Linux 2.4 Packet Filter HOWTO (NL)

Rusty Russell, mailing list `netfilter@lists.samba.org`, vertaald door Sjoerd Langkemper, `sjoerd@linuxonly.nl`
$Revision: 438 $ $Date: 2001-05-26 22:26:48 +0200 (Sat, 26 May 2001) $ $ Translation 2001/05/20 22:31:47 Version 0.04 $

Deze HOWTO beschrijft hoe je pakketjes op een netwerk kan filteren onder een Linux 2.4 kernel.

Inhoudsopgave

1 Inleiding **2**

2 Waar kan ik meer informatie vinden? **3**

3 Wat is een pakket filter? **3**

3.1 Waarom zou ik een pakket filter willen? . . . 3

3.2 Hoe kan je het pakket filter instellen onder Linux? . . . 4

3.2.1 iptables . . . 4

3.2.2 Regels instellen bij het opstarten . . . 4

4 Wie is Rusty? Wie is Sjoerd? **4**

5 Rusty's Turbo Pakket Filter cursus **5**

6 Hoe de filters werken **5**

7 iptables gebruiken **6**

7.1 Wat er gebeurt als je je computer opstart . . . 7

7.2 Een regel opgeven . . . 7

7.3 Filter specificaties . . . 8

7.3.1 Herkomst en doel . . . 8

7.3.2 Inversie opgeven . . . 8

7.3.3 Protocol opgeven . . . 9

7.3.4 Een interface opgeven . . . 9

7.3.5 Fragmenten opgeven . . . 9

7.3.6 Het uitbreiden van iptables . . . 10

7.4 Doel specificaties . . . 14

7.4.1 Gebruikersreeksen . . . 15

7.4.2 Uitbreidingen: Nieuwe doelen 15
7.4.3 Speciale ingebouwde doelen 16
7.5 Reeksen onderhouden 17
7.5.1 Een nieuwe reeks maken 17
7.5.2 Een reeks verwijderen 17
7.5.3 Een reeks leeg maken 17
7.5.4 De regels in een reeks bekijken 17
7.5.5 Tellers op nul zetten 18
7.5.6 Het beleid instellen 18

8 Ipchains of ipfwadm gebruiken 18

9 NAT en een pakket filter tegelijk gebruiken 18

10 Verschillen tussen ipchains en iptables 19

11 Pakket filter advies 20

1 Inleiding

Voor deze HOWTO is enige basiskennis vereist; je behoort te weten wat ip adressen, netmasks, routing en DNS zijn. Is dit niet het geval, dan raad ik je aan de Network Concepts HOWTO te lezen.

Toen ik (Sjoerd) deze HOWTO aan het vertalen was, liep ik tegen een paar problemen en het kan zijn dat je sommige dingen raar vertaald vindt. Is dat het geval, waarschuw me dan. Hier zijn wat dingen waar ik twijfelde:

- packet filter is vertaald met pakket filter. Hiermee wordt bedoeld een stukje software wat pakketjes kan tegenhouden en doorlaten.
- rule chain is een serie filter regels, en is vertaald met reeks.
- interface is vertaald met interface en is een netwerkkaart of modem, ofwel iets waarmee een netwerk benaderd kan worden.
- connection tracking is vertaald met connectie tracking. Connectie tracking is dat netfilter bijhoud bij welke connectie pakketjes horen.
- user-defined chains is vertaald met gebruikersreeks. Dit is een reeks die niet ingebouwd is, en die door de gebruiker samengesteld is.
- queue en queue-handler zijn niet vertaald, want dit is zo een specifiek onderwerp dat als je hier wat mee gaat doen, dat je echt wel weet wat dit betekend.
- Pakketjes die 'geDROPt' worden, worden 'tegengehouden'. Dat wil zeggen dat ze in /dev/null komen om voor altijd te verdwijnen.

Deze HOWTO gaat onder andere in op hoe het pakket filter werkt, hoe NAT werkt, en hoe je de oude ipchains en ipfwadm regels kan gebruiken. Na het lezen van deze HOWTO en het toepassen van alle dingen erin is je netwerk niet gegarandeerd **veilig**. Er is niet zoiets als veilig. Je kan niet alle pakketjes tegenhouden en verwachten dat het ook nog werkt.

Gelukkig kan je met iptables precies bepalen welke pakketjes je tegen wilt houden, dus je kan zelf bepalen hoe veel risico je neemt en hoeveel gebruiksvriendelijkheid je inlevert.

Vertaald door Sjoerd Langkemper, sjoerd@linuxonly.nl

2 Waar kan ik meer informatie vinden?

Er zijn drie officiele websites:

- Met dank aan *Filewatcher* `http://netfilter.filewatcher.org/` .
- Met dank aan *The Samba Team and SGI* `http://netfilter.samba.org/` .
- Met dank aan *Harald Welte* `http://netfilter.gnumonks.org/` .

Die laatste site was offline de laatste keer dat ik (Sjoerd) keek.

De officiele mailinglist kan je vinden op

Samba's Listserver `http://lists.samba.org` .

3 Wat is een pakket filter?

Een pakket filter is een stukje software wat naar de *header* van pakketjes kijkt terwijl ze langskomen. In de header staat informatie over het pakketje en dat wordt onder andere gebruikt om het lot van het pakketje te bepalen. Meestal is dat ofwel doorlaten (accept), ofwel tegenhouden (drop).

Onder Linux zit het pakket filter in de kernel ingebouwd en dat biedt wat meer mogelijkheden voor wat er met de pakketjes kan gebeuren. Natuurlijk is het idee nog steeds hetzelfde: aan de hand van informatie over het pakketje wordt besloten wat ermee gebeurt.

