tancar l'accés a un servei si el servei es troba en una xarxa destí concreta (vetada).
- tancar l'accés un servei excepte si va dirigit a un host en concret. Si va dirigit a qualsevol altre servidor denegar-lo.
- tancar rangs de ports de sortida.
- tancar l'accés en funció del destí (sigui quin sigui el servei a accedit). És a dir, no es permet sortir cap a tal host o no es permet accedir cap a tal xarxa destí.

Exemples de tancar tràfic sortint generat en el propi host/firewall, regles **OUTPUT**:

```
# Es denega l'accés al servei extern de
# web, telnet i ftp (tràfic tcp) i tftp (tràfic udp)

iptables -A OUTPUT  -p tcp   --dport 80       -j DROP
iptables -A OUTPUT  -p tcp   --dport 23       -j DROP
iptables -A OUTPUT  -p tcp   --dport 20:21  -j DROP
iptables -A OUTPUT  -p udp  --dport 69       -j DROP
```

```
# Es denega l'accés al servei "servei" des del host/firewall
# si el destí és un host concret
# no pot accedir a smtp ni tftp del host  231.45.134.23
# no pot accedir a ftp ni web del host 192.168.1.55

iptables -A OUTPUT -d 231.45.134.23   -p tcp   --dport 25       -j DROP
iptables -A OUTPUT -d 231.45.134.23   -p udp   --dport 69      -j DROP
iptables -A OUTPUT -d 192.168.1.55     -p tcp   --dport 20:21  -j DROP
iptables -A OUTPUT -d 192.168.1.55     -p tcp   --dport 80       -j DROP
```

```
# Es denega l'accés al servei "servei" des del host/firewall
# si el destí (servidor del servei) és en una xarxa concreta
# no accés al servei postgresql si el servidor és en la xarxa 192.168.1.0/24
# no accés al serve web si el servidor és en la xarxa 80.37.0.0/16
# no accés al servei telnet de qualsevol xarxa

iptables -A OUTPUT -d 192.168.1.0/24    -p tcp   --dport 5432   -j DROP
iptables -A OUTPUT -d 80.37.0.0/16       -p tcp   --dport 80       -j DROP
iptables -A OUTPUT -d 0.0.0.0/0            -p tcp   --dport 23       -j DROP
```

```
# Tancar els ports privilegiats 0-1023
# atenció: és molt perillós si no ens hem assegurat d'obrir serveis necessaris abans
iptables -A OUTPUT -p tcp --dport 1:1024 -j DROP
iptables -A OUTPUT -p udp --dport 1:1024 -j DROP

# Tancar explícitament ports o rangs de ports
iptables -A OUTPUT -p tcp   --dport 10000       -j DROP
iptables -A OUTPUT -p tcp   --dport 2000:3000 -j DROP
```

```
# Tancar accés a un host destí o a una  xarxa destí
# no permetre accedir al host 192.168.1.55
# no permetre accedir a la xarxa 80.37.0.0/16
iptables -A OUTPUT -d 192.168.1.55   -j DROP
iptables -A OUTPUT -d 80.37.0.0/16   -j DROP
```

Cas 4: tancar accés INPUT a un servei però permetre les respostes entrants establertes

Si no es permet l'accés al servei web del host/firewall significa que des del host/firewall no es pot nevegar per internet?. NO. Observem que són dues coses diferents:

- el host/firewall pot tenir un servidor web http en marxa i atendre peticions d'una intranet o peticions locals d'ell mateix, però no permet que trafic extern o cap mena de tràfic d'altres hosts accedeixi al servei web. És a dir, tenir tancat per la regla **INPUT** el tràfic d'accés al port 80 o 443.

- però si que es vol que des del host/firewall es pugui navegar per internet. Això significa que des de ports dinàmics es generin connexions i tràfic a ports 80 o 443 d'altres hosts (de servidors web). És a dir, obrir l'accés **OUTPUT** a ports externs 80 o 443 i permetre també les **respostes entrants** d'aquests servidors.

En resum: es permet tràfic http cap a fora i cap a dins, però no connexions de fora al propi servidor http (port 80 i 443).

Recorda:

Si el host ofereix el servei web i el firewall en vol prohibir l'accés extern a aquest servei, estem parlant de regles **INPUT** on el destination port **-dport** es el 80/443.

Si es vol filtrar el tràfic web permetent o impedint que el host accedeixi al servei web extern (es connecti cap a fora a internet), estem parlant de regles **OUTPUT** on el destination port **-dport** es el 80/443. Però recorda que cal permetre/prohibir el tràfic de tornada on la regla serà **INPUT** i el source port **-sport** serà el 80/443.

Aquest tipus de regles s'utilitzen sovint en polítiques per defecte DROP on cal explicitar què es permet (perquè tot l'altre tràfic no indicat està prohibit!).

```
# Permetre al host/firewall accedir a internet (cap a l'exterior)
# recordar que no es permet l'accés al seu servidor web (no accepta connexions web)

/sbin/iptables -A INPUT      -p tcp  -m tcp   --sport 80   -m state  \
                             --state RELATED,ESTABLISHED   -j ACCEPT
/sbin/iptables -A OUTPUT  -p tcp   -m tcp   --dport 80             -j ACCEPT


/sbin/iptables -A INPUT       -p tcp   -m tcp   --sport 443   -m state \
                              --state RELATED,ESTABLISHED  -j ACCEPT
/sbin/iptables -A OUTPUT  -p tcp   -m tcp    --dport 443           -j ACCEPT

iptables -A INPUT  -p tcp    --dport 80  -j DROP
iptables -A INPUT  -p tcp    --dport 443  -j DROP
```

```
# Permetre al host/firewall accedir a servidors ftp
# en aquest cas el host/firewall no té servei ftp
# atenció a permetre ftp-actiu i ftp-passiu

/sbin/iptables -A INPUT      -p tcp   -m tcp   --sport 20:21   -m state \
                             --state RELATED,ESTABLISHED   -j ACCEPT
/sbin/iptables -A OUTPUT -p tcp   -m tcp   --dport 20:21          -j ACCEPT

/sbin/iptables -A INPUT       -p tcp  -m tcp   --sport 1024:65535  --dport 1024:65535 \
                              -m state --state ESTABLISHED              -j ACCEPT
/sbin/iptables -A OUTPUT  -p tcp   -m tcp   --dport 1024:65535 -m state \
                              --state NEW,RELATED,ESTABLISHED   -j ACCEPT
```

Cas 5: tancar accés OUTPUT a un servei però permetre les respostes sortints establertes

Aquest és el cas antagònic al cas anterior. Si no es permet al host/firewall navegar per internet es pot permetre al mateix temps al propi host/firewall actual com a web server permetent l'accés exterior al seu port 80 i 443?. SI. Observem que són dues coses diferents:

- denegar l'accés des del host/firewall a internet significa prohibir que surti tràfic destinat als ports 80 i 443. És a dir, una regla **OUTPUT** denegant aquest tràfic http de sortida.

- permetre que el host/firewall actui de servidor web significa permetre l'accés de tràfic entrant destinat als ports 80 o 443 del host/firewall, es tracta de permetre aquest tràfic entrant en una regla **INPUT**. Però als clients web el servidor els enviarà

respostes de manera que també cal permetre que surtin aquestes respostes http fetes per el servidor web del host/firewall. És a dir, cal permetre el tràfic de sortida http originat pels ports 80 i 443 que és **resposta sortint** a les peticions dels clients.

En resum: es denega tràfic nou http cap a fora, dirigit a servidors de fora (ports 80 i 443), però no connexions entrants destinades al propi servidor i a les respostes d'aquest servidor (tràfic establert).

Recorda:

Si al host no se li permet accedir al serveis web externs (tot i que el propi host proporciona el servei http"), estem parlant de regles **OUTPUT** on el destination port **-dport** es el 80/443.

Si es vol permetre el tràfic exterior cap al servidor web del propi host i permetre que aquest servidor emeti respostes http cap als clients, estem parlant de regles **INPUT** on el destination port **-dport** es el 80/443. Però recorda que cal permetre el tràfic de resposta on la regla serà **OUTPUT** i el source port **-sport** serà el 80/443.

Aquest tipus de regles s'utilitzen sovint en polítiques per defecte DROP on cal explicitar què es permet (perquè tot lo altre està prohibit!).

```
# Denegar al host/firewall accedir a internet
# permetre l'accés extern al servidor web del host/firewall

/sbin/iptables -A INTPUT       -p tcp   -m tcp   --dport 80  -j ACCEPT
/sbin/iptables -A OUTPUT      -p tcp   -m tcp   --sport 80   -m state  \
                                        --state RELATED,ESTABLISHED   -j ACCEPT

/sbin/iptables -A INTPUT       -p tcp   -m tcp   --dport 443    -j ACCEPT
/sbin/iptables -A OUTPUT      -p tcp   -m tcp   --sport 443   -m state \
                                        --state RELATED,ESTABLISHED  -j ACCEPT

iptables -A OUTPUT  -p tcp    --dport 80    -j DROP
iptables -A OUTPUT  -p tcp    --dport 443  -j DROP
```

Cas 6: Com indicar connexions ja establertes

En aquest apartat es mostra com indicar 'en general' que s'accepta tràfic entrant o sortint si es tràfic ja establert:

```
# exemples de regles que 'obren' al tràfic ja establert

iptables   -t filter   -A INPUT   -m state  --state ESTABLISHED,RELATED    -j ACCEPT
iptables   -t filter   -A OUPUT   -m state  --state ESTABLISHED,RELATED    -j ACCEPT
```

La següent taula descriu tipus d'estats:

States for --ctstate:

INVALID
meaning that the packet is associated with no known connection

NEW
meaning that the packet has started a new connection, or otherwise associated with a connection which has not seen packets in both directions, and

ESTABLISHED
meaning that the packet is associated with a connection which has seen packets in both directions,

RELATED
meaning that the packet is starting a new connection, but is associated with an existing connection, such as an FTP data transfer, or an ICMP error.

UNTRACKED
meaning that the packet is not tracked at all, which happens if you use the NOTRACK target in raw table.

SNAT
A virtual state, matching if the original source address differs from the reply destination.

DNAT
A virtual state, matching if the original destination differs from the reply source.

Política DROP: regles per 'obrir' serveis

Quan la política és drop per defecte es tanquen tots els ports, tot tipus de tràfic, excepte el que abans s'ha obert explícitament. Ja s'ha dit que generar aquests tipus de regles és més difícil, perquè cal assegurar-se de deixar obert tot allò que és necessari per funcionar apropiadament. En contrapartida s'incrementa la seguretat en no permetre cap altre tipus de tràfic (sempre que s'hagi fet be!).

Sovint en aplicar polítiques DROP per defecte (i més en els principiants!) el sistema es queda bloquejat perquè s'han tancat serveis necessaris per al seu funcionament.

Així doncs en un firewall d'aquest tipus cal explicitar un a un el tipus de tràfic que es permet. Cal recordar sempre aquestes regles:

- tota comunicació de xarxa té dos sentits, el d'anada i el de tornada, petició/resposta.

