4. FIREWALL SI NAT

4.1. Concepte

4.1.1. Ce este un firewall: definitie, roluri, solutii alternative

Un firewall reprezinta un separator intre doua domenii de securitate, adesea intre „reteaua noastra” si „restul lumii”. El are rolul de control al accesului in ambele sensuri, prin filtrarea de pachete sau alte procedee precum NAT sau proxy, astfel incat sa permita trecerea numai pentru traficul autorizat. Un firewall poate fi un dispozitiv hardware sau o aplicatie software.

Iata cateva roluri posibile ale unui firewall:

- cel mai adesea este utilizat pentru a permite in mod securizat accesul utilizatorilor autorizati catre Internet
- separarea retelei publice a unei firme de reteaua privata din interior
- separarea in interiorul retelei private a unei companii a anumitor hosturi de toate celelalte dupa varii criterii (ex: se doreste filtrarea accesului la PC-urile celor de la dpt. de marketing sau la cele aflate in laboratorul de testare a noilor pachete software, etc.)
- monitorizarea/logging-ul traficului care ajunge la firewall.

Printre modalitatile de a crea un separator de securitate se numara:

1. **Filtrarea pachetelor** de date la un nivel cat mai scazut in stiva de protocoale de retea – in general la nivelul 2 (legatura de date) sau 3 (internet) unde pachetele pot fi admise/respinse pe baza adresei sursa/destinatie, portului etc.
2. **NAT** (Network Address Translation) – un procedeu prin care un numar de calculatoare sunt „ascunse” in spatele unor alte adrese decat cele reale. Desi nu poate fi gandit ca un firewall in sensul strict, NAT realizeaza totusi o separare din punct de vedere al securitatii unei retele.
3. **Proxy** – o aplicatie care se interpune intre clientul si serverul unui anumit protocol, acceptand cereri din partea clientului si trimitandu-le mai departe catre server. Spre deosebire de filtrele de pachete si NAT, proxy-ul este constient de continutul datelor ce-l tranziteaza, deoarece “intelege” protocoale de nivel mai inalt – in general de nivel aplicatie – ceea ce permite o mai riguroasa filtrare a continutului. In acelasi timp insa, un proxy dezavantajul vitezei mai scazute decat primele doua.

Pentru primele doua variante, sub Linux este prezenta, incepand cu kernelurile 2.4, o infrastructura de retea ce permite realizarea de filtrari complexe si care poarta numele de „arhitectura netfilter”.

4.1.2. Tipuri de firewall-uri

Firewall-urile pot fi clasificate din diferite puncte de vedere:

- in functie de felul in care trateaza pachetele – independent sau apartinatoare de un anumit dialog intre doua statii – avem:
  - firewall **stateless** – trateaza fiecare pachet in mod independent de celelalte, fie ele si facand parte din aceeasi conexiune/sesiune. Realizeaza filtrarea pachetelor strict in functie de informatiile aflate in pachetul curent (IP sursa/destinatie, port sursa/destinatie, tipul protocolului (TCP/UDP) etc.)
  - firewall **stateful** – tine cont de originea cererii, de eventuala apartenenta a pachetului la o conexiune (este o conexiune noua, deja existenta, noua dar in relatie cu una deja existenta, fara nicio legatura?). In cadrul unei conexiuni TCP firewall-ul va urmari ca succesiunea pachetelor sa fie cea corecta conform protocolului folosit (ex: o conexiune TCP debuteaza cu pachete ce au flag-ul SYN setat, iar

transferul de date se efectueaza cu flag-ul ACK setat; orice abatere poate fi "sanctionata" de catre firewall prin oprirea pachetelor neconforme)

- in functie de nivelul OSI la care se face filtrarea, firewalurile pot actiona la:
  - **layer 2** (nivelul legatura de date) – astfel de firewall-uri pot realiza filtrarea pachetelor in functie de adresele MAC sursa/destinatie
  - **layer 3** (nivelul retea) – firewall-urile de acest fel pot filtra pachetele in functie de adresa IP sursa/destinatie, TTL, TOS si alte campuri prezente in headerul IP
  - **layer 4** (nivelul transport) – aceste firewall-uri pot filtra dupa protocolul de trasnport folosit (TCP, UDP) si dupa portul sursa/destinatie
  - **layer 5** (nivelul sesiune) – firewall-ul va filtra pachetele in functie de parametri de nivel mai inalt (este o sesiune noua? A fost deschisa din reteaua protejata sau din Inet?, etc)
  - **layer 7** (nivelul aplicatie) – firewall-ul poate urmari cine a generat cererea sau carei aplicatii ii este destinat pachetul, si poate efectua filtrarea dupa parametri ai protocolului de nivel aplicatie folosit (ex: interzicerea anumitor tipuri de cereri HTTP, FTP etc)

In general, un soft de firewall va combina cele de mai sus, oferind posibilitati de filtrare la layere multiple.

- in functie de modalitatea de implementare:
  - **firewall-uri hardware** - sunt dispozitive de retea dedicate (ex. Cisco PIX - Private Internet Exchange Firewall) ce au rolul filtrarii traficului ce le tranziteaza. O buna parte a functiilor oferite sunt implementate hardware si foarte bine optimizate, ceea ce ofera performante net superioare solutiilor cu implementare exclusiv software
  - **firewall-uri software** – sunt aplicatii ce realizeaza functii de filtrare de pachete. Se intalnesc de obicei ca parte a unui sistem de operare (Windows, Linux, Unix etc) si sunt deseori gandite sa protejeze statia pe care ruleaza (ex: ZoneAlarm, Sygate Personal Firewall Pro pentru Windows, pf pentru OpenBSD, iptables pentru majoritatea distributiilor de Linux, s.a.).

4.2. Firewall in Linux

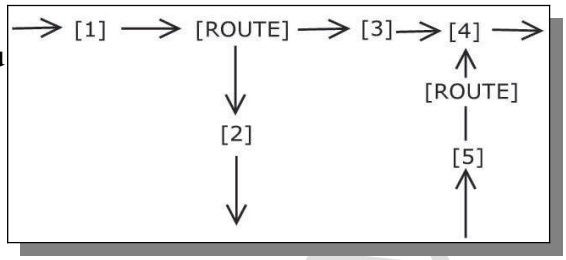
4.2.1. Arhitectura Netfilter

Arhitectura Netfilter reprezinta o infrastructura ce permite accesul la pachetele de date in drumul lor prin stiva de retea din kernel. Un pachet, odata sosit pe o statie, este preluat si plasat in memorie, urmand sa i se aplice diferite operatii ce tin de partea de retea (ex: verificarea integritatii, routare etc).