3.1 Waarom zou ik een pakket filter willen?

Gezag:

Als je je Linux computer gebruikt om je interne netwerk aan het internet of een ander netwerk te hangen, krijg je de mogelijkheid om bepaalde pakketjes wel door te laten, en andere niet. Wil je niet dat je werknemers naar www.playboy.com gaan, dan kan dat met een pakket filter.

Veiligheid:

Als je Linux computer de grens is tussen het chaotische internet vol met crackers, hackers en scriptkiddies en je eigen veilige netwerkje, dan is het fijn om te weten dat je al dit slechts buiten kan sluiten met een pakket filter. Aangezien de meeste mensen alleen de connecties naar buiten willen maken en voorkomen dat er een connectie naar binnen wordt gemaakt, kan een pakket filter in veel situaties veiligheid bieden.

Controle:

Als er iets vreemds gebeurd op je netwerk, zoals een machine die opeens allemaal pakketjes begint te versturen zonder reden, wil je er vast vanaf weten. Als je weet wat er gebeurt op je netwerk, kan je ingrijpen als het nodig is.

3.2 Hoe kan je het pakket filter instellen onder Linux?

Linux had al een pakket filter sinds de 1.1 kernels. Deze eerste filters kwamen oorspronkelijk uit BSD en werden door Alan Cox in Linux ingevoerd. Toen kernel 2.0 uit kwam was er een nieuwe tool, door Jos Vos: ipfwadm was nu het programma om de filter regels mee in te stellen. In kernel 2.2 werd weer een nieuw programma ontwikkeld, namelijk ipchains. Nu is het zover dat de 2.4 kernels er zijn en nogmaals een nieuwe pakket filter hebben: iptables. Iptables brengt ook echt nieuwe opties met zich mee, die ipchains en ipfwadm niet hadden. Zo kan je nu ook regels maken op basis van de gebruiker die het pakketje verstuurd, of op basis van hoeveel connecties er al zijn.

Om het pakket filter te kunnen gebruiken moet je je kernel wel compilen met netfilter. Netfilter is het deel van de kernel wat iptables gebruikt om zijn regels te realiseren. Je hebt minimaal kernel 2.3.15 nodig, en je moet de optie CONFIG_NETFILTER aanschakelen.

3.2.1 iptables

Het programma iptables geeft aan de kernel door welke regels in werking moeten treden, en de kernel zorgt ervoor dat elk pakketje gecontroleerd wordt. Dit betekent echter wel dat als je de computer opnieuw opstart, de regels weer verloren gaan. Er zijn wel manieren om dit tegen te gaan, zie hiervoor 3.2.2 (Regels instellen bij het opstarten).

iptables vervangt ipfwadm en ipchains, maar met een speciale module kan je nog steeds je oude regels gebruiken onder iptables. Zie hiervoor 8 (Ipchains of ipfwadm gebruiken)

3.2.2 Regels instellen bij het opstarten

Omdat de pakket filter regels in de kernel worden opgeslagen, zullen deze verloren gaan bij het opnieuw opstarten van de computer.

Er zijn twee manieren om dit te voorkomen. Er zijn twee programma's, namelijk iptables-save en iptables-restore die een set regels in een bestand kunnen opslaan. Je kan dan bij het opstarten van je computer de regels laden uit een bestandje.

Je kan ook de filter regels in een scriptje zetten en het scriptje elke keer laden bij het opstarten van de computer. In het scriptje staan dan de commando's die je ook gebruikte om de huidige regels in te stellen.

4 Wie is Rusty? Wie is Sjoerd?

Rusty is de man die het Linux IP pakket filter onderhoud. Hij schreef ipchains en heeft daar veel van geleerd om iptables beter te maken. Natuurlijk heeft hij dit niet alleen gedaan en hij zou ook niet genoeg tijd hebben, als

WatchGuard http://www.watchguard.com hem hier niet voor betaalde. Rusty is geen kernel guru. Andere mensen zoals David S. Miller, Alexey Kuznetsov, Andi Kleen en Alan Cox hebben hem hiermee geholpen.

Het schrijven van iptables kostte Rusty en anderen een jaar, maar dat was zodat het in één keer goed kon.

Sjoerd is een van de webmasters bij

LinuxOnly.nl http://www.linuxonly.nl en was van plan een iptables HOWTO te schrijven toen hij de Engelse versie van dit document tegen kwam. Hij vond het wel een goed idee om dit naar het Nederlands te vertalen.

Als je 'ik' leest, betekend het 'Sjoerd'. Als Rusty iets heeft gezegd als 'I think it's a good idea' dan wordt dat meestal vertaald met 'Het is misschien een goed idee' en soms met 'Rusty denkt...'.

5 Rusty's Turbo Pakket Filter cursus

De meeste mensen hebben een PPP connectie naar het internet en willen niet meer dan voorkomen dat er connecties gemaakt worden vanaf het internet.

```
## Voeg modules in (niet nodig als deze zijn ingebouwd in de kernel).
# insmod ip_conntrack
# insmod ip_conntrack_ftp

## Nieuwe reeks regels maken, die alle inkomende connecties tegenhoud.
# iptables -N block
# iptables -A block -m state --state ESTABLISHED,RELATED -j ACCEPT
# iptables -A block -m state --state NEW -i ! ppp0 -j ACCEPT
# iptables -A block -j DROP

## Ga naar die reeks vanuit de INPUT en FORWARD reeks.
# iptables -A INPUT -j block
# iptables -A FORWARD -j block
```

6 Hoe de filters werken

Je kan je pakket filter vorm geven door regels toe te voegen aan reeksen. Zulke reeksen (**chains** in het Engels) worden regel voor regel nagelopen om te kijken of er een regel is voor het huidige pakketje.

Er zijn standaard drie reeksen in de kernel, namelijk **INPUT** (inkomend), **OUTPUT** (uitgaand) en **FORWARD** (doorsturen).