- cal sempre permetre els **dos sentits** de la comunicació per considerar accessible un servei.
- el tràfic de resposta es pot identificar amb modificadors del tipus related, established, etc. A l'apartat "cas 6: com indicar connexions ja establertes" podeu consultar la taules d'estats.

Cal assegurar-se de no tancar (millor dit! de deixar oberts) els ports i tràfic necessari per el sistema. Són exemples de tràfic important:

- tràfic DNS per resoldre noms de host.
- tràfic de sincronització horària.
- pings?
- altres serveix en execució que poden ser fonamentals com LDAP, Kerberos, NFS, SAMBA, etc si la màquina depèn d'ells per al bon funcionament.

Cas 1: obrir ports en una política drop per defecte

Així doncs en el cas de política DROP per defecte cal saber activar cada tipus de tràfic a permetre en tots dos sentits. Aquests són alguns exemples:

```
# Permetre consultes DNS (primari i secundari)

/sbin/iptables -A INPUT      -s 80.58.61.250  -p udp   --sport 53 -j ACCEPT
/sbin/iptables -A OUTPUT  -d 80.58.61.250  -p udp  --dport 53 -j ACCEPT

/sbin/iptables -A INPUT      -s 80.58.61.254  -p udp    --sport 53 -j ACCEPT
/sbin/iptables -A OUTPUT  -d 80.58.61.254  -p udp   --dport 53 -j ACCEPT
```

```
# Actuar de servidor web el host/firewall

/sbin/iptables -A INPUT       -p tcp -m tcp --dport 80 -j ACCEPT
/sbin/iptables -A OUTPUT  -p tcp  -m tcp --sport 80 -m state \
                              --state RELATED,ESTABLISHED -j ACCEPT
```

```
# Permetre al host/firewall navegar per internet

/sbin/iptables -A INPUT        -p tcp -m tcp --sport 80 -m state \
                               --state RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -A OUTPUT   -p tcp -m tcp --dport 80 -j ACCEPT

/sbin/iptables -A INPUT        -p tcp -m tcp --sport 443 -m state  \
                               --state RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -A OUTPUT   -p tcp -m tcp --dport 443 -j ACCEPT
```

```
# Consultar al servidor de temps per permetre la sincronització horària
```

```
/sbin/iptables -A INPUT       -p udp -m udp  --dport 123 -j ACCEPT
/sbin/iptables -A OUTPUT   -p udp -m udp  --sport 123 -j ACCEPT
```

```
# Permetre el tràfic SMTP d'entrada i sortida
iptables -A INPUT      -s 0.0.0.0/0 -p tcp --dport 25 -j ACCEPT
iptables -A OUTPUT -s 0.0.0.0/0 -p tcp --dport 25 -j ACCEPT
```

```
# Servei FTP actiu i passiu

/sbin/iptables -A INPUT       -p tcp -m tcp --sport 20:21 -m state \
                              --state RELATED,ESTABLISHED -j ACCEPT
/sbin/iptables -A OUTPUT   -p tcp -m tcp --dport 20:21 -j ACCEPT
/sbin/iptables -A INPUT       -p tcp -m tcp --sport 1024:65535 --dport 1024:65535 -m state
\
                                --state ESTABLISHED -j ACCEPT
/sbin/iptables -A OUTPUT    -p tcp -m tcp --dport 1024:65535 -m state  \
                              --state NEW,RELATED,ESTABLISHED -j ACCEPT
```

En aquests exemples es permet el tràfic al propi servidor web i ftp, el tràfic DNS i smtp entrant i sortint als servidors primari o secundari, navegar per internet des del host/firewall i el tràfic entrant i sortint de sincronització horària.

Podeu consultar un exemple més complert de política de drop en l'apartat:

Cas 4: Política DROP per defecte

Cas 2: preparar una barrera de seguretat en una política DROP per defecte

Tal i com sabem un firewall basat en política per defecte DROP conté tot de regles explícites per obrir determinat tràfic perquè tota la resta de tràfic es tanca per defecte. Passa però que a vegades el firewall provocarà que algunes coses deixin de funcionar (precisament perquè el firewall fa molt bé la seva feina de tancar-ho tot!). Per exemple perquè s'ha fet malament i falten regles que obrin tràfic necessari. O perquè s'han afegit nous serveis a la xarxa que requereixen de nous ports que ara estan bloquejats per defecte. Així que cal modificar el firewall per permetre aquest nou tràfic.

A vegades però (ai les presses que males conselleres!!!) per resoldre ràpidament el problema de que hi ha serveis tallats per culpa del firewall DROP es decideix canviar a ACCEPT per defecte mentre "s'està arreglant" l'script del firewall.

Però això a part de resoldre el marrón de que hi havia serveis tallats, genera un altre marró que és que ara tot està obert per defecte, també coses que sabem que són nocives! El **problema** rau en que com que la política per defecte era DROP no ens em pres la molèstia

de tancar explícitament aquell tràfic que sabem que és insegur o que no desitgem. Aquell que si la política fos ACCEPT hauríem tallat.

Solució: sempre que fem una política per defecte DROP afegir al final unes regles 'extres o de seguretat' que tallin tot el tràfic que no es desitgi (si, si, encara que sigui DROP per defecte!). D'aquesta manera si es canvia la política per defecte de DROP a ACCEPT el sistema continua tenint els punts vitals protegits.

En el següent exemple es mostren regles afegides al final d'una política per defecte DROP on s'asseguren de tancar explícitament tràfic no desitjat:

```
# Barrera final DROP per si es canvia a ACCEPT temporalment
# tancar tràfic no desitjat

/sbin/iptables -A INPUT -p tcp  -m tcp   --dport 1:1024 -j DROP
/sbin/iptables -A INPUT -p udp -m udp  --dport 1:1024 -j DROP
/sbin/iptables -A INPUT -p tcp  -m tcp   --dport 1723   -j DROP
/sbin/iptables -A INPUT -p tcp  -m tcp   --dport 3306   -j DROP
/sbin/iptables -A INPUT -p tcp  -m tcp   --dport 5432   -j DROP
```

Forwarding - NAT (DNAT i SNAT)

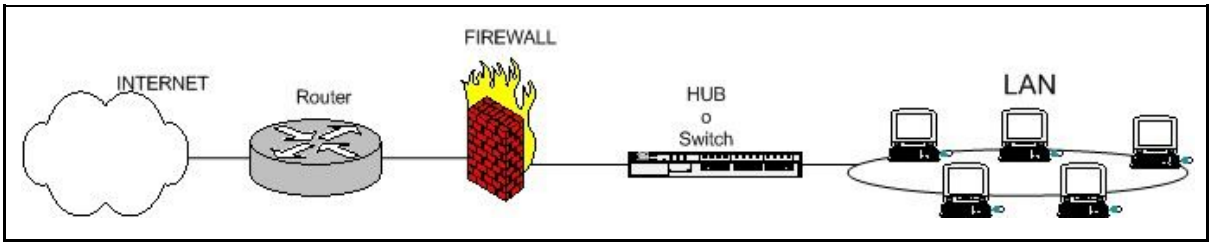
Network Addres Translation: NAT

En una xarxa local (sigui a casa o sigui empresarial) sovint tenim el model de xarxa on un router del ISP separa la xarxa privada (de casa) de la xarxa d'internet (pública). El router té dues interfícies una de pública adreçable des d'internet i una de privada que dona a la cara de dins, a la xarxa interna. Els hosts de la xarxa loval/privada/interna tenen adreces privades no adreçables des d'internet.

El següent esquema (extret de la documentació de pello www.pello.info) mostra un cas particular amb un router i un firewall, però podem considerar el cas on aquestes dues funcions les fa un mateix host rourer/firewall.

El router/firewall disposa almenys de dues interfícies de xarxa, una amb la ip pública (la externa) i una interfície interna amb una ip privada (generalment).

gràfic extret de la documentació de pello www.pello.info



Per tal de poder accedir a internet els hosts de la LAN són **'emmascarats'** amb l'adreça IP pública del router que fa NAT, Network Address Translation. És a dir, posa als paquets la seva pròpia adreça IP com origen dels paquets. En tornar les respostes el router fa un seguiment de a qui de la xarxa LAN li ha de tornar la resposta. Aquest seguiment el fa anotant en una taula cada port de sortida dinàmic que utilitza en realitat en nom de quin host local està enviant el paquet.

És a dir, el router 'parla' de cara a l'exterior en nom de cada un dels hosts de la LAN usant ports dinàmics diferents, les respostes que rep en aquests ports les retorna als hosts locals apropiats.

El tràfic entre internet (l'exterior) i la LAN (interior) es tràfic que creua el router/firewall, es tràfic al que se li aplica la regla **FORWARD**.

El següent exemple mostra com emmascarar la xarxa local 172.16.10.0/24 usant la ip pública del router/firewall de la interfície externa eth0.

```
# NAT usant la IP pública de eth0
```

```
iptables -t nat -A POSTROUTING -s 172.16.10.0/24 -o eth0 -j MASQUERADE
```

Atenció: a aquestes alçades el lector ja deu saber de sobres com fer que un host GNU/Lnux es comporti com un router, que faci encaminament o forwarding. Si el host no fa forwarding les regles FORWARD no s'apliquen!. Però per si de cas ho recordem:

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

o be:
# sysctl net.ipv4.ip_forward=1
net.ipv4.ip_forward = 1

$ sysctl -a | grep ip_forward
net.ipv4.ip_forward = 1
```

En tot script de regles iptables es pot afegir una d'aquestes instruccions per assegurar-se de que el sistema GNU/Linux actua com a router.

En el cas d'un router/firewall que separa la xarxa publica de la xarxa privada sovint els ports externs de INPUT al router estan tancats i els ports de la IP de la xarxa privada poden estar oberts per permetre l'administració del router/firewall.

Aquest és un exemple de configuració del touter/firewall que tanca l'accés al router des de l'exterior i l'obre a les connexions provinents de la xarxa local (recordeu eth1 és la interficie de la xarxa interna i eth0 la de connexió a internet):

```
iptables -A INPUT -s 0.0.0.0             -i eth0 -j DROP
iptables -A INPUT -s 172.16.10.0/24 -i eth1 -j ACCEPT
```

Forwarding

Donat un router/firewall que fa la tasca d'encaminament els paquets que el 'creuen' són subjectes a regles de **FORWARDING**. És especialment aplicable en el cas de xarxes locals separades de l'exterior per un router/firewall que fa NAT, però les regles de forwarding es poden aplicar a qualsevol router que faci encaminament.

