Iptables

Taules i cadenes

Netfilter (https://www.netfilter.org) és el tallafocs integrat dins el nucli Linux. Internament està format per tres *taules*:

> **FILTER:** Filtra els paquets segons les regles especificades. El més utilitzat.
> **NAT:** Responsable del Network Address Translation, procediment molt usat per sistemes actuant com "routers"
> **MANGLE:** Modifica els bits de la capçalera TCP responsables de QoS. No la veurem.

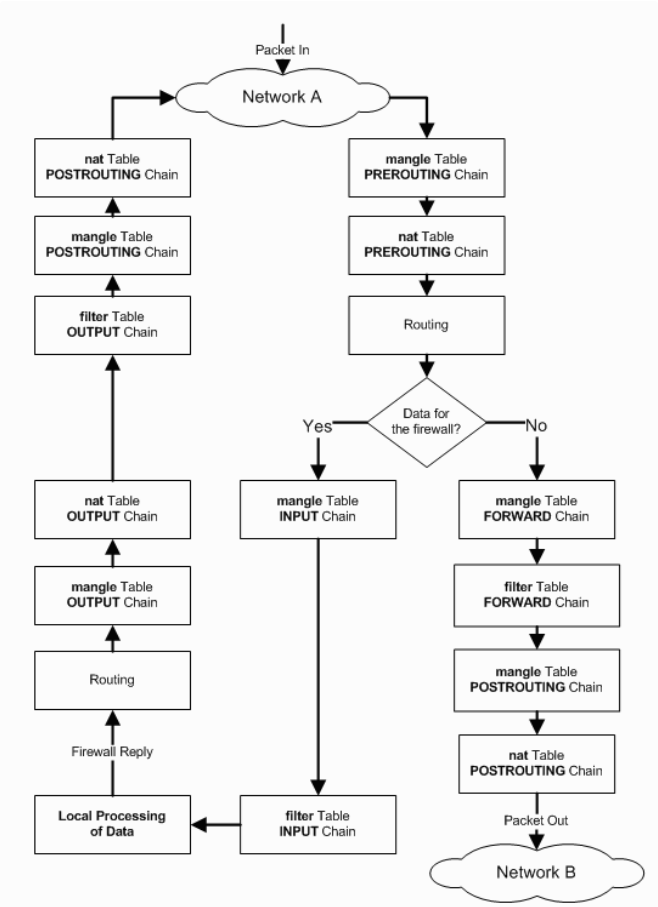
Dins la taula FILTER existeixen tres cadenes per defecte (es poden crear més si es vol). Una cadena és una “cua” per on passarà un paquet segons alguna característica que tingui. Són:

> **INPUT:** Paquets que entren i van destinats al sistema-tallafocs
> **OUTPUT:** Paquets que s'originen i surten del sistema-tallafocs
> **FORWARD:** Paquets que entren al sistema-tallafocs, però només de pas, amb destí altres sistemes

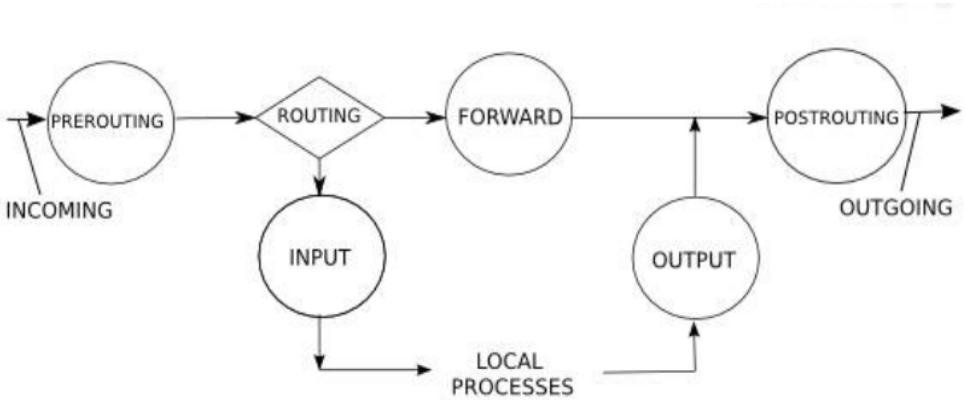
Dins la taula NAT les cadenes existents per defecte són:

> **PREROUTING:** Canvia l'adreça de destí del paquet (normalment s'usa en un paquet entrant a l'Iptables, on l'adreça original sol ser la IP "pública" -externa- de l'enrutador, que es canvia per l'adreça d'un node particular de la LAN). El nom d'aquesta cadena ve del fet de què es té en compte abans de què Iptables encamini el paquet on correspongui (o bé a la cadena INPUT o bé a la cadena FORWARD).
>
> **POSTROUTING:** Canvia l'adreça d'origen del paquet (normalment s'usa en un paquet sortint de l'Iptables, on l'adreça original sol ser l'adreça d'un node particular de la LAN, que es canvia per la IP "pública" -externa- de l'enrutador). El nom d'aquesta cadena ve del fet de què es té en compte després de que el paquet hagi sigut encaminat "cap a fora" per on correspongui (pot ser a través de la cadena OUTPUT o de FORWARD).

El següent esquema ho exposa gràficament:



Un esquema molt més senzill (i simplificat) seria aquest:



Un altre esquema molt més complet es pot trobar a https://www.garron.me/images/2012-04/Netfilter-packet-flow.svg
També és il.lustratiu l'esquema http://linux-ip.net/nf/nfk-traversal.png

Accions

Un cop el paquet ja està "encuat" a la cadena adequada,¿què es pot fer amb ell?. Les accions que es poden realitzar amb cada paquet són:

*A les totes cadenes de la taula FILTER:

ACCEPT : El paquet s'envia amb èxit a l'aplicació destí, i s'atura el seu processament
DROP : El paquet es rebutja (sense notificar-ho al remitent), i s'atura el seu processament
REJECT : Igual que DROP, però envia un missatge RST/ACK al remitent informant del rebuig.
NOTA: Amb el paràmetre *--reject-with* de la comanda Iptables es pot especificar el motiu a la màquina remota, motiu que hauria de ser algun dels següents: *icmp-port-unreachable* (default), *icmp-net-unreachable*, *icmp-host-unreachable*, *icmp-net-prohibited*, *icmp-proto-unreachable*, *icmp-host-prohibited*, *tcp-reset* o *echo-reply*

*A la cadena PREROUTING de la taula NAT:

DNAT : Fa el canvi efectiu al paquet de la IP (i port) de destí. Cal indicar *--to-destination ipdestifinal:[port]*
REDIRECT : Només canvia el port de destí. Cal indicar *--to-port nºport*

*A la cadena POSTROUTING de la taula NAT:

SNAT : Fa el canvi efectiu al paquet de la IP (i port) d'origen. Cal indicar *--to-source iporigen[:port]*
MASQUERADE : Igual que SNAT però canvia automàticament la IP de l'origen per la IP de la tarjeta de sortida sense que calgui especificar-la "a mà" amb el paràmetre *--to-source*, Si es volgués canviar el port d'origen caldria llavors usar específicament el paràmetre *–to-port nºport*

*A totes les cadenes de totes les taules:

LOG : El paquet s'envia al registre del sistema (si aquest és de tipus Systemd -com són la majoria de sistemes moderns, entre ells Ubuntu o Fedora-, aquest registre es pot gestionar mitjançant la comanda journalctl; concretament, un exemple per veure tots els missatges des del darrer inici de la màquina podria ser *journalctl -k | grep "IN=.*OUT=.*" | less*) i es continua el seu processament sense alteracions, encara que, normalment just després d'una acció LOG ve una acció DROP del mateix paquet.
NOTA: En la comanda Iptables s'usa amb el paràmetre *--log-prefix "frase"* per afegir al registre la informació que considerem rellevant

Les regles que diuen quins paquets (de quines cadenes i amb quines característiques concretes) han de ser rebutjats i quins no, s'han d'escriue en ordre. És molt important això perquè en el moment que un paquet coincideix amb la descripció que fa d'ell una regla, s'executa l'acció i ja no es continua investigant cap més regla que hi hagi per sota (excepte amb l'acció LOG). Si un paquet no coincideix amb cap regla existent, després d'haver passat per totes, se li aplicarà la regla per defecte corresponent a la cadena on estigui encuat.