Nota: *cand vorbim de "drumul" unui pachet prin kernel, ne referim la aceasta succesiune de operatii ce i se aplica; pachetul nu se deplaseaza in memorie, deoarece aceasta este o operatie costisitoare si in general inutila.*

Un pachet receptionat de kernel contine:

- informatia pachetului
  - **header-e** – informatiile de control ale diverselor protocoale incapsulate in pachet, in functie de care se realizeaza in cele mai dese cazuri filtrarea pachetului
  - **payload** – informatia utila prezenta in pachet (cea "carata" de catre pachetul respectiv)
- **meta-informatia** atasata pachetului – asa cum, pentru fiecare fisier din sistemul de fisiere, exista un inode ce contine informatiile despre acel fisier, in acelasi fel kernelul Linux mentine informatii despre fiecare pachet primit (ex: momentul receptionarii, diferiti parametri si flaguri etc)



Deciziile pe care firewall-ul le ia asupra unui pachet pot fi bazate pe informatii prezente in oricare dintre sectiunile de mai sus. Pentru a putea lua decizii asupra pachetelor este necesara prezenta unor puncte de interactiune in care firewall-ul sa aiba acces la pachete in drumul lor prin kernel. Arhitectura netfilter defineste 5 astfel de puncte; in fiecare din acestea, asupra fiecarui pachet se pot lua decizii privind:

- accesul pachetului prin acel punct de interactiune
  - acceptarea pachetului – pachetului i se va permite trecerea prin acel punct de interactiune
  - distrugerea pachetului – firewall-ul va distruge pachetul fara a instiinta expeditorul
  - respingerea pachetului – firewall-ul va distruge pachetul, trimitand insa un mesaj de eroare catre expeditor
- modificarea pachetului – fie a informatiilor din pachet (ex: TTL, TOS, etc.) , fie a meta-informatiei atasate. In aceasta categorie se incadreaza si NAT-ul

4.2.2. IP Tables

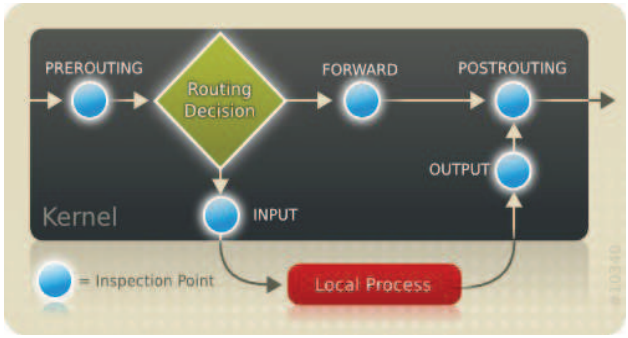
4.2.2.1. Module si tabele

IP Tables reprezinta un sistem software construit peste arhitectura netfilter si care permite filtrarea si modificarea pachetelor in functie de informatia continuta sau de meta-informatia atasata. In fiecare dintre cele 5 puncte de interactiune netfilter, IP Tables inregistreaza unul sau mai multe module software; fiecare dintre ele creeaza o tabela cu reguli ce se afla sub controlul administratorului Linux. Tabelele introduse de IP Tables sunt:

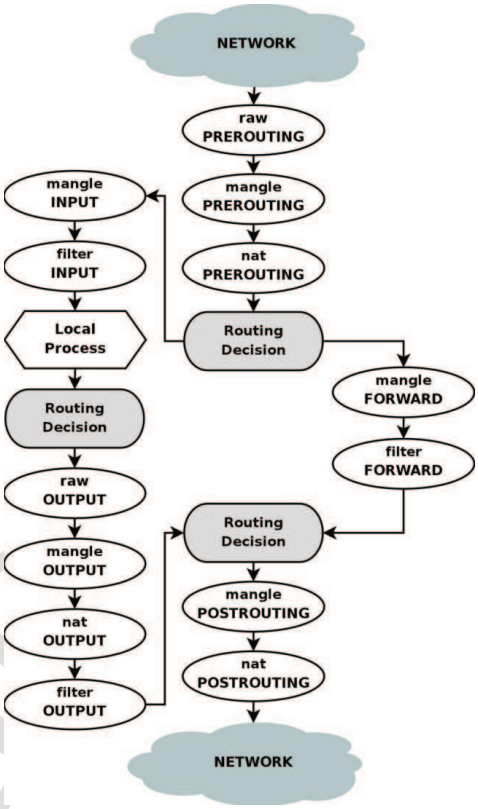
- **filter** – in aceasta tabela sunt cuprinse toate regulile ce tin de filtrarea pachetelor (fara modificarea continutului)
- **nat** – este tabela in care se introduc regulile de modificare a adresei sursa sau destinatie a pachetelor (impreuna cu porturi sursa/destinatie, daca este cazul). Tabela NAT este diferita de filter, in aceea ca numai primul pachet parcurge aceasta tabela; rezultatul acestei traversari va fi aplicat tuturor pachetelor ce fac parte din aceeasi conexiune
- **conntrack** – este tabela in care se mentin informatiile legate de dialogurile curente (conexiuni, sesiuni) ale statiei Linux sau ale altor statii ce o tranziteaza. Informatiile din aceasta tabela sunt folosite atat de catre NAT, cat si pentru partea de filtrare a pachetelor (ex: pt a realiza un firewall stateful)
- **mangle** – tabela folosita pentru modificarea continutului pachetelor sau a meta-informatiei acestora
- **raw** – tabela folosita pentru a marca explicit acele pachete pentru care nu se doreste connection tracking. Anularea connection tracking-ului pentru o parte a traficului determina o incarcare mai mica a statiei pe care ruleaza firewall-ul

Pentru fiecare tabela, in unul sau mai multe dintre cele 5 puncte de interactiune sunt definite seturi de reguli (“rule chain”). In functie de punctul de interactiune, chain-urile poarta urmatoarele denumiri (vezi si imaginea):

- **PREROUTING** – reprezinta punctul de interactiune aflat inaintea procesului de routare. Este locul ideal pentru modificarea adresei destinatie a pachetului (vezi DNAT)
- **INPUT** – reprezinta unicul punct de interactiune prin care trec pachetele destinate statiei in cauza - au ca adresa IP destinatie una dintre adresele statiei. Aceste pachete sunt pasate unui proces local
- **OUTPUT** – este punctul de interactiune prin care “curg” pachetele generate de catre statia Linux (ex: pachetele echo-request generate de o comanda ping rulata din shell)



- **FORWARD** – prin acest punct de interactiune trec pachetele care nu sunt destinate statiei – cele a caror adresa destinatie nu corespunde cu niciuna dintre adresele IP ale statiei
- **POSTROUTING** – acest punct de interactiune este ultimul loc in care se poate interveni asupra pachetelor inainte ca acestea sa paraseasca statia. Este locul potrivit pentru SNAT.



Fiecare modul IP Tables actioneaza numai in anumite puncte de interactiune, dupa cum urmeaza:

- filter – actioneaza numai in INPUT, OUTPUT si FORWARD
- mangle – actioneaza in toate cele 5 puncte de interactiune
- nat – actioneaza numai in PREROUTING, OUTPUT si POSTROUTING
- conntrack – actioneaza in toate cele 5 puncte
- raw – actioneaza in PREROUTING si in OUTPUT

4.2.2.2. Seturi de reguli

Un chain (set de reguli) reprezinta o insiruire de reguli, fiecare dintre ele avand forma *criterii => actiune*. Ordinea regulilor conteaza! Fiecare pachet care parcurge chain-ul este comparat pe rand cu setul de criterii al fiecarei reguli in ordinea in care regulile au fost plasate in chain si, daca a corespuns ("match"), i se aplica actiunea corespunzatoare ("target"). Toate regulile de dedesubt nu vor mai fi evaluate!