```
# ============================================================
# Regles de: xarxa_local-(R/F)-xarxa_externa
#   LOCAL_LAN ---(em1)Router/Firewall(wlan0)-- PUBLIC_LAN
#   172.16.100.0/24                                 192.168.2.0/24
# ============================================================
# no deixar sortir de la lan a exterior a serveis concrets
iptables -A FORWARD -s 172.16.100.0/24 -p tcp --dport 7  -j REJECT
```

```
iptables -A FORWARD -s 172.16.100.11    -p tcp --dport 13 -j REJECT

# no deixar sortir de la lan a exterior segons destí
iptables -A FORWARD -s 172.16.100.0/24 -d 192.168.1.34 -p tcp --dport 23 -j REJECT
iptables -A FORWARD -i em1 -d 192.168.1.0/24 -p tcp --dport 82 -j REJECT

# un host de la lan concret no pot accedir a un servei extern (segons destí)
iptables -A FORWARD -s 172.16.100.11 -d 192.168.1.34   -p tcp --dport 110 -j REJECT
iptables -A FORWARD -s 172.16.100.11 -d 192.168.1.0/24 -p tcp --dport 143 -j REJECT

# evitar que des de dins de la LAN es falsifiqui ip origen (spoofing)
iptables -A FORWARD ! -s 172.16.100.0/24 -i em1 -j REJECT
```

Port i host forwarding

Què és el redireccionament de ports (o ports i hosts)? Consisteix en que les peticions dirigides a un determinat port (o ip:port) en lloc de ser ateses per aquest port són desviades a un altre port (o ip:port).

Cas 1: redirecció d'un servei a un servidor intern de la LAN

Per exemple en una LAN hi ha un servidor web i un servidor smtp en dues màquines diferents de la LAN interna. Des de l'exterior no es pot accedir a les maquines de la LAN. De l'exterior únicament es veu el router/firewall que fa NAT. Quan s'intenta accedir al port 80 del router/firewall es fa port forwarding i es passa la petició a un host de la LAN interna. Tot aquest tràfic HTTP des de l'exterior es pensa que es fa 'contra' el router i en realitat el router enmascara que el servidor està 'dins' de la LAN. El mateix es faria amb el tràfic smtp que en accedir al port 25 del router/firewall s'accedeix al servidor SMTP intern a la LAN.

```
# dirigir tot el tràfic http port 80 a un altre hot de la LAN

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80   -j DNAT --to 192.168.10.12:80
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 443 -j DNAT --to 192.168.10.12:443
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 25   -j DNAT --to 192.168.10.11:25
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 110 -j DNAT --to 192.168.10.11:110
```

```
# si el tràfic hhtp prové de de casa meva (221.15.15.15) enviar-lo a un altre web server
# (aquesta regla ha d'anar abans de les anteriors)

iptables -t nat -A PREROUTING -s 221.15.15.15 -i eth0 -p tcp --dport  80  -j DNAT \
                                    --to 192.168.10.10:80
iptables -t nat -A PREROUTING -s 221.15.15.15 -i eth0 -p tcp --dport  443 -j DNAT \
                                    --to 192.168.10.10:443
```

En l'exemple anterior el tràfic que rep el router dirigit als ports 80 i 443 es desvia al host de la LAN 192.168.10.12. El tràfic smtp i pop (ports 25 i 110) es dirigeix a una altra màquina, la 192.168.10.11. Si el tràfic prové del host 221.15.15.15 dirigir el tràfic http a un altre host de la LAN (el 192.168.10.10).

Cas 2: obrir ports associant-los a un mateix servei en la LAN

El port forwarding es pot utilitzar també per obrir tràfic en ports diferents dels 'usuals'. Així per exemple el router/firewall té el port extern ssh (port 22) tancat però té obert el port 2022 que fa port forwarding per accedir al port 22 del host-A de la LAN. El port 2023 del router/firewall és un camí per accedit via port forwarding al ssg (port 22) del host-B dela LAN. I així per a altres ports.

És a dir, es pot obrir un port que permeti via port forwarding accedir per ssh a cada un dels hosts interns de la LAN.

```
# obrir ports com a camins per accedir a serveis interns de la LAN

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 2022  -j DNAT --to 192.168.10.10:22
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 2023  -j DNAT --to 192.168.10.11:22
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 2024  -j DNAT --to 192.168.10.12:22
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 2025  -j DNAT --to 192.168.10.13:22
```

En aquest exemple anterior s'associen els ports del 2022-2025 els ports ssh dels hosts de la LAN 192.168.10.[10-13].

Cas 3: obrir ports associant-los a serveis interns de la LAN

De fet aquest procés de l'exemple anterior es pot realitzar per a qualsevol servei! Així per exemple en el router/firewall es poden obrir els ports que es vulguin que via port forwarding accedeixin als serveis interns de la LAN que siguin.

Per exemple associar el port 2001 al ssh del host-A, port 2002 al telnet del host-A, port 2003 al smtp del host-A, etc.

```
# obrir ports com a camins per accedir a serveis interns de la LAN

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 3001  -j DNAT --to 192.168.10.12:80
```

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 3002  -j DNAT --to
192.168.10.12:443
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 3003  -j DNAT --to 192.168.10.12:22
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 3004  -j DNAT --to 192.168.10.12:25
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 3005  -j DNAT --to 192.168.10.12:23

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 4001  -j DNAT --to 192.168.10.13:80
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 4002  -j DNAT --to
192.168.10.13:443
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 4003  -j DNAT --to 192.168.10.13:22

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 5001  -j DNAT --to 192.168.10.14:80
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 5002  -j DNAT --to
192.168.10.14:443
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 5003  -j DNAT --to 192.168.10.14:22
```

En aquests exemple s'obren els ports 3001-3005 per accedir als serveis http, https, ssh, smtp i telnet del host 192.168.10.12, els ports 4001-4003 per als serveis http, https i ssh del host 192.168.10.13. Finalment els ports 5001-5003 per accedir als serveis http, https i ssh del host LAN 192.168.10.14.

Cas 4: port forwarding segons l'adreça/port origen

També es pot realitzar port forwarding en funció de l'adreça IP, de la xarxa origen o del port origen. Per exemple als equips de la xarxa-A quan accedeixen al port 80 del router/firewall accedeixen al web-server-A i els hosts de la xarxa-B en accedir al port 80 del router/firewall accedeixen al web-server-B.

```
# si el tràfic hhtp prové de de casa meva (221.15.15.15) enviar-lo a un altre web server
# (aquesta regla ha d'anar abans de les anteriors)

iptables -t nat -A PREROUTING -s 221.15.15.15 -i eth0 -p tcp --dport  80  -j DNAT \
                                  --to 192.168.10.10:80
iptables -t nat -A PREROUTING -s 221.15.15.15 -i eth0 -p tcp --dport  443 -j DNAT \
                                  --to 192.168.10.10:443
```

```
# port forwarding segons la xarxa origen

iptables -t nat -A PREROUTING  -s .221.15.15.0/24 -i eth0 -p tcp --dport  80  -j DNAT \
                                   --to 192.168.10.10:80
iptables -t nat -A PREROUTING  -s 221.15.15.0/24 -i eth0 -p tcp --dport  443 -j DNAT \
                                   --to 192.168.10.10:443
iptables -t nat -A PREROUTING  -s .150.10.0.0/16 -i eth0 -p tcp --dport  80    -j DNAT \
                                   --to 192.168.10.10:80
iptables -t nat -A PREROUTING  -s .150.10.0.0/16 -i eth0 -p tcp --dport  443    -j DNAT \
```

```
--to 192.168.10.10:443
```

En resum, a través del port forwarding podem:

- dirigir a un altre host/port el trafic destinat a un port.
- dirigir a un altre host/port el tràfic en funció del host i/o xarxa i/o port.
- dirigir a un altre host/port el tràfic d'un determinat tipus de tràfic

Cas 5: exemples amb snat

<pendent>

Miscel·lània

Protocol ICMP

El protocl ICMP (capa 3) pot ser de diferents subtipus a part dels omnipresents **request** i **reply**. Alguns dels tipus ICMP poden ser utilitzats per a accions nocives com per exemple el **redirect**. Podeu consultar la taules de tipus ICMP:

```
<pendent>
```

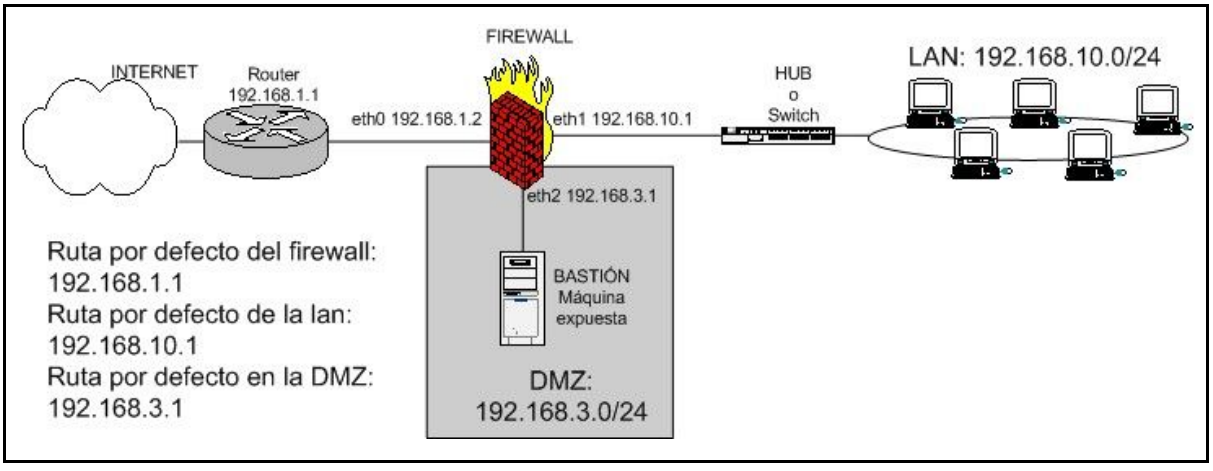
```
iptables  -A INPUT          -p icmp   --icmp-type    8   -j DROP
iptables  -A OUTPUT       -p icmp   --icmp-type    8   -j DROP

iptables  -A FORWARD   -p icmp   --icmp-type    8   -j DROP
```

Recordeu també que les regles amb acció DROP no generen cap resposta per part del firewall mentre que l'acció REJECT permet que el firewall envii una resposta ICMP indicant el rebuig de la connexió (podeu observar-ho analitzant el tràfic amb wireshark).

```
# icmp
iptables  -A INPUT  -p icmp   --icmp-type echo-reply   -j ACCEPT
iptables  -A OUTPUT -p icmp   --icmp-type echo-request -j ACCEPT
```