A partir de l'explicat al paràgraf anterior, és evident que quan més específica sigui una regla, menys paquets coincidiran amb la descripció i per tant a menys paquets afectarà. És per això que el més freqüent és indicar primer les regles que solen tenir més coincidències i deixar per després les regles més genèriques.

Comanda Iptables

La comanda de terminal que ens permet gestionar totes les regles és Iptables. Els seus paràmetres més importants són:

-L [cadena] [-t taula]: Mostra les regles actuals d'una cadena concreta de la taula indicada (si no s'indica cap taula, per defecte és FILTER; si no s'indica cap cadena, per defecte són totes les de la taula FILTER). Si s'afegeix **-n** es veurà en format numèric. Amb **-v** es mostren més columnes (mode verbós). Amb **–line-numbers** es mostra per cada regla la seva posició dins de la cadena (això és interessant a l'hora de saber on afegir properes regles). Amb **-x** es veuen les quantitats de bytes exactes (en comptes d'arrodonir a MB,GB, etc)

-F [cadena] [-t taula]: Esborra totes les regles de la cadena especificada (x def. de totes les de taula FILTER)
-Z [cadena] [-t taula]:Reinicia els comptadors de paquets i bytes (x def. de totes les cadenes de taula FILTER)

-P cadena [-t taula] acció : Estableix l'acció per defecte per la cadena especificada (és a dir, l'acció que s'aplicarà a un paquet si aquest no coincideix amb cap regla). Si la taula no és FILTER, s'ha d'especificar.

NOTA: Pel tràfic d'entrada el més habitual (perquè és més segur) és aplicar una acció per defecte de denegació (DROP o REJECT) i llavors només obrir els ports estrictament necessaris un a un amb regles específiques. Pel tràfic de sortida no se sol fer res especial (normalment s'executa ACCEPT per defecte).

-A cadena [-t taula]: Afegeix una regla al final de la cadena indicar. Si la taula no és FILTER, s'ha d'especificar. Aquesta opció ha de venir acompanyada d'una sèrie de paràmetres que serviran per identificar al paquet processat i així decidir la regla en qüestió l'afecta (o no, continuant llavors a la regla següent). Els paràmetres més importants (després en veurem més) són:

-p {tcp|udp|icmp|all} : El paquet és del protocol especificat.
NOTA: En el cas del protocol ICMP, es pot concretar més el tipus de paquet ICMP a detectar amb el paràmetre **--icmp-type tipus** (els tipus més habituals són echo-request i echo-reply).
NOTA: En el cas del protocol TCP es pot concretar més el tipus de paquet TCP a detectar indicant amb el paràmetre **--tcp-flags** el valors concrets dels flags que volem per a què hi hagi coincidència amb els del paquet examinat. La manera de fer servir aquest paràmetre és indicant primer la llista de flags a examinar (separats per comes; els més habituals són SYN ACK FIN, RST, ALL i NONE) i seguidament, després d'un espai en blanc, tornant a indicar la llista de flags però només dels que han de valer "1"). Un exemple: --tcp-flags SYN,ACK,FIN,RST SYN indica que el paquet ha de tenir el flag SYN a 1 i ACK, FIN i RST a 0

-s ip/mask : Màquina/xarxa remitent del paquet. La IP pot ser de host o de xarxa. 0/0 significa “any”.
-d ip/mask : Màquina/xarxa destí del paquet. La IP pot ser de host o de xarxa. 0/0 vol dir "any".