Nota: *chain-urile PREROUTING si POSTROUTING au regim special, in sensul in care numai primul pachet dintr-o conexiune/dialog intre doua statii traverseaza chain-ul. Odata ce s-a luat o decizie asupra lui, aceeasi actiune va fi aplicata automat tuturor celorlalte pachete apartinatoare de conexiunea/dialogul in cauza.*

Exista doua categorii de chain-uri:

- cele built-in (PREROUTING, INPUT, OUTPUT, FORWARD, POSTROUTING) – sunt predefinite si nu pot fi sterse
- chain-uri custom definite de catre administrator. Pachetele pot ajunge intr-un astfel de chain prin "deviere" dintr-unul dintre chain-urile built-in. Daca un pachet termina de parcurs un chain custom, nepotrivindu-i-se nicio regula, el se va "intoarce" in chain-ul parinte si va continua parcurgerea regulilor de acolo

In cazul chain-urilor built-in, atunci cand un pachet nu se potriveste cu niciuna dintre regulile unui chain, i se va aplica asa-numitul "policy" - politica implicita a chain-ului. Aceasta poate fi stabilita de catre administrator si consta in general in acceptarea sau distrugerea pachetului. Chain-urile custom nu dispun de policy!

Exemplu: fie chain-ul cu compozitia de mai jos:

```
1. daca adresa sursa a pachetului face parte din reteaua 192.168.0.0/24 distruge-l
2. daca adresa destinatie a pachetului este 10.0.0.5 permite-i trecerea
policy: distruge toate pachetele care ajung aici
```

Iata actiunile aplicate pentru diferite categorii de pachete posibile ce ar traversa acest chain:

Caracteristici pachet	Actiune aplicata	Nr regula aplicata	Explicatii
Adresa sursa 192.168.0.5, destinatie 10.0.0.100	Pachet distrus	1	Pachetul are adresa sursa din reteaua 192.168.0.0/24 si deci se potriveste cu regula 1
Adresa sursa 192.168.0.5, destinatie 10.0.0.5	Pachet distrus	1	Desi pachetul s-ar potrivi atat cu regula 1 cat si cu regula 2, regula 1 da verdictul deoarece este prima care face match si, in aceste conditii, cele de dedesubt nu se mai evalueaza
Adresa sursa 1.2.3.4, destinatie 10.0.0.5	Pachet acceptat	2	Adresa sursa nu se potriveste cu regula 1, insa adresa destinatie corespunde regulii 2
Adresa sursa 1.2.3.4 si destinatie 10.0.0.100	Pachet distrus	policy	Pachetul nu se potriveste cu niciuna dintre reguli, in consecinta I se aplica politica default a chain-ului

4.2.3. Administrarea regulilor IP Tables

4.2.3.1. Sintaxa generala

Regulile de firewall Linux se administreaza cu ajutorul comenzii iptables si al diversilor sai parametri si optiuni. Sintaxa de baza este urmatoarea:

```
# pentru stabilirea politicii default pe un chain
iptables -P  nume_chain  politica_default

# pentru operatii cu chain-uri (adaugare, stergere, vizualizare reguli)
iptables <-t nume_tabela>  -actiune  nume_chain

# pentru administrare reguli dintr-un chain (adaugare, stergere, modificare reguli)
iptables  <-t nume_tabela>  -actiune  nume_chain  -criterii  -j target
```

Optiunea -t specifica tabela in care se adauga regula – cu alte cuvinte, tipul de actiune aplicat pachetului. Atunci cand optiunea -t lipseste, este folosita implicit tabela filter. Nu vor functiona decat comenzile care combina in mod corect tabela si chain-ul (ex: nu putem face filtrare in PREROUTING).

Numele de chain poate fi unul dintre cele 5 built-in sau un nume de chain creat de catre administrator.

Vor fi detaliate in continuare celelalte optiuni iptables.

4.2.3.2. Componentele unei reguli

Dupa cum s-a spus anterior, o regula a unui chain iptables are doua portiuni:

- setul de caracteristici pe care trebuie sa il aiba un pachet pentru a i se aplica regula in cauza (in original “match”). Exista urmatoarele tipuri de match:
  - **built-in** – aceste match-uri reprezinta reprezinta acele caracteristici pe care le va avea orice pachet si care sunt deci suportate din oficiu de catre iptables. Spre exemplu, orice pachet are adrese IP sursa si destinatie, protocol incapsulat etc. Match-urile built-in pot avea optiuni (vezi mai jos)
  - **aditional** – IP Tables este un sistem extensibil, gandit modular, astfel incat sa poata fi adaugate oricand noi module de matching sau noi target-uri. Aceste noi criterii de selectie a pachetelor sunt in general disponibile sub forma de module de kernel, avand numele de forma *xt_numematch*.
- actiunea aplicata pachetelor in caz de potrivire (asa-numitul “target”). Si acestea se impart in target-uri built-in (disponibile permanent, cum ar fi ACCEPT si DROP) si in target-uri adaugate sub forma de modul

Atat match-urile cat si target-urile pot avea optiuni, care nu pot fi utilizate decat impreuna cu target-ul/match-ul respectiv. Spre exemplu, nu putem pune conditii in functie de portul sursa/destinatie al pachetelor decat daca acestea incapsuleaza protocolul TCP sau UDP; atunci cand decidem respingerea unui pachet (notificand expeditorul) putem alege ce fel de mesaj sa fie folosit pentru notificare etc.

4.2.3.3. Actiuni aplicabile unui pachet (“targets”)

In cadrul unei reguli iptables, actiunea aplicata pachetului in caz de potrivire se specifica folosind optiunea -j (“jump”). Ceea ce urmeaza lui -j poate fi:

- un nume de target. Target-ul respectiv, odata aplicat, poate cauza:
  - incheierea parcurgerii chain-ului (majoritatea target-urilor)
  - continuarea parcurgerii chain-ului (in cazul target-ului LOG)
- un nume de chain. Aceasta determina saltul catre un alt chain, evaluarea regulilor continuand de acolo. Chain-ul in cauza este de obicei unul custom, creat de catre administratorul firewall-ului. Aceasta abordare este utila pentru structurarea si optimizarea firewall-ului, astfel incat pachetele sa treaca pe cat posibil printr-un numar minim de reguli

Iata cele mai des folosite target-uri:

- **ACCEPT** – trecerea pachetului prin chain va fi permisa
- **DROP** – pachetul va fi distrus fara instiintarea expeditorului
- **REJECT** – pachetul va fi distrus, dar expeditorul va primi un mesaj de eroare ce poate fi ales de catre administratorul firewall-ului. Tipul de mesaj poate fi specificat folosind optiunea --reject-with, dupa cum urmeaza:
  - --reject-with tcp-reset – in cazul conexiunilor TCP este trimis catre expeditor un segment TCP cu flag-ul RST setat, ce cauzeaza inchiderea explicita a conexiunii. Se previne astfel timeout-ul aplicatiei care a trimis pachetul. Este varianta “politicoasa” fata de aplicatii de a filtra pachete TCP
  - --reject-with *mesaj_icmp* – unde *mesaj_icmp* poate fi *icmp-net-unreachable*, *icmp-host-unreachable*, *icmp-port-unreachable*, *icmp-proto-unreachable*, *icmp-net-prohibited* sau *icmp-host-prohibited*
- **LOG** – permite scrierea in syslog a informatiilor despre pachet (spre exemplu, daca dorim sa avem evidenta pachetelor care au fost oprite de catre firewall-ul nostru). **Atentie! LOG este unul dintre putinele target-uri care nu opresc parcurgerea chain-ului – vor fi evaluate si regulile de dedesubtul lui LOG!** Optiuni utile ale lui LOG:
  - --log-prefix prefix – introduce in fata fiecarei informatii de logging prefixul cerut. In acest fel putem separa in cadrul logurilor (vizual sau cu ajutorul unor utilitare de parsare de log) informatiile in cauza
  - --log-level nivel – specifica nivelul de importanta atasat informatiei de logging. Nivelurile posibile sunt, de la neimportant catre critic: debug, info, notice, warning, err, crit, alert, emerg. In acest fel, daemonul syslog poate trata diferentiat mesajele de logging un functie de importanta lor
  - --log-tcp-sequence, --log-tcp-options, --log-ip-options – adauga in log si numarul de secventa TCP/optiunile TCP/optiunile IP
- **REDIRECT** – realizeaza redirectionarea pachetelor catre masina locala. Este util, spre exemplu, in cazul realizarii unui proxy trasnparent: un gateway Linux poate actiona ca proxy fara ca masinile pe care le serveste sa fie constiente de acest fapt
- **RETURN** – cauzeaza incheierea parcurgerii chain-ului curent. Efectul este urmatorul:
  - daca chain-ul este unul built-in, pachetului i se aplica policy-ul chain-ului