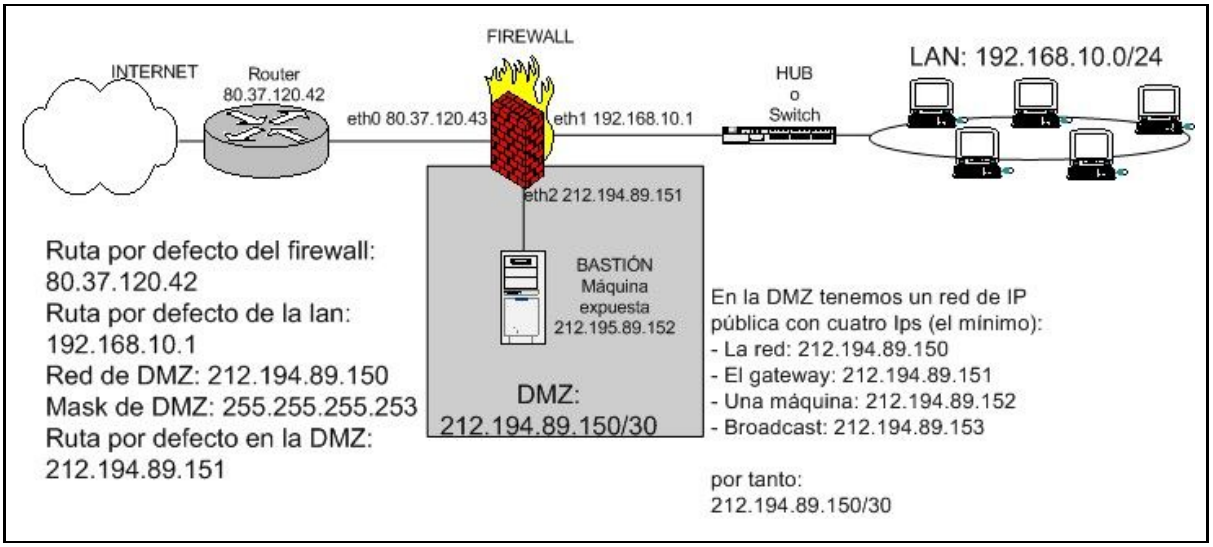
DMZ Demilitarized Zone

gràfic extret de la documentació de pello www.pello.info



VPN firewall amb tràfic de Virtual Private Networks

gràfic extret de la documentació de pello www.pello.info



Altres Topologies

La funció del firewall varia en funció de la topologia de la xarxa i de la funció de filtrat que ha de realitzar. Podem exposar per exemple:

- El model més fàcil d'entendre i aplicar els el de la xarxa privada com la de casa amb un router fent de firewall i NAT.
- El model del router empresarial que inclou una DMZ on s'allotgen els servidors exposats a l'exterior.
- Una estructura organitzativa amb dues xarxes A i B connectades per un router que filtra tràfic entre hosts de A i B segons els requeriments de l'organització, però que no necessita fer NAT.
- Ampliar l'exemple anterior amb una organització amb sortida exterior (on es requereix NAT, una DMZ amb servidors i dues xarxes A i B internes que es comuniquen a través del router (que te 4 interfícies de xarxa). I a més a més aplica regles de filttrat al tràfic entre xarxes.
- Una organització aïllada sense sortida exterior però que disposa de diverses xarxes internes amb regeles de filtrat de la comunicació entre elles i una DMZ amb servidors accessibles per a totes les xarxes internes.
- "Mas madera" i si li afegim VPNs per comunicar xarxes distants de la mateixa organització?

Exemples complerts

Cas 1: exemple de casos INPUT

Exemple de INPUT: política accept per defecte on s'apliquen regles input:

1. flush de regles
2. política accept per defecte
3. tancar accés a un servei del router/firewall (no accés extern al port 80 del router/firewall)
4. tancar accés a un servei del router/firewall si la petició externa prové d'un host determinat (el host-A no pot accedir al port 80 del router/firewall).
5. tancar accés a un servei del router/firewall si la petició externa prové d'un host en una determinada xarxa (els hosts de la xarxa-A no poden accedir al port 80 del router/firewall).
6. tancar un servei del router/firewall als hosts d'una xarxa-A però fer alguna excepció permetent a algun host de la xarxa vetada que si que accedeixi al servei del router/firewall.
7. Tancar rangs de ports no permetent accé extern a aquests ports.

```
#!/bin/bash
# ASIX M11-Seguretat i alta disponibilitat
# @edt 2015
# ============================================

# Activar si el host ha de fer de router
#echo 1 > /proc/sys/net/ipv4/ip_forward

# Regles Flush: buidar les regles actuals
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# Establir la politica per defecte (ACCEPT o DROP)
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

# Permetre totes les pròpies connexions via localhost
iptables -A INPUT   -i lo -j ACCEPT
iptables -A OUTPUT  -o lo -j ACCEPT

# Permetre tot el trafic de la pròpia ip(192.168.1.34))
iptables -A INPUT   -s 192.168.1.34   -j ACCEPT
```

```
iptables -A OUTPUT  -d 192.168.1.34   -j ACCEPT

# ------------------------------------------------------
# es denega l'accés al servei local del host/firewall
# de web, telnet i ftp (tràfic tcp) i tftp (tràfic udp)

iptables -A INPUT  -p tcp   --dport 80          -j DROP
iptables -A INPUT  -p tcp   --dport 23          -j REJECT
iptables -A INPUT  -p tcp   --dport 20:21       -j DROP
#iptables -A INPUT  -p tcp   --dport 13         -j REJECT

# es denega l'accés al servei "servei" local del host/firewall
# si l'origen és un host concret
# host 192.168.2.51 no pot accedir a ftp ni web

#iptables -A INPUT -s 192.168.2.51  -p tcp   --dport 7  -j DROP
iptables -A INPUT -s 192.168.2.51    -p tcp   --dport 81 -j REJECT

# es denega l'accés al servei "servei" local del host/firewall
# si l'origen és una xarxa concreta
# xarxa 192.168.2.0/24 no accés al servei postgresql
# qualsevol xarxa no accés al servei telnet

iptables -A INPUT -s 192.168.2.0/24 -p tcp   --dport 110   -j DROP
iptables -A INPUT -s 0.0.0.0/0          -p tcp   --dport 143   -j DROP

# Tancar un servei a tots els hosts excepte els
# indicats.
# host 192.168.2.51 accés als serveis echo, daytime. Alstres host denegat
# xarxa 192.168.2.0/24 accés al servei ssh, altres hosts/xarxes denegat

iptables -A INPUT -s 192.168.2.51    -p tcp   --dport 7   -j ACCEPT
iptables -A INPUT                -p tcp   --dport 7         -j DROP

iptables -A INPUT -s 192.168.2.0/24   -p tcp  --dport 13       -j ACCEPT
iptables -A INPUT                -p tcp  --dport 13        -j DROP

# Tancar els ports privilegiats 0-1023
# atenció: és molt perillós si no ens hem assegurat d'obrir serveis necessaris
#iptables -A INPUT -p tcp --dport 1:1024 -j DROP
3iptables -A INPUT -p udp --dport 1:1024 -j DROP

# Tancar explícitament ports o rangs de ports
#iptables -A INPUT -p tcp   --dport 10000     -j DROP
#iptables -A INPUT -p tcp   --dport 10000     -j DROP
#iptables -A INPUT -p tcp   --dport 2000:3000 -j DROP

# show
iptables -L
```

Cas 2: exemple de casos OUTPUT

Exemple de INPUT: política accept per defecte on s'apliquen regles input:

1. flush de regles
2. política accept per defecte
3. tancar accés a un servei extern, des del router/firewall no es té accés a quest servei ofert per qualsevol altre host.
4. tancar l'aaccés a un servei extern (de qualsevol host extern) excepte a un servidor forani concret. Des del router/firewall no es pot accedir a un servei a no ser que sigui el que ofereix el host-A concret.
5. Des del router/firewall no es pot accedir a un servei ofert per hosts externs a no ser que el servidor que l'ofereix sigui un host dins de una xarxa-A concreta.
6. No permetre que el router/firewall tingui accés a un servei extern si aquest servei l'ofereix una xarxa externa concreta, però fer una excepció amb determinats hosts servidors. Tot i estar en la xarxa-A vetada si el servidor és el host-A si que el router/firewall hi pot accedir.
7. Tancar rangs de ports no permetent l'acces a aquests ports en servidors externs.

```
#!/bin/bash
# ASIX M11-Seguretat i alta disponibilitat
# @edt 2015
# ============================================
# Exemples de regles OUTPUT
# host i19 (192.168.2.49)
# paquets: xinetd, telnet, telnet-server, httpd, uw-imap
# fets amb xinetd: echo-stream(7), daytime-stream(13),
# daytime2(82), ipop3(110), imap(143), https2(81)
# stand-alone: httpd, xinetd
# ============================================
# Activar si el host ha de fer de router
#echo 1 > /proc/sys/net/ipv4/ip_forward

# Regles Flush: buidar les regles actuals
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# Establir la politica per defecte (ACCEPT o DROP)
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

# Permetre totes les pròpies connexions via localhost
iptables -A INPUT   -i lo -j ACCEPT
iptables -A OUTPUT  -o lo -j ACCEPT
```

```
# Permetre tot el trafic de la pròpia ip(192.168.1.49))
iptables -A INPUT   -s 192.168.1.49   -j ACCEPT
iptables -A OUTPUT  -d 192.168.1.49   -j ACCEPT
#exit 0
# ------------------------------------------------------
# es denega l'accés al servei de web, telnet i ftp (tràfic tcp)
# i tftp (tràfic udp) de fora

iptables -A OUTPUT  -p tcp   --dport 80        -j DROP
iptables -A OUTPUT  -p tcp   --dport 23        -j REJECT
iptables -A OUTPUT  -p tcp   --dport 20:21     -j DROP
#iptables -A OUTPUT  -p tcp   --dport 13       -j REJECT
# exit 0
# es denega l'accés al servei "servei" si el desti és un host concret
# al host 192.168.2.51 es no pot accedir a echo-stream ni web (hhtp2)

#iptables -A OUTPUT -d 192.168.2.51  -p tcp   --dport 7  -j DROP
iptables -A OUTPUT -d 192.168.2.51  -p tcp   --dport 81 -j REJECT
# exit 0
# es denega l'accés al servei "servei" po3 i imap si el desti és
# una xarxa  concreta
# no acces al servei pop3 de la xarxa 192.168.2.0/24
# no acces al servei imap de qualsevol xarxa externa

iptables -A OUTPUT -d 192.168.2.0/24  -p tcp   --dport 110   -j DROP
iptables -A OUTPUT -d 0.0.0.0/0       -p tcp   --dport 143   -j DROP
# exit 0
# Tancar acces al servei a hosts externs excepte als hosts indicats.
# no access al servei echo(7) excepte al del host 192.168.2.51
# no access al servei daytime(13) excepte a servidors de la xarxa
# 192.168.2.0/24

iptables -A OUTPUT -d 192.168.2.51          -p tcp   --dport 7  -j ACCEPT
iptables -A OUTPUT                          -p tcp   --dport 7  -j DROP

iptables -A OUTPUT -d 192.168.2.0/24   -p tcp  --dport 13  -j ACCEPT
iptables -A OUTPUT                          -p tcp  --dport 13  -j DROP
exit 0

# Tancar els ports privilegiats 0-1023
# atenció: és molt perillós si no ens hem assegurat d'obrir serveis necessaris
#iptables -A OUTPUT -p tcp --dport 1:1024 -j DROP
#iptables -A OUTPUT -p udp --dport 1:1024 -j DROP

# Tancar explícitament ports o rangs de ports
#iptables -A OUTPUT -p tcp   --dport 10000            -j DROP
#iptables -A OUTPUT -p tcp   --dport 10000            -j DROP
#iptables -A OUTPUT -p tcp   --dport 2000:3000 -j DROP
```

```
# show
iptables -L
```

Cas 3: exemple de regles de flux de trafic relacionat, establert, new ...