De reeksen zijn als volgt gerangschikt:

```
                              -----
Inkomend                     /     \          Uitgaand
        -->[ Router ]--->|FORWARD|------->
           [ keuze  ]     \_____/        ^
                |                        |
```

```
       v                      ----
      ---                    /    \
     /   \                  |OUTPUT|
    |INPUT|                  \____/
     \___/                      ^
       |                        |
        --> Lokale programma's --^
```

De drie rondjes staan voor de drie reeksen. Als een pakketje bij zo'n cirkel aankomt worden de regels nagelopen om het lot van het pakketje te bepalen. Als hieruit blijkt dat het pakketje moet worden genegeerd (DROP) dan wordt het pakketje gelijk daar gestopt. Als uit de reeks blijkt dat er niets mis is met het pakketje (ACCEPT) dan doorloopt het pakketje de rest van het schema.

Elke reeks bestaat uit een aantal regels en een beleid (policy). Elke regel heeft de vorm van: "Als het pakketje deze eigenschappen heeft, doe dan dit en dat met het pakketje.". Als deze regel niet van toepassing is op het pakketje (het pakketje heeft niet de eigenschappen voor deze regel), dan wordt de volgende regel gelezen. Als de laatste regel is gelezen en er is geen regel die van toepassing is op dit pakketje, dan wordt de actie uitgevoerd die het beleid is. Dit is dus een standaard actie die van toepassing is op alle pakketjes die niet door een regel gespecifeerd zijn. Het beleid is meestal het pakketje negeren (DROP) zodat onbekende pakketjes niet door kunnen dringen.

1. Als een pakketje binnenkomt (via een modem of ethernetkaart bijvoorbeeld), bepaalt de kernel eerst waar het pakketje heen moet. Dit heet routing.
2. Als het pakketje voor deze computer bedoeld is, wordt het pakketje doorgegeven aan de INPUT reeks. Als het hierdoor komt, wordt het doorgegeven aan de programma's op de computer.
3. Als het pakketje voor een andere computer bedoeld is, en de kernel kan niet forwarden of het weet niet hoe het geforward moet worden, wordt het pakketje genegeerd. Als er wel geforward kan worden, dan komt het pakketje in de FORWARD reeks. Komt het hierdoor, dan wordt het doorgestuurd naar een andere machine.
4. Als een programma op de computer een pakketje verstuurd, moet het eerst door de OUTPUT reeks om verzonden te kunnen worden.

7 iptables gebruiken

iptables heeft een goede gebruikershandleiding (`man iptables`) en deze is dus te gebruiken als je ergens meer van wil weten. Als je al bekend bent met ipchains kan je het hoofdstuk 10 (Differences Between iptables and ipchains) lezen, maar je kan natuurlijk ook gewoon doorlezen.

Er zijn drie standaardreeksen die je niet kan verwijderen, maar je kan ook nog zelf extra reeksen maken. Hier zijn wat opties waarmee je reeksen kan onderhouden:

1. Een nieuwe reeks maken (-N);
2. Een lege reeks verwijderen (-X);
3. Het beleid (policy) voor een standaardreeks wijzigen (-P);

4. De regels weergeven in een reeks (-L);
5. Alle regels uit een reeks verwijderen (-F);
6. De pakket en byte tellers op nul zetten (-Z).

Er zijn een paar manieren om regels in een reeks te onderhouden:

1. Voeg een regel toe aan het einde van een reeks (append) (-A);
2. Voeg een regel toe op een bepaalde positie in een reeks (insert) (-I);
3. Vervang een regel door een andere regel in een reeks (replace) (-R);
4. Verwijder een bepaalde regel uit een reeks (-D).

7.1 Wat er gebeurt als je je computer opstart

iptables kan in een kernel module zitten, namelijk **iptable_filter.o**. Deze module zou automatisch geladen moeten worden zodra je iptables opstart. Deze code kan ook in de kernel ingebouwd worden.

Voordat iptables een keer gestart is zijn alle standaard reeksen leeg en hebben het beleid ACCEPT, ofwel alles doorlaten. Je kan het beleid van de FORWARD reeks wijzigen door de optie "forward=0"mee te geven aan de iptable_filter module. Als op jou computer niet alle reeksen leeg zijn bij het opstarten van je computer, dan kan het zijn dat je distributie in de bootscripts de regels aanpast.

7.2 Een regel opgeven

Natuurlijk kan je niets met een pakket filter als je geen regels op kan geven. Meestal wil je gewoon een regel aan het einde toevoegen (-A) of verwijderen. Een regel ergens invoegen (-I) is handig als de volgorde van je regels belangrijk is en een regel vervangen (-R) is handig als bijvoorbeeld een IP-adres verandert.

Elke regel bepaalt welke eigenschappen een pakketje moet hebben zodat deze regel ervoor geld, en wat in dat geval met het pakketje moet gebeuren. Je kan bijvoorbeeld alle pakketjes die van het protocol ICMP zijn en die van het IP 127.0.0.1 afkomen, tegenhouden (DROP). De eigenschappen zijn dan dat het protocol ICMP is, het IP 127.0.0.1. Het doel is DROP.

127.0.0.1 is de loopback adapter. Dit IP heb je zelfs als je geen netwerk hebt. Je kan ICMP pakketjes produceren met het programma ping, wat een pakketje stuurt en meld of het aangekomen is.

```
# ping -c 1 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
64 bytes from 127.0.0.1: icmp_seq=0 ttl=64 time=0.2 ms

--- 127.0.0.1 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.2/0.2/0.2 ms
# iptables -A INPUT -s 127.0.0.1 -p icmp -j DROP
# ping -c 1 127.0.0.1
PING 127.0.0.1 (127.0.0.1): 56 data bytes
```

```
--- 127.0.0.1 ping statistics ---
1 packets transmitted, 0 packets received, 100% packet loss
#
```

Je ziet hierboven dat de eerste ping wel lukt, maar niet meer als de regel is toegevoegd aan de INPUT reeks.