--sport nº: Port origen del paquet. Pot ser un rang si s'especifica nºportinicial:nºportfinal
--dport nº : Port destinatari del paquet. Pot ser un rang si s'especifica nºportinicial:nºportfinal

-i ethX : Tarja per on els paquets entraran
-o ethX : Tarja per on els paquets sortiran

-j acció :: Executa l'acció especificada si el paquet coincideix amb les característiques descrites

NOTA: Si s'indica un símbol "!" davant dels paràmetres anteriors (és a dir, **! -p, ! -s, ! -d**, etc) es vol dir que el paquet pot ser qualsevol excepte l'indicat pel paràmetre. Per exemple, si s'escriu **! -p tcp** s'estarà dient que el paquet pot ser de qualsevol protocol excepte tcp; si s'escriu **! -d 192.168.1.0/24** s'estarà dient que el paquet pot anar a qualsevol destí excepte la xarxa indicada, etc, etc.

-I cadena [noposicio] [-t taula] : Insereix una regla dins la cadena indicada a la posició indicada. Si no s'indica la posició, per defecte és al començament de tot (posició nº1). Si la taula no és FILTER, s'ha d'especificar

-R cadena noposicionova [-t taula] : Canvia de posició una regla ja existent dins la cadena indicada. Si la taula no és FILTER, s'ha d'especificar

-D cadena regla_exacta_a_borrar [-t taula]: Esborra la regla indicada a la cadena indicada. També es pot escriure -D cadena no,on no és la posició (dins de la cadena en qüestió) de la regla a eliminar. Si la taula no és FILTER, s'ha d'especificar

-C cadena regla_exacta_a_trobar [-t taula]: Comprova si la regla indicada ja existeix a la cadena indicada. Si la taula no és FILTER, s'ha d'especificar

-N nomcadena [-t taula]: Crea una nova cadena pròpia. Si la taula no és FILTER, s'ha de dir. Veurem exemples més endavant.
-X nomcadena [-t taula]: Esborra una cadena pròpia creada anteriorment. Si la taula no és FILTER, s'ha de dir
-E nomcadenaantic nomcadenanou [-t taula]: Renombra una cadena pròpia. Si la taula no és FILTER, " " "

Exemples bàsics

Se suposarà en els exemples que el sistema on està funcionant Iptables té dues tarjes: "eth0" i "eth1". La primera tindrà la IP 192.168.1.1 i és per on escoltaran els diferents servidors que tinguem configurats (Ssh, Http, Dhcp, etc) a la nostra LAN 192.168.1.0/24; la segona tindrà la IP 10.0.1.1 i servirà per connectar el sistema amb "l'exterior".

*Bloquejar tot el tràfic que entri per la tarja de xarxa eth0:

iptables -A INPUT -i eth0 -j DROP

*Bloquejar (només) el tràfic que entri per la tarja de xarxa eth0 i que a més provingui d'una IP concreta (també es podria bloquejar a la xarxa sencera especificant 192.168.1.0/24):

iptables -A INPUT -i eth0 -s 192.168.1.234 -j DROP

*Bloquejar (només) el tràfic que entri per la tarja de xarxa eth0 (provinent de qualsevol lloc) i que a més vagi dirigit a un port determinat del nostre sistema (en aquest cas, el port 22 TCP):

iptables -A INPUT -i eth0 -p tcp --dport 22 -j DROP

*Bloquejar (només) el tràfic que entri per la tarja de xarxa eth0 (provinent de qualsevol lloc) i que a més vagi dirigit a un port determinat del nostre sistema (en aquest cas, el port 22 TCP) però, a més, guardar un registre sobre aquest fet:

iptables -A INPUT -i eth0 -p tcp --dport 22 -j LOG --log-prefix "Acceso al puerto 22 detectado"
iptables -A INPUT -i eth0 -p tcp --dport 22 -j DROP

*Bloquejar (només) el tràfic que entri per la tarja de xarxa eth0 que a més provingui d'una IP concreta i que a més vagi dirigit a un port determinat del nostre sistema (en aquest cas, el port 22 TCP):

iptables -A INPUT -i eth0 -s 192.168.1.234 -p tcp --dport 22 -j DROP

Mòduls d'Iptables

L'Iptables incorpora una sèrie de mòduls interns que amplien les seves capacitats de detecció de paquets (cadascun de maneres diferents) més enllà de les capacitats "per defecte". Per fer-los servir en una regla cal afegir el paràmetre **-m nommòdul** i, en general, algun altre paràmetre més específic del mòdul escollit. A continuació es llisten els exemples d'ús més habituals (teniu més informació a la pàgina del manual anomenada "iptables-extensions"):