- tancar l'accés al servidor wbe del router/firewall però permetre que el reouter/firewall navegui per internet. És a dir, per exemple, no s'accepten connexions noves entrants als ports 80, 443. Però si el tràfic entrant que és resposta a tràfic http generat per el propi router/firewall. També es permet tràfic http now de sortida del router/firewall a l'exterior.
- tancar l'accés a internet des del propi router/firewall, però permetre que externament s'accedeixi al servidor web (80, 443) del router/firewall. És a dir, per exemple, no es permet tràfic http de sortida nou però si el que és resposta http a tràfic ja establer de clients que s'han connectat al servidor web. Cal permetre entrada de tràfic http al servei web del router/firewall.

```
#!/bin/bash
# ASIX M11-Seguretat i alta disponibilitat
# @edt 2015
# ============================================
# Exemples de regles OUTPUT
# host i19 (192.168.2.49)
# paquets: xinetd, telnet, telnet-server, httpd, uw-imap
# fets amb xinetd: echo-stream(7), daytime-stream(13),
# daytime2(82), ipop3(110), imap(143), https2(81)
# stand-alone: httpd, xinetd
# ============================================
# Activar si el host ha de fer de router
#echo 1 > /proc/sys/net/ipv4/ip_forward

# Regles Flush: buidar les regles actuals
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# Establir la politica per defecte (ACCEPT o DROP)
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

# Permetre totes les pròpies connexions via localhost
iptables -A INPUT   -i lo -j ACCEPT
iptables -A OUTPUT  -o lo -j ACCEPT
```

```
# Permetre tot el trafic de la pròpia ip(192.168.1.49))
iptables -A INPUT   -s 192.168.1.49   -j ACCEPT
iptables -A OUTPUT  -d 192.168.1.49   -j ACCEPT
#exit 0
# ---------------------------------------------------------
# es denega l'accés al propi servei web
# de fora cap al router/firewall, però el r/f si pot navegar
iptables -A INPUT  -p tcp       --dport 80  -j DROP
iptables -A INPUT  -p tcp       --dport 443  -j DROP
/sbin/iptables -A INPUT  -p tcp  -m tcp   --sport 80   -m state  \
                    --state RELATED,ESTABLISHED   -j ACCEPT
/sbin/iptables -A OUTPUT -p tcp  -m tcp   --dport 80   -j ACCEPT
/sbin/iptables -A INPUT  -p tcp  -m tcp   --sport 443  -m state \
                    --state RELATED,ESTABLISHED   -j ACCEPT
/sbin/iptables -A OUTPUT -p tcp  -m tcp   --dport 443  -j ACCEPT

# es denega que des del router/firewall s'accdeixi al servei
# daytime(13) però s'ofereix aquest servei a l'exterior
/sbin/iptables -A OUTPUT -p tcp  --dport 13             -j DROP
/sbin/iptables -A INTPUT -p tcp  -m tcp   --dport 13  -j ACCEPT
/sbin/iptables -A OUTPUT -p tcp  -m tcp   --sport 13  -m state  \
                    --state RELATED,ESTABLISHED   -j ACCEPT
#exit 0
# es denega que des del router/firewall s'accdeixi al servei
# web(80,443) però s'ofereix aquest servei a l'exterior
/sbin/iptables -A OUTPUT  -p tcp   --dport 80           -j DROP
/sbin/iptables -A OUTPUT  -p tcp   --dport 443          -j DROP
/sbin/iptables -A INTPUT  -p tcp   -m tcp   --dport 80  -j ACCEPT
/sbin/iptables -A OUTPUT  -p tcp   -m tcp   --sport 80  -m state  \
                    --state RELATED,ESTABLISHED   -j ACCEPT
/sbin/iptables -A INTPUT  -p tcp   -m tcp   --dport 443 -j ACCEPT
/sbin/iptables -A OUTPUT  -p tcp   -m tcp   --sport 443 -m state \
          --state RELATED,ESTABLISHED  -j ACCEPT
#exit 0

# permetre al host/firewall accedir a servidors ftp
# en aquest cas el host/firewall no té servei ftp
# atenció a permetre ftp-actiu i ftp-passiu
/sbin/iptables -A INPUT  -p tcp   -m tcp   --sport 20:21   -m state \
                            --state RELATED,ESTABLISHED   -j ACCEPT
/sbin/iptables -A OUTPUT -p tcp   -m tcp   --dport 20:21   -j ACCEPT
/sbin/iptables -A INPUT  -p tcp   -m tcp   --sport 1024:65535 \
          --dport 1024:65535 -m state --state ESTABLISHED  -j ACCEPT
/sbin/iptables -A OUTPUT  -p tcp  -m tcp   --dport 1024:65535 -m state \
                    --state NEW,RELATED,ESTABLISHED   -j ACCEPT
iptables -L
```

Cas 4: Política DROP per defecte

En establir una polítcia drop per defcete cal assegurar-se que:

- primerament s'estableix la política drop per defecte.
- s'obren tots els ports necessàris per al funcionament dela màquina.
- especial atenció a ports de serveis com: dns, chrony, etc.
- generar una 'barrera' de protecció per si hi ha un canvi de política

```
#!/bin/bash
# ASIX M11-Seguretat i alta disponibilitat
# @edt 2015
# ================================================
# Exemples de regles OUTPUT
# host i19 (192.168.2.49)
# paquets: xinetd, telnet, telnet-server, httpd, uw-imap
# fets amb xinetd: echo-stream(7), daytime-stream(13),
# daytime2(82), ipop3(110), imap(143), https2(81)
# stand-alone: httpd, xinetd
# ================================================
# Activar si el host ha de fer de router
#echo 1 > /proc/sys/net/ipv4/ip_forward

# Regles Flush: buidar les regles actuals
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# Establir la politica per defecte (ACCEPT o DROP)
iptables -P INPUT DROP
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

# Permetre totes les pròpies connexions via localhost
iptables -A INPUT   -i lo -j ACCEPT
iptables -A OUTPUT  -o lo -j ACCEPT

# Permetre tot el trafic de la pròpia ip(192.168.1.49))
iptables -A INPUT   -s 192.168.1.35   -j ACCEPT
iptables -A OUTPUT  -d 192.168.1.35   -j ACCEPT
#exit 0
# ------------------------------------------------------
# consulta dns primari
iptables -A INPUT   -s 80.58.61.250  -p udp -m udp --sport 53 -j ACCEPT
iptables -A OUTPUT  -d 80.58.61.250  -p udp -m udp --dport 53 -j ACCEPT
# consulta dns secundari
iptables -A INPUT   -s 80.58.61.254  -p udp -m udp --sport 53 -j ACCEPT
iptables -A OUTPUT  -d 80.58.61.254  -p udp -m udp --dport 53 -j ACCEPT
# consulta ntp
iptables -A INPUT   -p udp -m udp  --dport 123 -j ACCEPT
iptables -A OUTPUT  -p udp -m udp  --sport 123 -j ACCEPT
```

```
# ---------------------------------------------------------
# servei cups
iptables -A INPUT   -p tcp --dport 631  -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 631  -j ACCEPT
# port xinetd
iptables -A INPUT   -p tcp --dport 3411 -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 3411 -j ACCEPT
# port x11-x-forwarding
iptables -A INPUT   -p tcp --dport 6010 -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 6010 -j ACCEPT
# servei rpc
iptables -A INPUT   -p tcp --dport 111 -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 111 -j ACCEPT
# ---------------------------------------------------------
# icmp
iptables  -A INPUT  -p icmp   --icmp-type echo-reply   -j ACCEPT
iptables  -A OUTPUT -p icmp   --icmp-type echo-request -j ACCEPT
# ---------------------------------------------------------
# servei ssh
iptables -A INPUT   -p tcp --dport 22   -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 22   -j ACCEPT
# servei http
iptables -A INPUT   -p tcp --dport 80   -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 80   -j ACCEPT
# servei daytime
iptables -A INPUT   -p tcp --dport 13   -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 13   -j ACCEPT
# servei echo
iptables -A INPUT   -p tcp --dport 7    -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 7 -j ACCEPT
# servei smtp
iptables -A INPUT   -p tcp --dport 25 -j ACCEPT
iptables -A OUTPUT  -p tcp --sport 25 -j ACCEPT
# exit 0

# ============================================================
# atencio: cal millorar aquestes regles!!!                *****
# cal que nomes es permeti trafic relacionat                    *****
# ============================================================
# navegar web
#iptables -A INPUT    -p tcp  --sport 80  -j ACCEPT
iptables -A OUTPUT   -p tcp  --dport 80  -j ACCEPT
iptables -A INPUT      -p tcp  --sport 443 -j ACCEPT
iptables -A OUTPUT   -p tcp  --dport 443 -j ACCEPT
# accedir a servei echo
#iptables -A INPUT    -p tcp  --sport 7  -j ACCEPT
iptables -A OUTPUT   -p tcp  --dport 7  -j ACCEPT
# accedir al servei daytime
#iptables -A INPUT    -p tcp  --sport 13  -j ACCEPT
iptables -A OUTPUT   -p tcp  --dport 13  -j ACCEPT
```

```
# accedir al servei ssh
#iptables -A INPUT     -p tcp  --sport 22  -j ACCEPT
iptables -A OUTPUT   -p tcp  --dport 22  -j ACCEPT

 iptables -A INPUT -m state --state ESTABLISHED,RELATED       -j ACCEPT
 iptables -A OUTPUT -m state --state ESTABLISHED,RELATED    -j ACCEPT

# ---------------------------------------------------------------
# ftp i tftp (trafic udp)                                        *****
# ---------------------------------------------------------------
# oferir servei tftp
iptables -A INPUT   -p udp  --dport 69  -j ACCEPT
iptables -A OUTPUT  -p udp  --sport 69  -j ACCEPT
# accedir a serveis tftp externs
iptables -A INPUT   -p udp  --sport 69  -j ACCEPT
iptables -A OUTPUT  -p udp  --dport 69  -j ACCEPT
  # pendent obrir ports dinamics
# oferir servei ftp
iptables -A INPUT       -p tcp  --dport 20:21  -j ACCEPT
iptables -A OUTPUT   -p tcp  --sport 20:21  -j ACCEPT
# accdir a serveis ftp externs
iptables -A INPUT       -p tcp  --sport 20:21  -j ACCEPT
iptables -A OUTPUT   -p tcp  --dport 20:21  -j ACCEPT
  # pendent obrir ports dinamics!

# ---------------------------------------------------------------
# Barrera per tancar els serveis/ports en cas de passar a drop
# ...

iptables -L
```