Een regel kan op twee manieren verwijderd worden: ofwel door het nummer op te geven, ofwel door dezelfde opties op te geven als dat de regel gemaakt werd.

De nummering begint bovenaan met 1 en omdat er nog maar één regel in de reeks staat, is deze regel nummer 1. Deze regel kan dus als volgt verwijdert worden:

```
# iptables -D INPUT 1
#
```

De tweede manier is hetzelfde als de regel maken, alleen wordt de -A nu vervangen door -D. Dit is vooral handig als je veel regels hebt en je hebt geen zin om te tellen. Onze regel zouden we dus als volgt verwijderen:

```
# iptables -D INPUT -s 127.0.0.1 -p icmp -j DROP
#
```

Als er meerdere regels zijn die hetzelfde zijn, wordt alleen de eerste verwijderd met deze manier.

7.3 Filter specificaties

Behalve protocol (-p) en de herkomst (-s) zijn er nog andere eigenschappen waaraan je bepaalde pakketjes kan herkennen. Hier volgen alle eigenschappen waar je pakketjes mee kan aanduiden.

7.3.1 Herkomst en doel

Herkomst ('-s', '–source' of '–src') en doel ('-d', '–destination' or '–dst') IP adressen kunnen worden opgegeven op vier manieren. Natuurlijk kan je gewoon een IP-adres opgeven, maar je kan ook een hostname gebruiken, zoals 'localhost' of 'www.linuxhq.com'.

De derde en vierde manieren laten het toe om meerdere IP-adressen te specificeren. '199.95.207.0/255.255.255.0', bijvoorbeeld geeft 255 ip-adressen aan. Ook geldig is '199.95.207.0/24'. Allebei geven ze IP-adressen aan van 199.95.207.0 tot 199.95.207.255.

Twee speciale varianten hiervan zijn 1.2.3.4/32, waar de /32 aangeeft dat het IP-adres helemaal moet voldoen. Dit is standaard. 1.2.3.4/0 kan ook, in welk geval het IP-adres helemaal niet uitmaakt. In dit geval wordt meestal 0/0 opgegeven als IP adres. Dit is standaard als de -s of -d optie weggelaten wordt, dus als er geen herkomst of doel adres opgegeven wordt. Daarom wordt het ook weinig gebruikt:

```
[ NOTE: '-s 0/0' is redundant here. ]
# iptables -A INPUT -s 0/0 -j DROP
#
```

7.3.2 Inversie opgeven

Veel opties kunnen voorafgegaan worden door '!', wat staat voor 'niet'. '-s ! localhost' heeft dus betrekking tot alle pakketjes, behalve die van localhost afkomen.

7.3.3 Protocol opgeven

Het protocol kan worden opgegeven met '-p' of met '–protocol'. Het protocol kan een nummer zijn, of een naam zoals TCP, UDP of ICMP. Hoofdletters maken hier niets uit.

Ook de protocol optie kan vooraf worden gegaan door een uitroepteken: '-p! TCP' betekent alle pakketjes behalve TCP.

Het opgeven van een protocol is nodig als je regels wilt maken die afhankelijk zijn van het poortnummer waarop een connectie gemaakt wordt. Omdat ICMP geen poorten ondersteund, moet er TCP of UDP opgegeven worden.

7.3.4 Een interface opgeven

De '-i' of '–in-interface' optie geeft de interface (netwerkkaart/modem) aan waar de pakketjes de computer mee inkomen. De '-o' of '–out-interface' optie geeft de interface aan waar de pakketjes uit de computer gaan. Je kan `ifconfig` gebruiken om een lijst van de huidige interfaces te krijgen.

Pakketjes die door de INPUT reeks komen hebben geen interface waarmee ze uit de computer gaan. Regels met -o erin zullen dus nooit waar zijn en overgeslagen worden. Bij regels in de OUTPUT reeks werkt het net zo: deze hebben geen interface waarmee ze binnengekomen zijn.

Pakketjes die door de FORWARD reeks gaan hebben zowel een interface waarmee ze naar binnen zijn gekomen, als een interface waarmee ze weer naar buiten gaan.

Je mag ook een interface opgeven die nog niet bestaat. De regel geld niet zolang deze interface niet bestaat, maar zodra deze interface gaat werken treed de regel in werking. Dit is handig voor inbel-connecties, zodat er regels ingevoerd kunnen worden voor het geval er ingebeld wordt.

Bij interfaces werkt het '+' teken als een sterretje; het zal meerdere interfaces specificeren. `-i ppp+` bijvoorbeeld zal voor alle PPP interfaces gelden, maar niet voor ethernet interfaces.

Ook de interface optie kan voorafgegaan worden door een '!' om de optie om te keren.

7.3.5 Fragmenten opgeven

Soms is een pakketje te groot om in één keer verstuurd te kunnen worden. In zo'n geval wordt het pakketje opgesplitst in fragmenten en verzonden als meerdere pakketjes. Het probleem wat hierbij komt kijken is dat het eerste pakketje de headers heeft en dus moeiteloos door de regels komt, maar de opvolgende pakketjes hebben de header niet en dus zullen ze tegen gehouden worden.

Als je NAT gebruikt zullen de fragmenten aan elkaar gelijmd worden voordat ze door de reeksen gaan, dus hoef je je over fragmenten geen zorgen te maken.