-m mac --mac-source xx:yy:zz:ww:vv:uu : Per detectar la direcció MAC de la màquina origen del paquet. Només funciona a les cadenes INPUT, FORWARD i PREROUTING

-m multiports --dports nº,nº,nº :Per poder especificar més d'un port de destí en una sola regla
-m multiports --sports nº,nº,nº :Per poder especificar més d'un port d'origen en una sola regla

-m conntrack --ctstate NEW, ESTABLISHED : Per especificar si el paquet es correspon a l'inici d'una connexió o forma part del tràfic pertanyent a una connexió ja establerta. Altres valors possibles són RELATED (quan el paquet es correspon a l'inici d'una conexió però que està relacionada amb una conexió anterior; això és habitual en transferències FTP o errors ICMP) o INVALID (el paquet no està associat a cap conexió coneguda).

Per exemple, si una màquina tingués TOT el tràfic entrant i sortint tancat (és a dir, que tingués el valor DROP establert per defecte a les cadenes INPUT i OUTPUT i cap regla més definida), per aconseguir que un servidor SSH instal.lat en aquella màquina pogui comunicar-se normalment amb els client, hauríem d'escriure les següents dues regles:
iptables -A INPUT -i eth0 -p tcp --dport 22 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o eth0 -p tcp --sport 22 -m conntrack --ctstate ESTABLISHED -j ACCEPT

A continuació es mostra un exemple d'ús del mòdul conntrack (concretament,vd'un tallafocs en un sistema funcionant com servidor HTTP (i SSH només per un client determinat). És molt important fixar-se en l'ordre de les regles! Recordem que la primera regla que coincideixi amb les característiques del paquet processat és la que s'aplicarà i ja no es continuarà mirant altres regles!

NOTA: Aquests exemples es poden escriure en un shell script per tal de provar-los instantàniament

iptables -F INPUT
iptables -A INPUT -m conntrack --ctstate ESTABLISHED -j ACCEPT
iptables -A INPUT -m conntrack --ctstate NEW -i lo -j ACCEPT
iptables -A INPUT -m conntrack --ctstate NEW -i eth0 -p tcp –dport 80 -j ACCEPT
iptables -A INPUT -m conntrack --ctstate NEW -i eth0 -p tcp –dport 22 -s 192.168.1.234 -j ACCEPT
#Accepto peticions ping de fora i també respostes ping de fora
iptables -A INPUT -m conntrack --ctstate NEW -i eth0 -p icmp –icmp-type echo-request -j ACCEPT
iptables -A INPUT -m conntrack --ctstate NEW -i eth0 -p icmp –icmp-type echo-reply -j ACCEPT
iptables -P INPUT DROP
iptables -F FORWARD
iptables -P FORWARD DROP
iptables -F OUTPUT
iptables -P OUTPUT ACCEPT

En general, el que s'hauria de procurar aconseguir amb les regles que s'escriguin és:

*Permetre tot el tràfic a la interfície Loopback (-A INPUT -i lo -j ACCEPT i -A OUTPUT -o lo -j ACCEPT)
*Permetre en general tots els paquets sortints (-P OUTPUT ACCEPT) i denegar els entrants (-P INPUT DROP)
*Permetre els paquets entrants només quan formin part de connexions ja establertes o de connexions noves que estiguin relacionades amb prèviament existents (-A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT) o quan pertanyin a una connexió nova només si van dirigits a un port adient

NAT (teoria)

NAT és un mecanisme que permet compartir una direcció IP pública per molts equips que tenen cadascú una IP privada diferent. D'aquesta manera, es permet l'accés a "Internet" a tots aquests equips minimitzant el problema de l'escassetat de direccions IPv4 (públiques). Hi ha bàsicament dos tipus de NAT:

*"**Source NAT**": El que fa aquest NAT és canviar la IP d'origen del paquet (que sol ser la IP privada d'un ordinador de la nostra LAN) per la IP pública de la porta d'enllaç ("router"). Això vol dir que el destí del paquet "veurà" que l'origen del paquet és el "router" i, per tant, serà a ell a qui li reenviarà la resposta (més en concret, a un port determinat del "router"). Quan aquesta resposta arribi al "router", aquest consultarà una taula que tindrà en memòria on s'associa la IP d'origen real (i també el port d'origen) amb la seva IP (i el port que ha utilitzat el router per connectar amb l'exterior) i reenviarà convenientment aquesta resposta a l'origen real.

Fixeu-vos que a la taula hi ha quatre dades: IP origen, port origen, IP router, port router. D'aquesta manera, el router és capaç de reenviar una resposta rebuda per un determinat port seu a un determinat port de l'origen. Per tant, és perfectament possible que cada origen pugui tenir diferents connexions en paral.lel (cadascuna creades en un port diferent) perquè el router saps quina és quina en tot moment. Normalment, el port que fa servir el router per accedir a l'exterior és el mateix que el port utilitzat per l'origen, però si hi hagués més d'un origen utilitzant el mateix port, el router llavors automàticament en farà servir un altre (i,per tant, també canviarà al paquet el port d'origen).

Resumint,

1.-Un equip d'una LAN amb IP privada (suposarem 192.168.3.14) vol accedir a una pàgina web (port 80 TCP) com www.sindominio.net. Realitza la consulta DNS i obté que l'equip que allotja la pàgina té la direcció 76.74.254.126

2.-Consulta la seva taula d'encaminament i com no l'equip destí no està a la seva xarxa, envia la petició de pàgina a la seva porta d'enllaç definida (que suposarem que té la IP 192.168.3.254)

3.-El gateway, en comprovar que la direcció IP de destí no és la seva, l'enviarà a la seva pròpia porta d'enllaç, que ja serà una direcció IP pública. Abans de què el paquet surti per la seva tarja de xarxa externa, però, el gateway canvia la direcció IP d'origen (192.168.3.14) per la seva direcció IP pública (que suposarem que és 80.58.1.14) i es guarda la petició en una taula en memòria (anotant també el port d'origen, que suposarem que és el 5015 TCP).

4.-El paquet viatja per Internet saltant de router a router fins que arriba al seu destí

5.-L'equip 76.74.254.126 rep la petició des de la direcció 80.58.1.14 y la respón. Per tant, el paquet de resposta tindrà com direcció IP d'origen 76.74.254.126, direcció IP de destí 80.58.1.14, port d'origen 80 TCP i port de destí 5015 TCP

6.-Aquesta resposta arriba a la tarja externa del gateway de la nostra LAN, que consulta la seva taula en memòria i comprova (gràcies al port origen allà registrat) que correspon amb una petició realitzada des de l'equip 192.168.3.14, així que modifica la direcció IP de destí del paquet per aquesta IP i l'envia directament.

Hi ha un tipus de "Source NAT" específic anomenat "**IP masquerading**", el qual és el que passa quan la direcció IP pública que sustitueix a la IP d'origen és dinàmica (el cas més habitual en connexions a Internet domèstiques). En el "Source NAT" clàssic la IP pública és estàtica.

*"**Destination NAT"**: El que fa aquest NAT és canviar la IP de destí del paquet (que sol ser la IP pública de la tarja externa del nostre gateway) per una IP privada d'alguna màquina de la nostra LAN. Aquest tipus de NAT es realitza només quan el gateway detecta que aquest paquet no es correspon amb cap connexió iniciada des de la nostra LAN (és a dir, ha de ser un equip extern qui inicii la connexió) i té prèviament definida la regla DNAT adequada (si no la té, el paquet es descarta). Aquest tipus de NAT s'utilitza quan tenim algun servidor funcionant en una màquina de la nostra LAN i volem que sigui accessible des de l'exterior. Els passos concrets són aquests:

1.-Un equip qualsevol d'Internet amb direcció IP pública 150.212.23.6 vol connectar-se per SSH (port 22 TCP) a l'equip "miservidor.midominio.com". Realitza una consulta DNS i obté com a resposta que aquest equip té la direcció IP 85.136.14.7

2.-Estableix la connexió (suposem que el port d'origen que utilitza és 23014 TCP) amb l'equip 85.136.14.7, que resulta ser un dispositiu NAT que no té cap servei SSH escoltant al port 22 TCP però que té una regla DNAT que fa que tot el que li arribi a aquest port ho reenviarà a un equip de la seva xarxa local (suposarem que és el 10.0.0.2). Per tant, canvia la direcció IP de destí (85.136.14.7) per la 10.0.0.2 i registra aquesta associació a una taula en memòria

3.-A l'equip 10.0.0.2 li arriba una sol.licitud al port 22 TCP i la resposta del servidor SSH tindrà les següents característiques: IP d'origen 10.0.0.2, port d'origen 22 TCP, IP de destí 150.212.23.6 i port de destí 23014 TCP

4.-El dispositiu NAT canvia la direcció IP d'origen per la seva direcció IP pública (85.136.14.7) i el paquet de resposta arriba al seu destí.

Aquest tipus de NAT se sol anomenat "Port forwarding". No confondre amb el PAT ("Port Address Translation"), el qual és un mecanisme que només canvia els ports (d'origen o de destí, depèn) però no les Ips. El **PAT** és útil quan un servidor no escolta al seu port estàndard però no volem fer que els client hagin d'especificar manualment el port no estàndard al que han d'anar quan connectin a aquest servidor.

NAT (pràctica)

Si tenim per exemple un servidor HTTP escoltant al port 1234 en comptes del port per defecte (80), podem fer que totes les peticions al nostre port 80 es redireccionin automàticament al port 1234 sense que els clients (en aquest cas, els navegadors) tinguin constància. És a dir, fer un PAT. Això es fa amb la regla:

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT --to-port 1234

Si la màquina on tenim el tallafocs funcionant (que és la que rep les peticions dels navegador) no fos la mateixa que la que corre el servidor HTTP (és a dir, si el tallafocs ofereix una protecció al davant del servidor web, el qual funciona al

"darrera" -en una màquina amb IP 192.168.1.4, per exemple-), es podria fer igualment una redirecció de màquina (+ port). És a dir, fer un DNAT, així:

iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j DNAT --to-destination 192.168.1.4:1234

El més habitual, de totes formes, és utilitzar la taula NAT en un sistema que tingui dues tarjes, per tal de funcionar com a "router", on es rep el tràfic per una tarja i es redirigeix a l'altra, que és per on surt. No obstant, per a què aquesta funcionalitat es pugui fer servir, primer hem d'activar el "IP forwarding" entre ambdues tarjes. Això es fa simplement escrivint el valor 1 a l'arxiu /proc/sys/net/ipv4/ip_forward (així, per exemple: **echo 1 > /proc/sys/net/ipv4/ip_forward**

> NOTA: Una altra opció, en comptes d'editar manualment l'arxiu /proc/sys... seria utilitzar la comanda sysctl. En el cas concret de voler manipular el valor "ip_forward", primer podem mirar el seu valor actual amb *sysctl net.ipv4.ip_forward* i seguidament, per canviar-lo de forma permanent, amb *sysctl -w net.ipv4.ip_forward=1*

No obstant, aquest arxiu és temporal (és a dir, el seu valor només es manté mentre la màquina estigui encesa). Si volem que l'IP forwarding" sigui permanent, haurem de crear/modificar l'arxiu "origen" d'aquesta configuració, el qual està dins la carpeta /etc/sysctl.d (o també pot estar dins de /usr/lib/sysctl.d) El nom de l'arxiu en qüestió sol ser "99-sysctl.conf". En aquest arxiu, doncs, hem d'assignar a la línia net.ipv4.ip_forward el valor 1

> NOTA: Altres valors interessants dins l'arxiu "99-sysctl.conf" són net.ipv4.icmp_echo_ignore_all (si el seu valor és 1, s'ignoren totes les peticions de resposta a un ping) o net.ipv4.icmp_echo_ignore_broadcasts (si el seu valor és 1, s'ignoren peticions broadcast), entre altres.