Cas 5: Network Address Translation i Port forwarding

```
#!/bin/bash
# ASIX M11-Seguretat i alta disponibilitat
# @edt 2015
# ================================================
# Exemples de regles OUTPUT
# host i19 (192.168.2.49)
# paquets: xinetd, telnet, telnet-server, httpd, uw-imap
# fets amb xinetd: echo-stream(7), daytime-stream(13),
# daytime2(82), ipop3(110), imap(143), https2(81)
# stand-alone: httpd, xinetd
# ================================================
# Activar si el host ha de fer de router
echo 1 > /proc/sys/net/ipv4/ip_forward

# Regles Flush: buidar les regles actuals
```

```
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# Establir la politica per defecte (ACCEPT o DROP)
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

# Permetre totes les pròpies connexions via localhost
iptables -A INPUT   -i lo -j ACCEPT
iptables -A OUTPUT  -o lo -j ACCEPT

# Permetre tot el trafic de la propia ip exterior (wlan0:192.168.1.41))
iptables -A INPUT   -s 192.168.1.41   -j ACCEPT
iptables -A OUTPUT  -d 192.168.1.41   -j ACCEPT

# Permetre tot el trafic de la propia ip interior (em1:172.16.100.1))
iptables -A INPUT   -s 172.16.100.1   -j ACCEPT
iptables -A OUTPUT  -d 172.16.100.1   -j ACCEPT
#exit 0

# Ahora hacemos enmascaramiento de la red local (HACIA FUERA)
iptables -t nat -A POSTROUTING -s 172.16.100.0/24 -o wlan0 -j MASQUERADE

#
================================================================
==
# Regles de: xarxa_local-(R/F)-xarxa_externa
#   LOCAL_LAN ---(em1)Router/Firewall(wlan0)-- PUBLIC_LAN
#
================================================================
==
# no deixar sortir de la lan a exterior a serveis concrets
#iptables -A FORWARD -s 172.16.100.0/24 -p tcp --dport 7  -j REJECT
#iptables -A FORWARD -s 172.16.100.11   -p tcp --dport 13 -j REJECT

# no deixar sortir de la lan a exterior segons destí
#iptables -A FORWARD -s 172.16.100.0/24 -d 192.168.1.34 -p tcp --dport 23 -j REJECT
#iptables -A FORWARD -i em1 -d 192.168.1.0/24 -p tcp --dport 82 -j REJECT

# un host de la lan concret no pot accedir a un servei extern (segons destí)
#iptables -A FORWARD -s 172.16.100.11 -d 192.168.1.34   -p tcp --dport 110 -j REJECT
#iptables -A FORWARD -s 172.16.100.11 -d 192.168.1.0/24 -p tcp --dport 143 -j REJECT

# evitar que des de dins de la LAN es falsifiqui ip origen (spoofing)
#iptables -A FORWARD ! -s 172.16.100.0/24 -i em1 -j REJECT
```

```
# evitar que tràfic extern entri a la lan
#Incorrecte: iptables -A FORWARD -i wlan0 -d 172.16.100.0/24  -j REJECT
# aquesta regla nopermet res que entri a la lan, no anira res perque no hi ha
# retorn
#exit 0
# ============================================================
# Regles de: xarxa_local-(R/F)-xarxa_externa PORT FORWARDING
#   LOCAL_LAN ---(em1)Router/Firewall(wlan0)-- PUBLIC_LAN
# ============================================================
# ports del router/firewall que porten a hosts de la lan
#iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 7   -j DNAT --to 172.16.100.11:7
#iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 13  -j DNAT --to 172.16.100.11:13
#iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 23  -j DNAT --to 172.16.100.11:7
#iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 110 -j DNAT --to
172.16.100.11:13

# associar ports del router/firewall a host/port de la lan
#iptables -t nat -A PREROUTING  -i wlan0 -p tcp --dport 30001 -j DNAT --to
172.16.100.11:7
#iptables -t nat -A PREROUTING  -i wlan0 -p tcp --dport 30002 -j DNAT --to
172.16.100.11:13
#iptables -t nat -A PREROUTING  -i wlan0 -p tcp --dport 30003 -j DNAT --to
172.16.100.11:110
#iptables -t nat -A PREROUTING  -i wlan0 -p tcp --dport 30004 -j DNAT --to
172.16.100.11:143
#iptables -t nat -A PREROUTING  -i wlan0 -p tcp --dport 30005 -j DNAT --to
172.16.100.11:22
#iptables -t nat -A PREROUTING  -i wlan0 -p tcp --dport 30011 -j DNAT --to
172.16.100.12:7
#iptables -t nat -A PREROUTING  -i wlan0 -p tcp --dport 30012 -j DNAT --to
172.16.100.12:13
#exit 0

# port forwarding també en funció de la ip origen
iptables -t nat -A PREROUTING  -s 192.168.1.34   -i wlan0 -p tcp --dport 13 -j DNAT \
                        --to 172.16.100.11:7
iptables -t nat -A PREROUTING  -s 192.168.1.0/24 -i wlan0 -p tcp --dport 30007 -j DNAT \
                        --to 172.16.100.11:7
iptables -t nat -A PREROUTING  -s 192.168.1.34   -i wlan0 -p tcp --dport 30022 -j DNAT \
                        --to 172.16.100.11:22
iptables -t nat -A PREROUTING  -s 192.168.1.34   -i wlan0 -p tcp --dport 30023 -j DNAT \
                        --to 172.16.100.12:22

# LListar regles
iptables -L
```

Cas 6: Demilitarized zone: DMZ

```
#!/bin/bash
# ASIX M11-Seguretat i alta disponibilitat
# @edt 2015
# ================================================
# Exemples de regles OUTPUT
# host i19 (192.168.2.49)
# paquets: xinetd, telnet, telnet-server, httpd, uw-imap
# fets amb xinetd: echo-stream(7), daytime-stream(13),
# daytime2(82), ipop3(110), imap(143), https2(81)
# stand-alone: httpd, xinetd
# ================================================
# Activar si el host ha de fer de router
echo 1 > /proc/sys/net/ipv4/ip_forward

# Regles Flush: buidar les regles actuals
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# Establir la politica per defecte (ACCEPT o DROP)
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

# Permetre totes les pròpies connexions via localhost
iptables -A INPUT   -i lo -j ACCEPT
iptables -A OUTPUT  -o lo -j ACCEPT

# Permetre tot el trafic de la propia ip exterior (wlan0:192.168.1.41))
iptables -A INPUT   -s 192.168.1.41   -j ACCEPT
iptables -A OUTPUT  -d 192.168.1.41   -j ACCEPT

# Permetre tot el trafic de la propia ip interior (em1:172.16.100.1))
iptables -A INPUT   -s 172.16.100.1   -j ACCEPT
iptables -A OUTPUT  -d 172.16.100.1   -j ACCEPT

# Permetre tot el trafic de la propia ip interior (em1:172.32.1.1))
iptables -A INPUT   -s 172.32.1.1   -j ACCEPT
iptables -A OUTPUT  -d 172.32.1.1   -j ACCEPT
#exit 0

# Ahora hacemos enmascaramiento de la red local (HACIA FUERA)
iptables -t nat -A POSTROUTING -s 172.16.100.0/24 -o wlan0 -j MASQUERADE
iptables -t nat -A POSTROUTING -s 172.32.1.0/24   -o wlan0 -j MASQUERADE
```

```
#exit 0

# ========================================================
# Regles de: xarxa_local-(R/F)-xarxa_externa
#
#   LOCAL_LAN ---(em1)Router/Firewall(wlan0)-- PUBLIC_LAN
#                           |
#                           DMZ (em1)
# ========================================================
# de la lan no accedir al router/fireall, excepte ssh, telnet i daytime
iptables -A INPUT -s 172.16.100.0/24 -p tcp --dport 22  -j ACCEPT
iptables -A INPUT -s 172.16.100.0/24 -p tcp --dport 23  -j ACCEPT
iptables -A INPUT -s 172.16.100.0/24 -p tcp --dport 13  -j ACCEPT
iptables -A INPUT -s 172.16.100.0/24 -j DROP
#exit 0

# de la lan NO es pot navegar,telnet,ssh a fora
iptables -A FORWARD -s 172.16.100.0/24 -p tcp --dport 80 -j DROP
iptables -A FORWARD -s 172.16.100.0/24 -p tcp --dport 23 -j DROP
iptables -A FORWARD -s 172.16.100.0/24 -p tcp --dport 22 -j DROP

# de la lan acces al servidor web de la dmz: intranet
iptables -A INPUT -s 172.16.100.0/24 -p tcp --dport 80  -j ACCEPT
iptables -A INPUT -s 172.16.100.0/24 -p tcp --dport 443  -j ACCEPT

# de exterior acces ports 3081:3085 redirigits a servers dins dmz
iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 3081 -j DNAT \
                        --to 172.32.1.11:81
iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 3082 -j DNAT \
                        --to 172.32.1.11:82
iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 3083 -j DNAT \
                        --to 172.32.1.11:83
iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 3084 -j DNAT \
                        --to 172.32.1.11:84
iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 3085 -j DNAT \
                        --to 172.32.1.11:85

# des de exteriors ports 3022:3025 redirigits a ssh host de la lan
iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 3022 -j DNAT \
                        --to 172.16.100.11:22
iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 3023 -j DNAT \
                        --to 172.16.100.12:22
iptables -t nat -A PREROUTING -i wlan0 -p tcp --dport 3024 -j DNAT \
                        --to 172.16.100.13:22

# NO - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# de la lan accedir a serveis dmz: 81(http2),82(echo),83(daytime),
# 84(po3),85(imap) altres serveis tancats
#iptables -t nat -A PREROUTING -s 172.16.100.0/24 -d 172.32.2.11 -p tcp \
```

```
#                               --dport 81 -j DNAT --to 172.32.1.11:81
#iptables -t nat -A PREROUTING -s 172.16.100.0/24 -d 172.32.2.0/24 -p tcp \
#                               --dport 82 -j DNAT --to 172.32.1.11:82
#iptables -A FORWARD -s 172.16.100.0/24 -d 172.32.1.0/24 -p tcp
#                                       --dport 81:85 -j ACCEPT
#iptables -A FORWARD -s 172.16.100.0/24 -d 172.32.1.0/24 -j DROP
#exit 0

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# LListar regles
iptables -L
```

Firewalld

<under under construction>

Man / Exemples iptables

iptables

extracte del man (8) iptables

TARGETS

A firewall rule specifies criteria for a packet and a target. If the packet does not match, the next rule in the chain is examined; if it does match, then the next rule is specified by the value of the target, which can be the name of a user-defined chain, one of the targets described in **iptables-extensions(8)**, or one of the special values ACCEPT, DROP or RETURN.

ACCEPT means to let the packet through. **DROP** means to drop the packet on the floor. **RETURN** means stop traversing this chain and resume at the next rule in the previous (calling) chain. If the end of a built-in chain is reached or a rule in a built-in chain with target RETURN is matched, the target specified by the chain policy determines the fate of the packet.