Is dat niet het geval, dan is het belangrijk om te weten hoe fragmenten behandeld worden door de regels. Als een regel vraagt om bepaalde informatie, kan een fragment die niet verschaffen (het heeft tenslotte geen headers) en dus zal de regel niet gelden voor dit pakketje. Het eerste pakketje zal dus waarschijnlijk door de reeks komen, maar op de andere pakketjes wordt het beleid uitgevoerd.

Om dit probleem te omzeilen kan je fragments doorlaten of tegenhouden met de -f optie.

Het wordt als veilig beschouwd om fragmenten door te laten, hoewel er wel bugs in sommige systemen zitten die het laten crashen als er een los fragment op komt. Deze bugs zijn echter opgelost in moderne software.

De volgende regel zal alle fragmenten tegenhouden die naar 192.168.1.1 gaan:

```
# iptables -A OUTPUT -f -d 192.168.1.1 -j DROP
#
```

7.3.6 Het uitbreiden van iptables

`iptables` is makkelijk uit te breiden, wat nieuwe mogelijkheden met zich mee brengt. Sommige van deze mogelijkheden zijn standaard, andere worden slechts in enkele gevallen gebruikt. In het laatste geval worden ze meestal apart verstrekt voor de mensen die de extra mogelijkheid nodig hebben.

Uitbreidingen van de kernel gaan meestal in de kernel module directory, meestal /lib/modules/2.4.x/net. Als je kernel gecompileerd is met CONFIG_KMOD, dan worden ze geladen als dat nodig is en hoef je ze niet handmatig te laden.

Uitbreidingen van iptables zijn 'shared libraries', die meestal in de /usr/local/lib/iptables/ directory staan, hoewel sommige distributies deze in /usr/lib/iptables zetten.

Er zijn twee type uitbreidingen: nieuwe doelen en nieuwe regelopties. Sommige protocollen bieden je automatisch nieuwe regelopties. Deze zijn TCP, UDP en ICMP, die onder andere poorten kunnen specificeren.

Met de '-p' optie wordt automatisch een nieuwe uitbreiding geladen en om expliciet een uitbreiding te laden gebruik je de '-m' optie.

Om hulp te krijgen bij een uitbreiding, geef een optie om deze te laden ('-p', '-j' of '-m') en geef de optie '-h' of '–help':

```
# iptables -p tcp --help
#
```

TCP uitbreidingen De TCP uitbreidingen worden automatisch geladen als het TCP protocol wordt gekozen. De volgende opties kunnen dan gebruikt worden:

–tcp-flags

wordt gevolgd door twee lijsten met TCP markeringen. De eerste optie geeft de TCP markeringen (flags) aan die je wilt onderzoeken, de tweede geeft aan welke hiervan geactiveerd moeten zijn. Bijvoorbeeld:

```
# iptables -A INPUT --protocol tcp --tcp-flags ALL SYN,ACK -j DROP
```

Deze regel geeft aan dat de TCP pakketjes die de SYN en ACK markeringen hebben geactiveerd, maar de rest niet, tegengehouden moeten worden. 'ALL' is hetzelfde als 'SYN,ACK,FIN,RST,URG,PSH'. Wil je geen markeringen opgeven, dan kan je 'NONE' gebruiken.

–syn

Dit is hetzelfde als '–tcp-flags SYN,RST,ACK SYN'.

–source-port

kan gevolgd worden door een poortnummer of een poortbereik. Voor poortnummers kan je ook namen gebruiken, zoals die in /etc/services worden genoemd. Een poortbereik kan worden aangegeven door twee poorten, gescheiden met een dubbele punt, zoals 1000:2000. Wordt het eerste poortnummer weggelaten (':2000') dan geldt deze regel voor alle poorten tot 2000. Wordt het laatste poortnummer weggelaten ('1000:') dan geldt deze regel voor de poort 1000 en verder.

–sport

is hetzelfde als '–source-port'.

–destination-port

en

–dport

zijn hetzelfde als de bovenstaande, alleen geven ze de poorten van de doelbestemming aan in plaats van de herkomst.

–tcp-option

wordt gevold door een nummer. Deze regel geldt voor TCP pakketjes met de markering voor dit nummer. Als het pakketje een niet volledige header heeft, dan wordt het pakketje automatisch tegengehouden.

Een uitleg van TCP markeringen Soms is het handig om TCP connecties maar één kant op te laten werken; je wilt wel connecties maken naar buiten, maar niet andersom.

De oplossing hiervoor is om de pakketjes tegen te houden die een connectie willen openen. Deze pakketjes heten **SYN** pakketjes (het zijn eigenlijk pakketjes met de SYN markering geactiveerd en de RST en ACK markeringen niet geactiveerd). Door deze pakketjes tegen te houden, wordt er geen connectie gemaakt en zullen dus opvolgende pakketjes genegeerd worden.

De '–syn' optie wordt hiervoor gebruikt. Deze is alleen geldig voor het TCP protocol. Om een regel te maken die geldt voor de pakketjes die een connectie willen maken vanaf 192.168.1.1 kan je de volgende (niet complete) regel gebruiken:

```
-p TCP -s 192.168.1.1 --syn
```

Natuurlijk kan ook deze optie weer voorafgegaan worden door een uitroepteken, om pakketjes te specificeren die geen connectie willen maken.

UDP Uitbreidingen Deze uitbreidingen worden automatisch geladen als er een regel is die het UDP protocol gebruikt (dus als er ergens in de regel '-p udp' voorkomt). Deze uitbreiding brengt de opties '–source-port', '–sport', '–destination-port' en '–dport' met zich mee en deze zijn identiek aan die van TCP.

ICMP Extensions Deze uitbreiding wordt automatisch geladen als het ICMP protocol gebruikt wordt en komt met slechts één extra optie:

–icmp-type

wordt gevolgd door een ICMP type naam (zoals 'host-unreachable'), door een nummer wat staat voor een type, of door twee nummers gescheiden door een '/'. De twee nummers staan voor het type en de code van het ICMP pakketje. De nummers kan je opzoeken door '-p icmp –help' te gebruiken.