- daca chain-ul este unul custom, parcurgerea sa se incheie, evaluarea regulilor continuand in chain-ul parinte

- **MARK –** folosit pentru "stampilarea" pachetelor cu o valoare de care apoi vor putea tine cont alte facilitati ale codului de retea din kernel. Spre exemplu, putem ca in functie de "stampila" aplicata sa limitam/garantam banda (QoS), sa routam intr-un mod mai special pachetele (policy routing) etc. "Stampila" este o valoare numerica asociata pachetului, dar care nu face parte din continutul acestuia, ci din meta-informatia asociata lui in kernel. Target-ul MARK functioneaza numai in tabela mangle si dispune de o singura optiune: --set-mark *valoare*.
- **TTL –** permite modificarea campului TTL (Time to Live) din pachetele IP, fie prin setare explicita, fie prin incrementarea/decrementarea cu un numar de unitati specificat
- **SNAT** – adresa sursa a pachetului va fi modificata la iesirea din statie (vezi capitolul de NAT). Acest target este valabil numai in tabela nat
- **DNAT** – adresa destinatie a pachetului va fi modificata; valabil doar in tabela nat
- **MASQUERADE** – asemanator cu SNAT, insa potrivit pentru interfetele de retea cu adresa asignata dinamic

4.2.3.4. Operatii asupra unui chain

Iata cateva operatii ce se pot efectua asupra chain-urilor iptables:

- **-A** *numechain*– adaugarea unei reguli la sfarsitul chain-ului ("append")
- **-L** *numechain* – listarea regulilor ce compun chain-ul. Pentru o afisare detaliata se recomanda combinatia de optiuni -vnL (afisare detaliata, numerica)
- **-D** *numechain pozitie* – stergerea unei reguli din chain. Pentru determinarea pozitiei regulilor este indicata folosirea optiunii --line-numbers la listarea chain-ului
- **-R** *numechain pozitie regula_noua*– modificarea unei reguli existente ("replace")
- **-P** *numechain policy* – setare policy (ce se intampla cu pachetele care nu corespund nici unei reguli din chain)
- **-F** *chain* – golirea chain-ului ("flush"), adica stergerea tuturor regulilor sale
- **-N** *numechain* – creare chain custom
- **-X** *numechain* – stergere chain
- **-Z** *numechain*– zero (resetare countere) chain

```
# listare detaliata a regulilor dintr-un chain
ipatbles -vnL --line-numbers FORWARD

# crearea unui chain custom
iptables -N pachete_tcp

# stabilire policy
iptables -P INPUT DROP

# stergerea tuturor regulilor din chain-ul INPUT; ATENTIE! Nu reseteaza si policy!
iptables -F INPUT
```

4.2.3.5. Criterii de selectie a pachetelor ("matches")

Exista doua categorii de criterii de selectie a pachetelor:

- cele built-in, aplicabile oricarui pachet IP. Spre exemplu, toate pachetele IP au un camp care indica protocolul incapsulat, o adresa IP sursa si una destinatie etc.
- cele suplimentare, prezente sub forma de module si accesibile prin intermediul optiunii -m.

Vom numi criteriile incluse in cele doua categorii de mai sus “criterii de baza”. Ele se specifica in cadrul instructiunii iptables folosind optiuni scurte (-p, -m, -i etc).

Multe dintre criteriile de baza (atat cele built-in, cat si cele suplimentare) dispun de optiuni proprii – subcriterii de selectie care sunt aplicabile numai in prezenta unui anumit criteriu de baza. Spre exemplu, nu putem filtra dupa port sursa/destinatie decat daca protocolul incapsulat este TCP sau UDP; nu putem filtra dupa tipul de mesaj ICMP decat daca protocolul incapsulat in pachetul IP este ICMP etc. Subcriteriile se specifica in linia de comanda iptables sub forma de optiuni GNU-style (incep cu -- urmat de numele optiunii, format din unul sau mai multe cuvinte) – exemplu: --icmp-type, --source-port etc.

Separarea traficului caruia dorim sa ii aplicam un anume tratament se face identificand caracteristicile comune ale pachetelor componente si descriindu-le in regulile din chain-urile corespunzatoare. Spre exemplu, “pachetele care vin din reteaua 10.0.0.0/24” vor avea in comun subreteaua sursa, “navigarea pe internet” va avea drept corespondent pachetele TCP cu port sursa sau destinatie 80 sau 443 etc.

Iata principalele criterii de baza definite in iptables:

- **-i interfata_intrare –** pune conditia ca pachetul sa fi intrat pe o anumita interfata de retea. Acest criteriu nu este valabil in chain-ul OUTPUT (unde pachetele sunt generate local, nu venite din retea) si in POSTROUTING (unde de asemenea pachetele pot proveni din generare locala)
- **-o interfata_iesire** – analog pentru interfata pe care pachetul paraseste statia. Nu poate fi folosit in chain-ul INPUT sau PREROUTING
- **-s adresa/subnet sursa** – pune conditia ca pachetul sa aiba o anumita adresa sursa sau ca adresa sa IP sursa sa faca parte dintr-un anumit subnet. Ex: *-s 10.0.0.5* sau *-s 192.168.0.0/27*
- **-d adresa_destinatie** – analog, dar pentru adresa/subnet destinatie
- **-p nume/numar_protocol** – filtrarea dupa campul *protocol* din headerul IP. Protocolul poate fi specificat in doua moduri:
  - numeric, pentru orice protocol. Se vor folosi numerele protocoalelor incapsulabile in IP asa cum sunt ele definite in standarde. Exemplu: protocolul GRE are numarul 47
  - text, pentru protocoalele care suporta acest lucru. Dispunem de cuvintele cheie *tcp, udp* si *icmp*

In functie de protocolul incapsulat in IP, sunt disponibile suboptiuni precum:

- pentru UDP avem subcriteriile **--sport** si **--dport**, care pun conditia ca pachetul sa contina un anumit port sursa sau/si destinatie. Sunt permise si range-uri de porturi (ex: --dport 21:23 refera porturile destinatie 21, 22 si 23) dar nu si liste de porturi individuale. Exemplu: pachetele destinate unui server web pot fi in general selectate cu *-p tcp --dport 80*
- pentru TCP exista urmatoarele subcriterii de interes:
  - **--sport** si **--dport**, cu aceleasi semnificatii ca in cazul UDP
  - **--tcp-flags** *flaguri_tinta flaguri_setate* – acest subcriteriu specifica configuratia exacta de flag-uri TCP a pachetului. Primul parametru reprezinta lista de flaguri de interes separate prin virgula, iar cel de-al doilea lista de flaguri care trebuie sa fie setate dintre cele de interes. Sunt acceptate si cvintele cheie NONE si ALL, cu semnificatiile evidente. Spre exemplu, *-p tcp --tcp-flags SYN,ACK,PSH,URG,FIN,RST SYN* va selecta pachetele care au doar flag-ul SYN setat; acelasi exemplu se putea formula si *-p tcp --tcp-flags ALL SYN*
  - **--syn** – un shortcut pentru exemplul de mai devreme; selecteaza doar pachetele SYN (cele responsabile cu deschiderea de conexiune)
- pentru ICMP avem subcriteriul **--icmp-type** care indica tipul de mesaj ICMP. Exemplu: pachetele trimise de comanda ping pot fi identificate prin *-p icmp --icmp-type echo-request*

Criteriile suplimentare sunt accesibile prin intermediul optiunii -m din linia de comanda iptables, sub forma **-m modul_matching**. Cateva exemple utile:

- **-m iprange –** permite specificarea de range-uri (domenii) de IP-uri pe post de sursa/destinatie a pachetului. Permite subcriteriile --src-range si --dst-range ce primesc ca parametru un domeniu de adrese de forma Ipstart-IPstop. Exemplu: *-m iprange --src-range 10.0.0.5-10.0.0.9*
- **-m mac** – permite filtrarea dupa adresa MAC sursa a frame-ului de date. Are subcriteriul atasat **--mac-source**
- **-m multiport** – permite filtrarea dupa o lista de porturi sursa sau destinatie. Are atasate subcriteriile --sports si --dports. Exemplu: -m multiport --dports 80,443
- **-m state** – permite filtrarea pachetelor in functie de starea lor de apartenenta la un dialog/conexiune existent. Accepta subcriteriul --state, cu valori posibile NEW, ESTABLISHED (apartinator de o conexiune in derulare), RELATED (apartinator de o conexiune dependenta de o alta aflata in derulare), INVALID (nu apartine de o conexiune established sau related si nu este un pachet valid de deschidere de conexiune). Acest criteriu este utilizat pentru implementarea unui firewall stateful
- **-m limit** – permite limitarea numarului de aplicari ale unei reguli in unitatea de timp. Spre exemplu, putem permite conexiunile incoming (prin intermediul unei reguli cu target ACCEPT) insa nu mai mult de 20/secunda, sau putem face logging pachetelor invalide (cu -j LOG) insa nu mai mult de 60 loguri/minut etc. Suboptiuni posibile:
  - --limit *numar/interval_timp* – regula in cauza se va aplica numai daca rata pachetelor nu depaseste numarul de pachete specificat in intervalul de timp specificat. Intervalul poate fi second, minute, hour sau day. Spre exemplu, *-m limit --limit 5/second -j ACCEPT* va accepta numai primele 5 pachete venite intr-o secunda, iar pentru urmatoarele din aceeasi secunda regula nu va mai fi aplicata (nu va face match)
  - --limit-burst *rezerva_pachete* – specifica dimensiunea maxima a unui burst (grup de pachete ce pot veni simultan si se potrivesc regulii)
- **-m mark** – permite filtrarea pachetelor in functie de mark (“stampila” setata intr-o alta regula din cadrul firewall-ului). Accepta ca subcriteriu –mark urmat de valoarea mark-ului. **Atentie! Mark-ul unui pachet face parte din meta-informatia pachetului si nu se propaga in afara masinii Linux!**
- **-m owner** – permite filtrarea pachetelor in functie de caracteristicile procesului care le-a generat. Aplicandu-se numai pachetelor generate local, este firesc ca acest criteriu de selectie sa fie aplicabil doar in chain-ul OUTPUT. Subcriteriile posibile sunt:
  - --uid-owner UID – filtreaza dupa UID-ul atasat procesului
  - --gid-owner GID – filtreaza dupa GID-ul cu care ruleaza procesul
  - --pid-owner PID – filtrare dupa PID-ul procesului care a generat pachetul
  - --sid-owner – filtrare dupa session id-ul procesului care a generat pachetul. Este considerata “sesiune” totalitatea subproceselor si thread-urilor pornite de un anume proces; session ID-ul va fi PID-ul acestui proces parinte
- **-m hashlimit** – ofera un serviciu asemanator cu -m limit, insa limita se poate pune per IP sau per combinatie (IP, port). Optiuni posibile:
  - FIXMEEE
- **-m string** – permite selectia pachetelor in functie de prezenta unui anumit string in cadrul continutului pachetului. Sirul de cautat se specifica cu suboptiunea --string urmat de pattern-ul cautat
- **-m tos** – filtreaza pachete in functie de valoarea campului TOS (type of service) din headerul IP
- **-m ttl** – filtreaza pachete in functie de valoarea campului TTL (Time To Live) din headerul IP
- -m u32 – filtreaza pachete in functie de prezenta unei anumite succesiuni de octeti (maxim 4) in cadrul pachetului

Nota: *informatii generale despre optiunile iptables se pot obtine cu iptables --help. Daca se doreste vizualizarea subcriteriilor unui anumit protocol sau modul de macthing putem folosi iptables -p nume_sau_numar_protocol --help sau iptables -m modul_matching --help.*

4.2.4. Firewall-uri stateful

4.2.4.1. Stateful vs stateless

Un firewall stateless este unul care trateaza fiecare pachet primit in mod independent de celalalte, tinand cont strict de informatia inclusa in pachet, fara a o corela cu cea a altor pachete primite anterior. Un astfel de firewall va permite in mod eronat trecerea unor pachete, din cauza ca nu tine cont de ordinea in care ar trebui sa se succeada pachetele unui anumit protocol. Iata scurte exemple:

- o sesiune de ping debuteaza cu un pachet echo request, urmat de unul echo reply; un firewall stateless ar permite, in mod independent, pachetele echo request intr-un sens si pachetele echo reply in celalalt. Aceasta lasa posibilitatea unui pachet echo reply sa treaca prin firewall chiar daca nu corespunde unui pachet echo request trimis anterior
- o conexiune TCP cu un server web debuteaza cu etapa de hanshake, in care primele pachete schimbate au flag-ul SYN setat. Daca un firewall stateless permite trecerea pachetelor catre portul 80 intr-un sens si de la portul 80 in celalalt sens, atunci el va accepta si pachete care au o combinatie de flag-uri TCP nepotrivita cu stadiul curent al conexiunii (ex: va accepta orice pachet cu portul sursa 80, indiferent de configuratia sa de flag-uri). Un atacator poate folosi aceasta vulnerabilitate pentru scanarea retelei din spatele firewall-ului sau chiar pentru atacuri active

Un firewall stateful este unul care identifica dialogurile/conexiunile de care apartin pachetele care il tranziteaza, putand tine cont de stadiul dialogului si neacceptand decat pachetele legitime pentru acel moment. Un astfel de firewall nu va mai trata fiecare pachet in mod independent, ci il va identifica mai intai ca apartinand sau nu de o conexiune deja existenta, putandu-i aplica reguli in functie de aceasta apartenenta. Spre exemplu, in cadrul unei conexiuni TCP, firewall-ul nu va permite trecerea unui pachet cu flag-ul SYN setat decat in etapa de handshake; de asemenea, daca un pachet este primul dintr-o conexiune TCP in formare, acesta nu va fi acceptat decat daca are flag-ul SYN setat, eliminandu-se astfel o intreaga serie de tehnici de scanare a retelei.