TABLES

There are currently five independent tables (which tables are present at any time depends on the kernel configuration options and which modules are present).

-t, --table table
This option specifies the packet matching table which the command should operate on.

The tables are as follows:

filter:

This is the default table (if no -t option is passed). It contains the built-in chains **INPUT** (for packets destined to local sockets), **FORWARD** (for packets being routed through the box), and **OUTPUT** (for locally-generated packets).

nat:

This table is consulted when a packet that creates a **new connection** is encountered. It consists of three built-ins: **PREROUTING** (for altering packets as

soon as they come in), **OUTPUT** (for altering locally-generated packets before routing), and **POSTROUTING** (for altering packets as they are about to go out).

mangle:

This table is used for specialized packet alteration. Since kernel 2.4.18, three built-in chains are also supported: INPUT (for packets coming into the box itself), FORWARD (for altering packets being routed through the box), and POSTROUTING (for altering packets as they are about to go out).

raw:

This table is used mainly for configuring exemptions from connection tracking in combination with the NOTRACK target.

security:

This table is used for Mandatory Access Control (MAC) networking rules, such as those enabled by the SECMARK and CONNSECMARK targets. Mandatory Access Control is implemented by Linux Security Modules such as SELinux.

OPTIONS COMMANDS

-A, --append chain rule-specification

Append one or more rules to the end of the selected chain. When the source and/or destination names resolve to more than one address, a rule will be added for each possible address combination.

-C, --check chain rule-specification

Check whether a rule matching the specification does exist in the selected chain. This command uses the same logic as -D to find a matching entry, but does not alter the existing iptables configuration and uses its exit code to indicate success or failure.

-D, --delete chain rule-specification
-D, --delete chain rulenum

Delete one or more rules from the selected chain. There are two versions of this command: the rule can be specified as a number in the chain (starting at 1 for the first rule) or a rule to match.

-I, --insert chain [rulenum] rule-specification

Insert one or more rules in the selected chain as the given rule number. So, if the rule number is 1, the rule or rules are inserted at the head of the chain. This is also the default if no rule number is specified.

-R, --replace chain rulenum rule-specification

Replace a rule in the selected chain. If the source and/or destination names resolve to multiple addresses, the command will fail. Rules are numbered starting at 1.

-L, --list [chain]

List all rules in the selected chain. If no chain is selected, all chains are listed. Like every other iptables command, it applies to the specified table (filter is the default), so NAT rules get listed by iptables -t nat -n -L.
Please note that it is often used with the -n option, in order to avoid long reverse DNS lookups. It is legal to specify the -Z (zero) option as well, in which case the chain(s) will be atomically listed and zeroed. The exact output is affected by the other arguments given. The exact rules are suppressed until you use iptables -L -v

-S, --list-rules [chain]

Print all rules in the selected chain. If no chain is selected, all chains are printed like iptables-save. Like every other iptables command, it applies to the specified table (filter is the default).

-F, --flush [chain]

Flush the selected chain (all the chains in the table if none is given). This is equivalent to deleting all the rules one by one.

-Z, --zero [chain [rulenum]]

Zero the packet and byte counters in all chains, or only the given chain, or only the given rule in a chain. It is legal to specify the -L, --list (list) option as well, to see the counters immediately before they are cleared. (See above.)

-N, --new-chain chain

Create a new user-defined chain by the given name. There must be no target of that name already.

-X, --delete-chain [chain]

Delete the optional user-defined chain specified. There must be no references to the chain. If there are, you must delete or replace the referring rules before the chain can be deleted. The chain must be empty, i.e. not contain any rules. If no argument is given, it will attempt to delete every non-builtin chain in the table.

-P, --policy chain target

Set the policy for the chain to the given target. See the section TARGETS for the legal targets. Only built-in (non-user-defined) chains can have policies, and neither built-in nor user-defined chains can be policy targets.

-E, --rename-chain old-chain new-chain

Rename the user specified chain to the user supplied name. This is cosmetic, and has no effect on the structure of the table.

-h

Help. Give a (currently very brief) description of the command syntax.

OPTION PARAMETERS

The following parameters make up a rule specification (as used in the add, delete, insert, replace and append commands).

-4, --ipv4
-6, --ipv6

[!] -p, --protocol protocol

The protocol of the rule or of the packet to check. The specified protocol can be one of **tcp**, **udp**, **udplite**, **icmp**, **icmpv6**, **esp**, **ah**, **sctp**, **mh** or the special keyword "all", or it can be a numeric value, representing one of these protocols or a different one. A protocol name from /etc/protocols is also allowed.
A "**!**" argument before the protocol **inverts** the test. The number zero is equivalent to all. "all" will match with all protocols and is taken as **default** when this option is omitted.

[!] -s, --source address[/mask][,...]

Source specification. Address can be either a **network** name, a **hostname**, a network IP address (with /mask), or a plain IP address.
Hostnames will be resolved once only, before the rule is submitted to the kernel. Please note that specifying any name to be resolved with a remote query such as DNS is a really bad idea.
The mask can be either an ipv4 network mask (for iptables) or a plain number, specifying the number of 1's at the left side of the network mask. Thus, an iptables mask of 24 is equivalent to 255.255.255.0.
A "!" argument before the address specification inverts the sense of the address. The flag --src is an alias for this option. Multiple addresses can be specified, but this will expand to multiple rules (when adding with -A), or will cause multiple rules to be deleted (with -D).

[!] -d, --destination address[/mask][,...]

Destination specification. See the description of the -s (source) flag for a detailed description of the syntax. The flag --dst is an alias for this option.

-m, --match match

Specifies a match to use, that is, an extension module that tests for a specific property. The set of matches make up the condition under which a target is invoked. Matches are evaluated first to last as specified on the command line and work in short-circuit fashion, i.e. if one extension yields false, evaluation will stop.

-j, --jump target

This specifies the target of the rule; i.e., what to do if the packet matches it. The target can be a user-defined chain (other than the one this rule is in), one of the special builtin targets which decide the fate of the packet immediately, or an extension (see EXTENSIONS below). If this option is omitted in a rule (and -g is not used), then matching the rule will have no effect on the packet's fate, but the counters on the rule will be incremented.

-g, --goto chain

This specifies that the processing should continue in a user specified chain. Unlike the --jump option return will not continue processing in this chain but instead in the chain that called us via --jump.

[!] -i, --in-interface name

Name of an interface via which a packet was received (only for packets entering the **INPUT**, **FORWARD** and **PREROUTING** chains). When the **"!"** argument is used before the interface name, the sense is inverted. If the interface name ends in a "+", then any interface which begins with this name will match. If this option is omitted, any interface name will match.

[!] -o, --out-interface name

Name of an interface via which a packet is going to be sent (for packets entering the **FORWARD**, **OUTPUT** and **POSTROUTING** chains). When the **"!"** argument is used before the interface name, the sense is inverted. If the interface name ends in a "+", then any interface which begins with this name will match. If this option is omitted, any interface name will match.

[!] -f, --fragment

This means that the rule only refers to second and further IPv4 fragments of fragmented packets.

-c, --set-counters packets bytes

This enables the administrator to initialize the packet and byte counters of a rule (during INSERT, APPEND, REPLACE operations).

OTHER OPTIONS

-v, --verbose

Verbose output. This option makes the list command show the interface name, the rule options (if any), and the TOS masks. For appending, insertion, deletion and replacement, this causes detailed information on the rule or rules to be printed. -v may be specified **multiple times** to possibly emit more detailed debug statements.

-w, --wait

Wait for the xtables lock. To prevent multiple instances of the program from running concurrently.

-n, --numeric

Numeric output. IP addresses and port numbers will be printed in numeric format. By default, the program will try to display them as host names, network names, or services (whenever applicable).

-x, --exact

Expand numbers. Display the exact value of the packet and byte counters, instead of only the rounded number.

--line-numbers

When listing rules, add line numbers to the beginning of each rule, corresponding to that rule's position in the chain.

Exemples de pràctiques a realitzar amb iptables

Gestió de les regles:

- ❏ Eliminar totes les regles existents (flush)
- ❏ Afegir regles al final
- ❏ Inserir regles a l'inici

- ❏ Llistar regles format llistat
- ❏ Llistar regles format iptables-save

- ❏ Inserir regles en una posició concreta.
- ❏ Eliminar regles concretes
- ❏ Check de l'existència d'una regla

- ❏ Crear chain definida per l'usuari
- ❏ Renombrar user-chain
- ❏ Eliminar user-chain

- ❏ Observar els comptadors de les regles/chains
- ❏ Posar a zero els comptadors de packets filtrats per regles/chains

Aplicació d'opcions generals:

- ❏ [!] -s soource
- ❏ [!] -d destination
- ❏ [!] -i input interface
- ❏ [!] -o output interface

- ❏ [!] -p protocol
- ❏ -m match
- ❏ -j jump

- ❏ -c set counters

iptables-extensions

extracte del man (8) iptables-extensions

MATCH EXTENSIONS

iptables can use extended packet matching modules with the -m or --match options, followed by the matching module name; after these, various extra command line options become available, depending on the specific module. You can specify multiple extended match modules in one line. The extended match modules are evaluated in the order they are specified in the rule.

addrtype

This module matches packets based on their address type. Address types are used within the kernel networking stack and categorize addresses into various groups. The exact definition of that group depends on the specific layer three protocol.

The following address types are possible:

UNSPEC an unspecified address (i.e. 0.0.0.0)
UNICAST an unicast address
LOCAL a local address
BROADCAST a broadcast address
ANYCAST an anycast packet
MULTICAST a multicast address
BLACKHOLE a blackhole address
UNREACHABLE an unreachable address
PROHIBIT a prohibited address
THROW
NAT
XRESOLVE

[!] --src-type type Matches if the source address is of given type

[!] --dst-type type Matches if the destination address is of given type

--limit-iface-in

The address type checking can be limited to the interface the packet is coming in. This option is only valid in the PREROUTING, INPUT and FORWARD chains. It cannot be specified with the --limit-iface-out option.

--limit-iface-out

The address type checking can be limited to the interface the packet is going out. This option is only valid in the POSTROUTING, OUTPUT and FORWARD chains. It cannot be specified with the --limit-iface-in option.

ah (IPv4-specific)

This module matches the SPIs in Authentication header of IPsec packets.

[!] --ahspi spi[:spi]

comment

Allows you to add comments (up to 256 characters) to any rule.