Andere optie uitbreidingen Deze uitbreidingen kunnen worden gebruikt door de '-m' optie te gebruiken.

mac

Deze module kan gebruikt worden om het hardware adres van de netwerkkaart te gebruiken in je regels. Het werkt alleen met het hardware adres van de kaart waarmee de pakketjes in de computer komen. Het heeft één optie:

–mac-source

gevolgd door een hardware adres, zoals in '–mac-source 00:60:08:91:CC:B7'.

limit

Deze module wordt gebruikt om het aantal geldende regels in een bepaalde tijd terug te dringen. Zo kan je bijvoorbeeld ervoor zorgen dat er slechts 3 keer per uur een berichtje in je log komt te staan als er de hele tijd geldende pakketjes komen. Het specificeerd twee opties:

–limit

wordt gevolgd door een nummer en het geeft het aantal pakketjes waar deze regel voor geldt, in een bepaald tijdsbestek. Het nummer kan ook een tijd aangeven, door gebruik te maken van '/second', '/minute', '/hour' of '/day' of de eerste letter ervan. '5/second' is dus hetzelfde als '5/s'.

–limit-burst

wordt gevolgd door een nummer en geeft de maximale grens voordat de bovenstaande optie gaat gelden.

Om te kijken hoe het werkt kijken we naar de onderstaande regel, die de pakketjes door de FORWARD reeks logt, maar alleen als ze door de 'limit' module heen komen.

```
# iptables -A FORWARD -m limit -j LOG
```

De standaard 'limit-burst' is vijf en daardoor worden de eerste vijf pakketjes gelogd. Hierna duurt het twintig minuten voordat het volgende pakketje gelogd wordt. Als er 20 minuten lang geen pakketje komt, dan kunnen er na nog eens 20 minuten twee pakketjes doorheen. Na 100 minuten kunnen er dus weer 5 pakketjes doorheen.

NB: Je kan niet een regel maken met een tijdsbestek van meer dan 59 uur.

Je kan deze regel ook gebruiken om DoS (Denial of Service) aanvallen tegen te gaan, door een overvloed aan pakketjes tegen te houden.

Syn-flood bescherming:

```
# iptables -A FORWARD -p tcp --syn -m limit --limit 1/s -j ACCEPT
```

Furtive port scanner:

```
# iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit --limit 1/s -j ACCEPT
```

Ping of death:

```
# iptables -A FORWARD -p icmp --icmp-type echo-request -m limit --limit 1/s -j ACCEPT
```

Hoe dit werkt laat de volgende grafiek zien:

```
   snelheid (pkt/s)
             ^        .---.
             |       / DoS \
             |      /       \
Rand van DoS-|.....:.........\.......................
 = (limit *  |    /:          \
limit-burst) |   / :           \         .-.
             |  /  :            \       /   \
             | /   :             \     /     \
Eide van DoS-|/....:..............:.../.......\..../.
 = limit     |     :              :`-'         `--'
-------------+-----+--------------+------------------> tijd (s)
RESULTAAT=>  Geldt |  Geldt niet  |    Geldt
```

De volgende grafiek is van een grens van één pakketje per seconde, met een 'burst' van vijf pakketjes. Pakketjes komen binnen met vier per seconde, drie seconden lang. Vervolgens nog eens na weer drie seconden:

```
        <--Flood 1-->           <---Flood 2--->

Total  ^                   Line  __--      YNNN
Packets|               Rate  __--      YNNN
       |            mum  __--      YNNN
    10 |        Maxi __--         Y
       |         __--            Y
       |     __--               Y
       | __--    YNNN
       |-    YNNN
     5 |    Y
       |   Y                                Key:  Y -> Matched Rule
       |  Y                                       N -> Didn't Match Rule
       | Y
       |Y
     0 +-------------------------------------------------->  Time (seconds)
        0   1   2   3   4   5   6   7   8   9  10  11  12
```

Je ziet hier dat de eerste vijf pakketjes sneller binnenkomen dan één per seconde en ook doorgelaten worden. Dan pas begint de grens te gelden. De volgende pakketjes zullen dus slechts met één pakketje per seconde doorgelaten worden. Na de pauze kunnen er weer meer pakketjes doorgelaten worden, totdat de grens weer gaat gelden.

owner

Deze module zorgt ervoor dat je de eigenaar van het pakketje kan laten meetellen in een regel. Het werkt alleen in de OUTPUT reeks, en sommige pakketjes hebben geen eigenaar.

–uid-owner userid

Geldt als het pakketje gemaakt werd door deze gebruiker.

–gid-owner groupid

Geldt als het pakketje gemaakt werd door een gebruiker in deze groep.

–pid-owner processid

Geldt als het pakketje gemaakt werd door dit programma.

–sid-owner sessionid

Geldt als het pakketje gemaakt werd door een programma in een sessie groep.

unclean

Deze module kijkt of de pakketjes wel geldig zijn en verschaft geen extra opties. Het zou niet gebruikt moeten worden om de veiligheid van een computer te verhogen.

De 'state' uitbreiding De 'state' uitbreiding stelt je in staat om regels te maken op basis van wat van een pakketje bekend is in verband met connectie tracking. Connectie tracking wordt gerealiseerd door de 'ip_conntrack' module. Deze uitbreiding is makkelijk als je wilt weten of een pakketje deel uitmaakt van een bestaande connectie of niet. Het wordt aangeraden deze optie te gebruiken, zodat je geen pakketjes doorlaat die een ander doel hebben dan een connectie te openen.