Trebuie precizat ca in acest capitol folosim termenul de conexiune intr-un sens foarte larg; un termen mai potrivit ar fi acela de dialog intre doua statii. Doar o parte dintre protocoalele stivei TCP/IP sunt orientate pe conexiune, insa, chiar si in cadrul unui protocol conextionless, pot fi identificate fluxuri de pachete, separabile fata de restul traficului. Spre exemplu, toate pachetele unei sesiuni ping au un indentificator comun (camp cuprins oficial in header-ul ICMP) si un numar de secventa crescator, deci sunt tratabile ca un dialog distinct. Vom numi in continuare “dialog” intre doua statii o succesiune de pachete schimbate intre aceeasi pereche de adrese sursa/destinatie, cu acelasi protocol incapsulat in IP si, daca este cazul, aceleasi porturi sursa si destinatie.

4.2.4.2. Connection tracking in Linux

In Linux, urmarirea dialogurilor este realizata cu ajutorul modulului de connection tracking. Acesta actioneaza automat in toate chain-urile in afara de FORWARD si mentine pentru fiecare dialog detectat o intrare intr-o tabela de conexiuni. Continutul tabelei poate fi afisat prin vizualizarea fisierului */proc/net/ip_conntrack*.

Din punct de vedere al modulului de connection tracking, un pachet poate avea urmatoarele stari posibile:

- **NEW** – este primul pachet al unui nou dialog. Spre exemplu, in cazul unei conexiuni TCP acesta este in cele mai dese cazuri un pachet SYN. Trebuie insa inteles ca un pachet care nu are flag-ul SYN setat este considerat tot NEW, in masura in care este primul pachet “vazut” de modulul de connection tracking din acel dialog
- **ESTABLISHED** – pachetul este recunoscut ca facand parte dintr-un dialog deja existent in tabela de connection tracking. O conexiune devine established atunci cand a existat deja trafic in ambele directii
- **RELATED** – pachetul apartine de un dialog aflat in legatura cu unul deja stabilit. Spre exemplu, conexiunile de date deschise intre clientul si serverul FTP sunt considerate RELATED cu cea de

comanda; de asemenea, mesajele de eroare ICMP care se intorc ca urmare a unui pachet IP sunt considerate RELATED
- **INVALID** – este cazul pachetelor pentru care nu se poate determina starea (de ex, din cauza epuizarii resurselor sistemului) sau pachete de eroare ICMP care nu apartin de nicio conexiune established
- **UNTRACKED** – corespunde pachetelor care au fost marcate ca "untracked" in tabela raw. Pentru aceste pachete modulul de connection tracking nu ia nici un fel de masura, neocupand resurse

IP Tables permite filtrarea pachetelor si in functie de starea lor, asa cum este ea detectata de modulul de connection tracking, cu ajutorul match-ului **-m state**. Acesta dispune de subcriteriul --state, starile posibile fiind cele enumerate mai sus. Astfel, pentru a permite trecerea traficului web prin FORWARD, putem scrie:

```
iptables -A FORWARD -p tcp --dport 80 -m state --state NEW --syn -j ACCEPT
iptables -A FORWARD -p tcp -m state --state ESTABLISHED,RELATED -j ACCEPT
```

4.3. NAT (Network Address Translation)

4.3.1. Concepte si tipuri de NAT

NAT este un procedeu prin care adresa sursa sau destinatie a pachetelor este inlocuita la trecerea acestora prin statia Linux. Linux defineste doua tipuri de NAT:
- **SNAT** (Source NAT) - este acea forma de NAT in care adresa sursa a pachetelor (si eventual portul sursa) este inlocuita cu alta la parasirea statiei Linux, insa cu mentinerea unei evidente astfel incat pachetelor de intoarcere sa li se aplice transformarea inversa. Scenariul tipic de utilizare a SNAT este acela al retelei cu adrese IP private care iese in internet prin una sau mai multe adrese IP publice.
- **DNAT** (Destination NAT) – este acea forma de NAT in care este modificata adresa destinatie de pachetelor, in scopul de a fi routate diferit. Scenariul tipic de utilizare este port forwarding – trimiterea pachetelor catre o alta statie in functie de portul lor destinatie. Spre exemplu, atunci cand avem un server web cu adresa privata, aflat in spatele unui NAT si care dorim sa fie accesibil din afara retelei

SNAT presupune schimbarea adresei sursa a pachetelor care parasesc statia. Singurul loc in care avem acces la toate aceste pachete, indiferent de provenienta lor – pachete care doar tranziteaza statia sau pachete generate local – este chain-ul POSTROUTING. Daca un pachet paraseste statia avand drept adresa IP sursa una dintre adresele statiei, pachetele de intoarcere vor trebui sa treaca prin transformarea inversa inainte de a patrunde in procesul de routare, asadar in PREROUTING (in caz contrar, ele ar fi livrate stivei de protocoale de pe statia care face NAT). Modulul de IP Tables reponsabil cu NAT-ul actioneaza in ambele aceste chain-uri, cele doua lucrand in tandem.

4.3.2. SNAT

4.3.2.1. Explicarea mecanismului

Sa consideram cazul unei statii Linux care joaca rolul de gateway pentru o retea cu adrese private de forma 10.0.0.*. Statia dispune de doua interfete de retea – una conectata in reteaua locala, avand adresa IP privata, si cealalta conectata la internet, cu adresa IP publica 1.2.3.4. Pachetele nu trebuie sa paraseasca reteaua locala cu adrese susa private, de aceea masina Linux va efectua NAT pachetelor provenite de la statiile locale si destinate statiilor aflate in afara retelei.

Fie doua statii client, C1 si C2, cu adresele 10.0.0.1 si 10.0.0.2. Vom analiza cazul cel mai dezavantajos – cel in care ambii clienti se conecteaza la acelasi server extern (fie adresa acestuia 2.2.2.2), folosind acelasi port sursa. Portul sursa este ales de catre fiecare sistem de operare in parte la nivel de fiecare conexiune, asadar exista o

probabilitate destul de mare ca doua statii sa aleaga, in mod independent, acelasi port sursa, mai ales tinand cont de faptul ca fiecare statie poate avea mai multe conexiuni in desfasurare la un moment dat.