--comment comment
Example: iptables -A INPUT -i eth1 -m comment --comment "my local LAN"

esp

This module matches the SPIs in ESP header of IPsec packets.
[!] --espspi spi[:spi]

icmp (IPv4-specific)

This extension can be used if `--protocol icmp' is specified.
[!] --icmp-type {type[/code]|typename}
This allows specification of the ICMP type, which can be a numeric ICMP type, type/code pair, or one of the ICMP type names shown by the command iptables -p icmp -h

iprange

This matches on a given arbitrary range of IP addresses.
[!] --src-range from[-to]
Match source IP in the specified range.
[!] --dst-range from[-to]
Match destination IP in the specified range.

mac

[!] --mac-source address
Match source MAC address. It must be of the form XX:XX:XX:XX:XX:XX. Note that this only makes sense for packets coming from an Ethernet device and entering the PREROUTING, FORWARD or INPUT chains.

quota

Implements network quotas by decrementing a byte counter with each packet. The condition matches until the byte counter reaches zero. Behavior is reversed with negation (i.e. the condition does not match until the byte counter reaches zero).
[!] --quota bytes The quota in bytes.

state

The "state" extension is a subset of the "conntrack" module. "state" allows access to the connection tracking state for this packet.
[!] --state state
Where state is a comma separated list of the connection states to match. Only a subset of the states unterstood by "conntrack" are recognized: INVALID, **ESTABLISHED**, **NEW**, **RELATED** or **UNTRACKED**. For their description, see the "conntrack" heading in this manpage.

NEW The packet has started a new connection or otherwise associated with a connection which has not seenpackets in both directions.

ESTABLISHED The packet is associated with a connection which has seen packets in both directions.

RELATED The packet is starting a new connection, but is associated with an existing connection, such as an FTP data transfer or an ICMP error.

UNTRACKED The packet is not tracked at all, which happens if you explicitly untrack it by using -j CT --notrack in the raw table.

string

This modules matches a given string by using some pattern matching strategy. It requires a linux kernel >=2.6.14.

--algo {bm|kmp}
Select the pattern matching strategy. (bm = Boyer-Moore, kmp = Knuth-Pratt-Morris)
--from offset
Set the offset from which it starts looking for any matching. If not passed, default is 0.
--to offset
Set the offset up to which should be scanned. That is, byte offset-1 (counting from 0) is the last onethat is scanned. If not passed, default is the packet size.

[!] --string pattern Matches the given pattern.
[!] --hex-string pat tern Matches the given pattern in hex notation.

Examples:
The string pattern can be used for simple text characters.
iptables -A INPUT -p tcp --dport 80 -m string --algo bm --string 'GET /index.html' -j LOG
The hex string pattern can be used for non-printable characters, like |0D 0A| or |0D0A|.
iptables -p udp --dport 53 -m string --algo bm --from 40 --to 57 --hex-string '|03|www|09|netfilter|03|org|00|'

tcp

These extensions can be used if `--protocol tcp' is specified. It provides the following options:

[!] --source-port,--sport port[:port]
Source port or port range specification. This can either be a service name or a port number. An inclusive range can also be specified, using the format first:last. If the first port is omitted, "0" is assumed; if the last is omitted, "65535" is assumed. If the first port is greater than the second one they will be swapped. The flag --sport is a convenient alias for this option.

[!] --destination-port,--dport port[:port]
Destination port or port range specification. The flag --dport is a convenient alias for this option.

[!] --tcp-flags mask comp
Match when the TCP flags are as specified. The first argument mask is the flags which we should examine, written as a comma-separated list, and the second argument comp is a comma-separated list of flags which must be set. Flags are: SYN ACK FIN RST URG PSH ALL NONE. Hence the command iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST SYN will only match packets with the SYN flag set, and the ACK, FIN and RST flags unset.

[!] --syn
Only match TCP packets with the SYN bit set and the ACK,RST and FIN bits cleared. Such packets are used to request TCP connection initiation; for example, blocking such packets coming in an interface will prevent incoming TCP connections, but outgoing TCP connections will be unaffected. It is equivalent to --tcp-flags SYN,RST,ACK,FIN SYN. If the "!" flag precedes the "--syn", the sense of the option is inverted.

[!] --tcp-option number
Match if TCP option set.

tos

This module matches the 8-bit Type of Service field in the IPv4 header (i.e. including the "Precedence" bits) or the (also 8-bit) Priority field in the IPv6 header.

[!] --tos value[/mask]
Matches packets with the given TOS mark value. If a mask is specified, it is logically ANDed with the TOS mark before the comparison.

[!] --tos symbol
You can specify a symbolic name when using the tos match for IPv4. The list of recognized TOS names can be obtained by calling iptables with -m tos -h. Note that this implies a mask of 0x3F, i.e. all but the ECN bits.

ttl (IPv4-specific)

This module matches the time to live field in the IP header.
[!] --ttl-eq ttl Matches the given TTL value.
--ttl-gt ttl Matches if TTL is greater than the given TTL value.
--ttl-lt ttl Matches if TTL is less than the given TTL value.

udp

These extensions can be used if `--protocol udp' is specified. It provides the following options:
!] --source-port,--sport port[:port]

Source port or port range specification. See the description of the --source-port option of the TCP extension for details.

[!] --destination-port,--dport port[:port]
Destination port or port range specification. See the description of the --destination-port option of the TCP extension for details.

TARGET EXTENSIONS

iptables can use extended target modules: the following are included in the standard distribution.

DNAT

This target is only valid in the nat table, in the PREROUTING and OUTPUT chains, and user-defined chains which are only called from those chains. It specifies that the destination address of the packet should be modified (and all future packets in this connection will also be mangled), and rules should cease being examined. It takes the following options:

--to-destination [ipaddr[-ipaddr]][:port[-port]]
which can specify a single new destination IP address, an inclusive range of IP addresses. Optionally a port range, if the rule also specifies one of the following protocols: tcp, udp, dccp or sctp. If no port range is specified, then the destination port will never be modified. If no IP address is specified then only the destination port will be modified. In Kernels up to 2.6.10 you can add several --to-des- tination options. For those kernels, if you specify more than one destination address, either via an address range or multiple --to-destination options, a simple round-robin (one after another in cycle) load balancing takes place between these addresses. Later Kernels (>= 2.6.11-rc1) don't have the ability to NAT to multiple ranges anymore.

--random
If option --random is used then port mapping will be randomized (kernel >= 2.6.22).

--persistent
Gives a client the same source-/destination-address for each connection. This supersedes the SAME target. Support for persistent mappings is available from 2.6.29-rc2.

MASQUERADE

This target is only valid in the nat table, in the POSTROUTING chain. It should only be used with dynamicallyassigned IP (dialup) connections: if you have a static IP address, you should use the SNAT target. Masquerading is equivalent to specifying a mapping to the IP address of the interface the packet is going out, but also has the effect that connections are forgotten when the interface goes down. This is the

correct behavior when the next dialup is unlikely to have the same interface address (and hence any established connections are lost anyway).

--to-ports port[-port]

This specifies a range of source ports to use, overriding the default SNAT source port-selection heuristics (see above). This is only valid if the rule also specifies one of the following protocols: tcp, udp, dccp or sctp.

--random

Randomize source port mapping If option --random is used then port mapping will be randomized (kernel >= 2.6.21).

REDIRECT

This target is only valid in the nat table, in the PREROUTING and OUTPUT chains, and user-defined chains whichare only called from those chains. It redirects the packet to the machine itself by changing the destination IP to the primary address of the incoming interface (locally-generated packets are mapped to the localhost address, 127.0.0.1 for IPv4 and ::1 for IPv6).

--to-ports port[-port]

This specifies a destination port or range of ports to use: without this, the destination port is never altered. This is only valid if the rule also specifies one of the following protocols: tcp, udp, dccp or sctp.

--random

If option --random is used then port mapping will be randomized (kernel >= 2.6.22).

REJECT (IPv4-specific)

This is used to send back an error packet in response to the matched packet: otherwise it is equivalent to DROP so it is a terminating TARGET, ending rule traversal. This target is only valid in the INPUT, FORWARD and OUTPUT chains, and user-defined chains which are only called from those chains. The following option controls the nature of the error packet returned:

--reject-with type

The type given can be icmp-net-unreachable, icmp-host-unreachable, icmp-port-unreachable, icmp-proto-unreachable, icmp-net-prohibited, icmp-host-prohibited, or icmp-admin-prohibited (*), which return the appropriate ICMP error message (icmp-port-unreachable is the default). The option tcp-reset can be used on rules which only match the TCP protocol: this causes a TCP RST packet to be sent back. This is mainly useful for blocking ident (113/tcp) probes which frequently occur when sending mail to broken mail hosts (which won't accept your mail otherwise).

(*) Using icmp-admin-prohibited with kernels that do not support it will result in a plain DROP instead of REJECT

SNAT

This target is only valid in the nat table, in the POSTROUTING and INPUT chains, and user-defined chains which are only called from those chains. It specifies that the source address of the packet should be modified (and all future packets in this connection will also be mangled), and rules should cease being examined. It takes the following options:

--to-source [ipaddr[-ipaddr]][:port[-port]]

which can specify a single new source IP address, an inclusive range of IP addresses. Optionally a port range, if the rule also specifies one of the following protocols: tcp, udp, dccp or sctp. If no port range is specified, then source ports below 512 will be mapped to other ports below 512: those between 512 and 1023 inclusive will be mapped to ports below 1024, and other ports will be mapped to 1024 or above. Where possible, no port alteration will occur. In Kernels up to 2.6.10, you can add several --to-source options. For those kernels, if you specify more than one source address, either via an address range or multiple --to-source options, a simple round-robin (one after another in cycle) takes place between these addresses. Later Kernels (>= 2.6.11-rc1) don't have the ability to NAT to multiple ranges anymore.

--random

If option --random is used then port mapping will be randomized (kernel >= 2.6.21).

--persistent

Gives a client the same source-/destination-address for each connection. This supersedes the SAME target. Support for persistent mappings is available from 2.6.29-rc2.

TEE

The TEE target will clone a packet and redirect this clone to another machine on the local network segment. In other words, the nexthop must be the target, or you will have to configure the nexthop to forward it further if so desired.

--gateway ipaddr

Send the cloned packet to the host reachable at the given IP address. Use of 0.0.0.0 (for IPv4 packets) or :: (IPv6) is invalid.
To forward all incoming traffic on eth0 to an Network Layer logging box:
-t mangle -A PREROUTING -i eth0 -j TEE --gateway 2001:db8::1

TRACE

This target marks packets so that the kernel will log every rule which match the packets as those traverse the tables, chains, rules.

A logging backend, such as ip(6)t_LOG or nfnetlink_log, must be loaded for this to be visible. The packets are logged with the string prefix: "TRACE: tablename:chainname:type:rulenum " where type can be "rule" for plain rule, "return"

for implicit rule at the end of a user defined chain and "policy" for the policy of the built in chains. It can only be used in the raw table.

TTL (IPv4-specific)

This is used to modify the IPv4 TTL header field. The TTL field determines how many hops (routers) a packet can traverse until it's time to live is exceeded.

Setting or incrementing the TTL field can potentially be very dangerous, so it should be avoided at any cost.This target is only valid in mangle table.
Don't ever set or increment the value on packets that leave your local network!

--ttl-set value
Set the TTL value to `value'.
--ttl-dec value
Decrement the TTL value `value' times.
--ttl-inc value
Increment the TTL value `value' times.