Je gebruikt deze uitbreiding met de optie '-m state' en het levert een extra '–state' optie, die gevolgd wordt door een reeks toestanden (states) van de pakketjes, gescheiden door komma's. De toestanden die beschikbaar zijn:

NEW

Dit pakketje maakt een nieuwe verbinding.

ESTABLISHED

Dit pakketje behoort tot een bestaande verbinding (d.w.z. een antwoord-pakketje, of een pakketje op een connectie waar al verkeer is geweest).

RELATED

Een pakketje wat te maken heeft, maar niet deel uitmaakt van een bestaande connectie. Zulke pakketjes zijn ICMP error pakketjes of pakketjes deel uitmakend van een FTP connectie.

INVALID

Van dit pakketje kon niet uitgemaakt worden waar het bij hoort. Dit kan voorkomen als je geen vrij geheugen meer hebt of als je ICMP error pakketjes krijgt die niets te maken hebben met een bestaande connectie. Zulke pakketjes kunnen normaal gesproken tegen gehouden worden.

7.4 Doel specificaties

Nu we weten hoe we een pakketje kunnen specificeren, kunnen we iets doen met zo'n pakketje. Wat er moet gebeuren met een pakketje, heet het doel (target).

Er zijn twee simpele doelen al ingebouwd: DROP en ACCEPT. Met DROP wordt het pakketje tegengehouden. Met ACCEPT wordt het pakketje gewoon doorgelaten. Als een pakketje een doel heeft bereikt (DROP, ACCEPT of een ander doel) doorloopt het de resterende regels niet meer maar wordt het doel gelijk afgehandeld.

Er zijn twee soorten doelen behalve de bovenstaande: uitbreidingen en gebruikersreeksen.

7.4.1 Gebruikersreeksen

Een van de kenmerken waar iptables zijn kracht vandaan haalt is dat het de gebruiker in staat stelt om zijn eigen reeksen te maken. Behalve de ingebouwde reeksen (INPUT, FORWARD en OUTPUT) kan je dus nog meer reeksen maken, die naar elkaar kunnen verwijzen en die ook weer regels bevatten. Meestal zijn de namen van gebruikersreeksen in kleine letters, om ze te kunnen onderscheiden van de ingebouwde regels.

Als een pakketje overeenkomt met een bepaalde regel en die regel verwijst naar een andere reeks, dan wordt die reeks doorlopen. Bepaald die reeks niet wat er met het pakketje gebeuren moet, dan wordt de volgende regel in de vorige reeks uitgevoerd.

In de mooie ASCII art tekening zie je twee reeksen: `INPUT` en `test`.

```
 ‘INPUT’                          ‘test’
-----------------------------    ----------------------------
| Regel1: -p ICMP -j DROP   |    | Regel1: -s 192.168.1.1    |
|---------------------------|    |---------------------------|
| Regel2: -p TCP -j test    |    | Regel2: -d 192.168.1.1    |
|---------------------------|    -----------------------------
| Regel3: -p UDP -j DROP    |
-----------------------------
```

Zoals je ziet verwijst regel 2 naar de reeks ‘test’. Stel dat er nu een pakketje binnenkomt vanaf 192.168.1.1, dat gaat naar 1.2.3.4. Het komt binnen in de `INPUT` reeks en er wordt naar de eerste regel gekeken. Aangezien dit pakketje geen ICMP pakketje is, wordt de tweede regel bekeken. Deze geldt wel voor dit pakketje, dus wordt het doel uitgevoerd: reeks `test` wordt uitgevoerd op dit pakketje. Regel 1 geldt voor dit pakketje, maar er is hier geen doel opgegeven. Er gebeurt dus niets met dit pakketje. Regel 2 geldt niet en we zijn aan het einde van deze reeks. Er wordt meer verder gegaan bij regel 3 van de `INPUT` reeks.

De weg van het pakketje is dus als volgt:

```
                         v    ___________________________
 ‘INPUT’                 |   /    ‘test’                 v
-------------------------|--/    ------------------------|----
| Regel1                 | /|    | Regel1                |   |
|------------------------|/-|    |-----------------------|---|
| Regel2                 /  |    | Regel2                |   |
|---------------------------|    ------------------------v----
| Regel3                 /--+____________________________/
-------------------------|---
                         v
```

Gebruikersreeksen kunnen verwijzen naar andere gebruikersreeksen. Als je echter heen en weer verwijst kan het voorkomen dat je pakketjes in een lus komen. Dit vergt veel processortijd en je pakketjes worden tegengehouden.

7.4.2 Uitbreidingen: Nieuwe doelen

Met uitbreidingen kan je nieuwe doelen opgeven. Een doel-uitbreiding bestaat uit een kernel module en eventueel een `iptables` uitbreiding. De volgende uitbreidingen zijn beschikbaar bij netfilter:

LOG

Met deze uitbreiding kan je pakketjes loggen in je syslog. Het biedt de volgende opties:

–log-level

wordt gevolgd door een nummer of een naam, zoals 'debug', 'info', 'notice', 'warning', 'err', 'crit', 'alert' of 'emerg'. Deze namen komen overeen met de nummers 7 tot en met 0. De handleiding van syslog.conf geeft aan wat deze namen betekenen.

–log-prefix

wordt gevolgd door een regel van maximaal 29 letters. Deze regel komt vóór de eigenlijke boodschap in het logbestand, zodat je de boodschap makkelijk kan terugvinden.

Het wordt aangeraden om de limit-uitbreiding te gebruiken in combinatie met deze module, zodat je niet je log (en misschien je harde schijf) vol zet als je opeens veel pakketjes krijgt.

REJECT

Deze module werkt hetzelfde als 'DROP', alleen wordt de afzender op de hoogte gesteld dat het pakketje is tegengehouden door middel van een 'port unreachable' (onbereikbaar) ICMP pakketje. Zo'n pakketje wordt niet verzonden als

- Het pakketje wat gefilterd werd een ICMP error pakketje was, of een onbekend ICMP pakketje.
- Het pakketje een fragment was.
- Er al teveel ICMP error pakketjes zijn verzonden.