Succesiunea de operatii in cazul conectarii celor doi clienti la acelasi server si folosind acelasi port sursa este urmatoare:

- C1 trimite un pachet cu adresa sursa 10.0.0.1 si portul sursa 1025, destinat statiei 2.2.2.2 portul 80
- motorul NAT de pe statia Linux primeste pachetul si ii modifica adresa sursa. Pachetul va pleca cu sursa 1.2.3.4:1025 si destinatia neschimbata (2.2.2.2:80). Motorul NAT va memora faptul ca 10.0.0.1:1025 a fost transformat in 1.2.3.4:1025, astfel incat la primirea unui pachet cu adresa destinatie 1.2.3.4:1025 sa stie ca trebuie sa modifice inapoi adresa 1.2.3.4 in 10.0.0.1. Setul de astfel de reguli de transformare este stocat in ceea ce se numeste *tabela NAT*
- serverul 2.2.2.2 receptioneaza pachetul (traind cu impresia ca dialogheaza cu statia 1.2.3.4) si genereaza un pachet de raspuns. Acesta va avea adresa/port sursa 2.2.2.2:80 si destinatie 1.2.3.4:1025
- statia Linux primeste pachetul de raspuns. Daca pachetul ar ajunge nemodificat in procesul de routare, acesta din urma ar decide ca este pentru masina Linux si l-ar pasa stivei de protocoale locale, nu l-ar mai transmite lui C1. De aceea adresa destinatie 1.2.3.4 trebuie inlocuita cu 10.0.0.1 inainte de intrarea pachetului in procesul de routare. Singurul loc in care modificarea poate fi efectuata este PREROUTING. Statia Linux efectueaza inlocuirea, efectul fiind ca procesul de routare va trimite acum pachetul prin FORWARD catre destinatia lui de drept, C1
- C2 trimite un pachet catre acelasi server. Pachetul are adresa/port sursa 10.0.0.2:1025 (C2 poate alege, independent de C1, acelasi port sursa)
- Motorul NAT primeste pachetul si inlocuieste adresa si portul sursa. Daca s-ar limita la inlocuirea adresei, pachetul ar pleca cu aceeasi adresa/port sursa ca in cazul lui C1 (1.2.3.4:1025), consecinta fiind ca pachetele de intoarcere pentru C1 nu ar mai putea fi distinse de pachetele destinate lui C2, deoarece toate ar avea adresa destinatie 1.2.3.4 si portul destinatie 1025. De aceea motorul NAT este fortat sa efectueze si o translatie de port, astfel incat sa poata decide destinatia reala a pachetelor pe care le primeste pe baza portului lor destinatie. Practic se realizeaza o corespondenta port-client pentru fiecare conexiune a mai multor clienti cu acelasi server extern

4.3.2.2. Modalitate de realizare cu iptables

Pentru a implementa SNAT pe o masina Linux sunt necesare urmtoarele conditii:

- **incarcarea modulelor de kernel necesare**, daca functionalitatea dorita nu a fost inclusa ca parte a kernel-ului. Distingem aici doua aspecte:
  - incarcarea modulului care creeaza tabela IP Tables numita nat. Acesta se numeste de obicei *iptable_nat*
  - incarcarea modulelor corespunzatoare asa-numitelor NAT helper-e – sunt cele care “customizeaza” operatiile necesare in procesul de NAT pentru protocoalele mai speciale. Spre exemplu, protocolul FTP transmite adrese IP si porturi ale clientului/serverului ca parte a mesajelor FTP; aceste adrese/porturi trebuie la randul lor modificate la trecerea prin NAT

```
modprobe iptable_nat
# helper pentru FTP
modprobe nf_conntrack_ftp
modprobe nf_nat_ftp
```

- **activarea facilitatii IP forwarding**. Aceasta permite statiei Linux sa comute pachete de pe o interfata de retea pe alta, sau in cadrul aceleiasi interfete. Chiar daca are doua interfete de retea, implicit o statie

Linux nu va comuta pachetele de pe una pe alta. Activarea facilitatii se face shimband un parametru de kernel, prin modificarea unui pseudo-fisier din /proc:

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

- **adaugarea unei reguli iptables pentru SNAT**. Regula este adaugata in POSTROUTING si in tabela nat. Procedeul NAT poate fi aplicat exact pachetelor dorite prin folosirea de criterii si subcriterii. In cadrul regulii SNAT se specifica si adresa sau adresele sursa cu care vor pleca pachetele, folosind optiunea --to-source. In cazul in care exista mai multe adrese publice Iata un exemplu de SNAT cu adresa 1.2.3.4 aplicat numai pachetelor web provenite din reteaua 10.0.0.0/24:

```
iptables -t nat -A POSTROUTING -s 10.0.0.0/24 -p tcp -m multiport --dports 80,443 -j SNAT --to-
source 1.2.3.4
iptables -t nat -A POSTROUTING -s 192.168.0.0/28 -j SNAT --to-source 1.2.3.4-1.2.3.10
```

Atentie! *Cand o statie Linux actioneza ca gateway pentru o retea, trebuie sa avem grija sa aplicam SNAT numai pachetelor care parasesc reteaua, nu si celor destinate retelei locale! Aceasta problema poate fi in general rezolvata prin aplicarea optiunii -o pentru a specifica interfata de iesire a pachetelor*

- **permiterea accesului pachetelor prin FORWARD sau OUTPUT**. Procedeul SNAT se aplica numai pachetelor care tranziteaza POSTROUTING; pachetele parcurg chain-ul POSTROUTING abia dupa trecerea prin chain-urile amintite anterior, de aceea este esential ca acele doua chain-uri sa permita trecerea pachetelor. Avand in vedere ca modificarea adresei sursa se efectueaza abia in POSTROUTING, pachetele traverseaza FORWARD sau OUTPUT cu adresa sursa originala, nealterata.

4.3.2.3. Alte target-uri de tip SNAT utile

Sistemul IP Tables pune la dispozitia administratorului si alte target-uri de tip SNAT, utilizabile in scenarii mai speciale:

- **MASQUERADE** - desemneaza o forma dinamica de SNAT. Aceasta se intalneste in retele in care toate calculatoarele cu IP-uri private folosesc pentru iesirea in internet o adresa IP rutabila unica. Observam faptul ca SNAT presupune specificarea de la bun inceput a adresei/adreselor sursa pentru pachete; acest lucru insa nu este convenabil atunci cand adresa interfetei de iesire este alocata dinamic si se schimba in timp (ex: conexiuni de tip dial-up, PPPoE etc). Operatia de masquerading sub Linux (cu actiunea aferenta -j MASQUERADE) va inlocui intotdeauna adresa sursa a pachetelor cu adresa instantanee a interfetei de iesire. Concluzionam ca, in conditiile in care pachetele pleaca cu o singura adresa sursa si aceasta este stabila, SNAT si MASQUERADE sunt echivalente; in cazul dinamic vom prefera MASQUERADE.
- **SAME** – o forma de SNAT care incearca sa aloce aceluiasi client aceeasi adresa de fiecare data, chiar si atunci cand exista mai multe adrese publice disponibile pt NAT
- **NETMAP** – efectueaza NAT mapand IP-urile 1:1 pentru doua subnet-uri de aceeasi dimensiune

4.3.3. DNAT

DNAT presupune schimbarea adresei si eventual portului destinatie ale pachetelor. Aceasta schimbare are sens doar in masura in care se poate tine cont de noua adresa, asadar inainte procesului de routare; in consecinta, locul potrivit pentru DNAT este chain-ul PREROUTING.