A continuació es presenta un script d'exemple per convertir un sistema Iptables en "router" que fa un "IP masquerading" (el tipus de SNAT més habitual), on la seva tarja eth0 està conectada a la LAN interna i eth1 a "Internet". Els ordinadors de la LAN hauran de tenir com a porta d'enllaç per defecte la direcció IP de eth0:

```
iptables -t nat -F
iptables -F
iptables -t nat -X
iptables -X
#No permeto cap tràfic amb destí al "router" específicament (excepte un parell d'excepcions "tècniques"...
iptables -P INPUT DROP
iptables -A INPUT -i lo -j ACCEPT
iptables -A INPUT -i eth1 -p icmp -j ACCEPT
#...i el tràfic de resposta originat a partir d'alguna conexió prèvia feta des de la LAN interna)
iptables -A INPUT -p tcp -m conntrack --ctstate RELATED, ESTABLISHED -j ACCEPT
#No permeto cap tràfic originat des del "router" específicament
iptables -P OUTPUT DROP
#Activo el IP forwarding però aplico una política per defecte restrictiva
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -P FORWARD DROP
#Sí permeto el tràfic que va de eth0 (dins) a eth1 (fora)...
iptables -A FORWARD -i eth0 -o eth1 -j ACCEPT
#...i el que va d'eth1 (fora) a eth0 (dins) però només si està relacionat amb tràfic anterior
iptables -A FORWARD -i eth1 -o eth0 -m conntrack –ctstate RELATED,ESTABLISHED -j ACCEPT
#A l'hora de sortir a Internet, faig "IP masquerading"
#(és a dir, substitueixo la IP de l'origen ubicat a la LAN per la IP -pública- del "router")
iptables -t nat -A POSTROUTING -i eth0 -o eth1 -s 192.168.1.0/24 -j MASQUERADE
```

Persistència de les regles Iptables

A Ubuntu

Iptables a Ubuntu es pot gestionar com un servei qualsevol (mitjançant *systemctl{start|stop|status} iptables)* sempre i quan s'hagi instal.lat prèviament el paquet "iptables-persistent". Durant la instal.lació es preguntarà si les regles actualment en memòria es volen guardar (al fitxer /etc/iptables/rules.v4). Un cop guardades, es poden editar "a mà" si es desitja, respectant la sintaxi interna del fitxer. Si es vol comprovar que les modificacions que haguem introduït estiguin ben escrites, podem executar la comanda *iptables-restore -t /etc/iptables/rules.v4* en busca d'errors.

Per recarregar a memòria les regles guardades al fitxer anterior es pot fer servir (com a root) la comanda *netfilter-persistent reload*. Per tornar a gravar al fitxer les possibles noves regles actuals que haguem definit, es pot fer servir (com a root) la comanda *netfilter-persistent save* . El fitxer de configuració d'aquesta comanda és /etc/default/netfilter-persistent.

A Fedora

Iptables a Fedora es podia gestionar tradicionalment com un servei qualsevol (mitjançant *systemctl{start|stop|status} iptables*) sempre i quan s'hagi instal.lat prèviament el paquet "iptables-services", d'una forma similar a com passa a Ubuntu (en aquest cas, el fitxer de regles és /etc/sysconfig/iptables). No obstant, des de fa unes poques versions s'ha introduït un nou servei del sistema anomenat firewalld (systemctl status firewalld, etc), el qual funciona com una capa per sobre d'Iptables i és l'encarregat de gestionar la persistència de les regles (les quals es poden definir dins de fitxers específics del propi Firewalld sense haver de recòrrer a Iptables, que roman com eina de més baix nivell). Per saber més sobre Firewalld, llegiu https://www.certdepot.net/rhel7-get-started-firewalld o https://fedoraproject.org/wiki/FirewallD .