Reusita DNAT presupune aceleasi 3 conditii ca in cazul SNAT. Ce difera este forma comenzii pentru adaugarea unei reguli de DNAT. Sa consideram urmatorul exemplu: intr-o retea cu adrese private se afla un

server web, cu adresa 10.0.0.100 si care asculta pe portul 8888. Intreaga retea “iese in internet” prin adresa publica 5.6.7.8. Dorim ca serverul web sa fie accesibil din afara retelei, pe portul 80; in acest scop, trebuie sa trimitem pachetele destinate adresei publice a retelei (unica vizibila din internet) si portului 80 catre statia 10.0.0.100, schimband si portul destinatie. Regula DNAT corespunzatoare se adauga astfel:

```
iptables -t nat -A PREROUTING -d 5.6.7.8 -p tcp --dport 80 -j DNAT --to-destination
10.0.0.100:8888
```

4.4. Exemple de firewall-uri

4.4.1. Filtrarea accesului din exterior pentru o statie

Ipoteza: avem un singur host conectat la internet, echipat cu o singura interfata eth0.

```
1 #iptables -P INPUT DROP
2 #iptables -A INPUT -i eth0 -p icmp –icmp-type echo-request -m limit --limit 1/s -j ACCEPT
3 #iptables -A INPUT -i eth0 -s 10.0.0.1 -p tcp --dport ssh -j ACCEPT
4 #iptables -A INPUT -i eth0 -p udp --dport 137:138 -j LOG
5 #iptables -A INPUT -i eth0 -p udp --dport 137:138 -j DROP
6 #iptables -A INPUT -i eth0 -p tcp -m multiport --destination-ports 139,445 -j LOG
7 #iptables -A INPUT -i eth0 -p tcp -m multiport --destination-ports 139,445 -j DROP
8 #iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

Explicatii:

1 Se alege politica de DROP, urmand ca regulile urmatoare sa permita in mod controlat trecerea doar pentru traficul dorit
2 Se accepta echo-request din exterior, adica verificarea conectivitatii prin ping. Se limiteaza numarul pachetelor icmp care se primesc la 1/s – evita o forma de DOS (denial of service)
3 Se acepta conectivitatea pentru serviciul SSH doar de la o anumita adresa IP
4-7 Se logheaza si apoi se arunca pachetele destinate serviciului de File and Printing Sharing al Microsoft
8 Se accepta toate pachetele a caror conexiune a fost initiata de pe calculatorul local

Nota: *daca doriti logarea pachetelor este dezirabila modificarea /etc/syslog.conf cu specificarea unui fisier de logging separat pentru kern.=warning, in caz contrar riscand poluarea logurilor cu multe informatii generate de firewall.*

4.4.2. Filtrarea acccesului la o retea aflata in spatele firewall-ului

Ipoteza: avem un calculator cu 2 interfete de retea, eth0 conectata catre LAN (IP 10.0.0.1) si eth1 conectata la providerul de internet - WAN (IP 193.0.0.100). Cerinte:

- se doreste ca PC-urile din retea sa poata accesa doar serviciile de web si FTP din Internet
- conexiunile din LAN catre orice server de SMTP din internet vor fi redirectionate catre serverul local de mail (aflat pe acelasi calculator cu firewall-ul) – acest server nu va accepta conexiuni din exterior. Procedeul poate face parte dintr-o politica de spam filtering. Daca se va incerca accesul la un alt server de mail (tcp 25), tentativele se vor loga iar conexiunile se vor redirecta catre serverul local de mail
- serverul de DNS si de web de pe statie firewall vor fi accesibile in Internet si in LAN

```
01 #iptables -P INPUT DROP
02 #iptables -P FORWARD DROP
03 #iptables -P OUTPUT ACCEPT
04 #iptables -t nat -A POSTROUTING -o eth1 -j SNAT --to-source 193.0.0.100
```

```
05 #iptabels -t nat -A PREROUTING -i eth0 ! -d 10.0.0.1/24 -p tcp --dport 25 -j LOG --log-prefix
"SMTP ALERT"
06 #iptables -t nat -A PREROUTING -i eth0 ! -d 10.0.0.1/24 -p tcp --dport 25 -j DNAT --to-
destination 10.0.0.1:25
07 #iptables -A INPUT -i lo -j ACCEPT
08 #iptables -A INPUT -i eth0 -s 10.0.0.0/24 -j ACCEPT
09 #iptables -A INPUT -i eth1 -p udp --dport 53 -j ACCEPT
10 #iptables -A INPUT -i eth1 -p tcp --dport 53 -j ACCEPT
11 #iptables -A INPUT -i eth1 -p tcp --dport 80 -j ACCEPT
12 #iptables -A INPUT -i eth1 -m state --state RELATED,ESTABLISHED -j ACCEPT
13 #iptables -A FORWARD -i eth0 -o eth1 -s 10.0.0.0/24 -p tcp --dport 80 -j ACCEPT
14 #iptables -A FORWARD -i eth0 -o eth1 -s 10.0.0.0/24 -p tcp --dport 21 -j ACCEPT
15 #iptables -A FORWARD -i eth1 -o eth0 -m state --state RELATED,ESTABLISHED -j ACCEPT
```

Explicatii:

1-3. Alegem politici de DROP pentru INPUT si FORWARD si de ACCEPT pt OUTPUT
4. Se face SNAT, adica toate pachetele routate catre Inet vor avea, chiar inainte de fi injectate pe mediul de transmisie, IP-ul sursa schimbat cu IP-ul rutabil al firewall box-ului – in caz contrar, pachetul ar ajunge la destinatie dar nu ar putea fi rutat prin Inet inapoi catre firewall-ul nostru.
5-6. Toate pachetele plecate din LAN cu port destinatie 25 (SMTP) si care nu au IP destinatie serverul nostru de e-mail, vor fi logate – vom avea astfel evidenta acelor tentative de a evita folosirea serverului ‘oficial’ de mail al companiei, server care ar putea avea implementate facilitati de scanare antivirus, filtrari anti-spam, etc. Dupa logarea pachetelor, acestea sunt redirectionate catre serverul nostru de e-mail, conform cu regula 6.
7. Pachetele de la interfata logica loopback sunt acceptate – este obligatorie aceasta regula pentru ca anumite servere sa functioneze!
8. Acceptam toate conexiunile originate in LAN cu destinatia firewall.
9-11. Acceptam toate conexiunile originate in WAN daca ele se adreseaza serverelor de web (tcp 80) sau/si DNS (udp/tcp 53) aflate pe firewall.
12. Acceptam din WAN raspunsurile la conexiunile initate de pe firewall (chainul este de INPUT).
13-14. Acceptam toate conexiunile cu originea in LAN si destinatie in WAN daca ele se adreseaza serviciilor de web sau/si FTP
15. Se accepta pachetele dinspre Inet catre LAN daca ele fac parte dintr-o conexiune deja stabilita.

Nota: *pentru ca acest scenariu sa functioneze este necesara activarea facilitatii IP forwarding pe statia Linux (vezi capitolul despre NAT)*

4.5. BIBLIOGRAFIE SI LINK-URI UTILE

- Documentatia oficiala netfilter: http://www.netfilter.org/documentation/
- Packet Filtering HOWTO: http://www.netfilter.org/documentation/HOWTO/packet-filtering-HOWTO.html
- Tutorial firewall&iptables: http://www.frozentux.net/iptables-tutorial/iptables-tutorial.html
- Tutorial firewall cu sub-chain-uri: http://www.pettingers.org/code/firewall.html
- Linux Advanced Routing&Traffic Control HOWTO: http://lartc.org/howto/
- Manpage-ul utilitarului iptables: http://linux.die.net/man/8/iptables
- Comparatie NAT vs proxy: http://en.tldp.org/HOWTO/IP-Masquerade-HOWTO/what-is-masq.html
- Carti:
  - Designing and Implementing Linux Firewalls and QoS
  - Linux Firewall – Attack Detection and Response with iptables, psad and fwsnort