REJECT heeft ook een extra optie '–reject-with' waarmee je kan bepalen welk pakketje teruggestuurd moet worden. Zie de handleiding voor meer informatie hierover.

7.4.3 Speciale ingebouwde doelen

Er zijn twee ingebouwde doelen die een speciale functie hebben: `RETURN` en `QUEUE`.

`RETURN` beeindigd een gebruikersreeks. Het heeft hetzelfde effect als aan het einde van de reeks komen, namelijk dat naar de oorspronkelijke reeks wordt teruggekeerd en dat de volgende regel uitgevoerd wordt. Als dit in een ingebouwde reeks wordt gebruikt, wordt het beleid van die reeks uitgevoerd op het pakketje.

`QUEUE` bewaart het pakketje om later door een programma verwerkt te worden. Dit is alleen nuttig als:

is a special target, which queues the packet for userspace processing. For this to be useful, two further components are required:

- er een "queue handler"(afhandelaar) is, die het pakketje van de kernel aanpakt.
- er een programma is wat iets met het pakketje doet.

De queue-handler voor IPv4 is de ip_queue kernel module, die nog experimenteel is.

Als er geen programma is wat op de pakketjes wacht, dan worden ze tegengehouden. Om zo'n programma te schrijven kan je de libipq API gebruiken. Het bestand `/proc/net/ip_queue` geeft de status van ip_queue aan en het bestand `/proc/sys/net/ipv4/ip_queue_maxlen` geeft de maximale lengte van de queue (normaal 1024).

7.5 Reeksen onderhouden

Als je veel regels hebt kan het wel eens een chaos worden als je ze allemaal in één reeks zet. Daarom is het handig om zelf reeksen te maken. Je kan je eigen reeksen noemen zoals je wilt, maar aangeraden wordt kleine letters te gebruiken om ze te kunnen onderscheiden van ingebouwde reeksen, die namen in hoofdletters hebben. De namen van reeksen kunnen niet langer zijn dan 31 letters.

7.5.1 Een nieuwe reeks maken

Hier volgt een voorbeeld van hoe je een reeks kan maken met een hele originele naam: `test`:

```
# iptables -N test
#
```

Dat is alles. Nu heb je een reeks zonder regels erin, die `test` heet.

7.5.2 Een reeks verwijderen

Een reeks verwijderen is ook simpel, maar kan alleen als de reeks helemaal geen regels meer bevat. Er moeten ook geen verwijzingen meer zijn naar de reeks die je wilt verwijderen. Je verwijdert een regel met '-X' of '–delete-chain':

```
# iptables -X test
#
```

Als je de naam van de reeks weg laat, worden alle gebruikersreeksen verwijdert.

7.5.3 Een reeks leeg maken

Er is een simpele manier om alle regels uit een reeks te verwijderen, namelijk met de '-F' of '–flush' optie:

```
# iptables -F FORWARD
#
```

Als je geen reeks opgeeft, worden alle reeksen leeg gemaakt.

7.5.4 De regels in een reeks bekijken

Om alle regels in een reeks te bekijken gebruik je de '-L' optie, of de '–list' optie.

Bij gebruikersreeksen zie je een 'refcnt' staan. Dit is het aantal verwijzingen naar deze reeks. Dit moet nul zijn voordat de reeks kan worden verwijdert.

Je kan nog drie extra opties gebruiken naast '-L'. Met de '-n' optie kan je ervoor zorgen dat IP adressen niet omgezet worden naar domeinnamen. Dit is nuttig als je DNS niet goed werkt. Het opzoeken van namen zorgt dan namelijk voor grote vertragingen. Poorten worden ook weergegeven als nummers ('80') in plaats van namen ('www').

Met de '-v' optie krijg je alle pikante details te zien, zoals de interfaces en pakket tellers.

The pakket en byte tellers gebruiken achtervoegsels zoals 'K' (kilo) en 'M' (mega) voor 1.000 en 1.000.000. Gebruik de '-x' optie als je de hele cijfers wilt weten, zonder achtervoegsels.

7.5.5 Tellers op nul zetten

Je kan een teller op nul zetten met de '-Z' of '–zero' optie.

Als je het volgende doet:

```
# iptables -L FORWARD
# iptables -Z FORWARD
#
```

Kunnen er nog pakketjes doorkomen in de tijd dat je het tweede commando aan het typen bent. Om dit te voorkomen kan je de '-L' en '-Z' opties tegelijk gebruiken, om de tellers tegelijk te lezen en op nul te zetten.

7.5.6 Het beleid instellen

Als een pakketje aan het einde van een reeks komt, wordt het beleid uitgevoerd. Alleen ingebouwde reeksen (`INPUT`, `OUTPUT` en `FORWARD`) hebben een beleid.

Het beleid kan `ACCEPT` (doorlaten) of `DROP` (tegenhouden) zijn.

```
# iptables -P FORWARD DROP
#
```

Dit voorbeeld stelt het beleid van de reeks `FORWARD` in op `DROP`.

8 Ipchains of ipfwadm gebruiken

Twee modules, ipchains.o en ipfwadm.o, stellen je in staat om je oude firewall regels te gebruiken. Laad één van deze twee als module in je kernel en je kan gewoon ipchains of ipfwadm gebruiken.

Je kan slechts één van de drie modules laden die uitmaken welke regels je gebruikt: ipfwadm.o, ipchains.o of iptables.o. Laad er niet meer dan één.

Ipfwadm en ipchains worden natuurlijk niet eeuwig ondersteund. Er komt een tijd dat je iptables wel moet gebruiken. Volgens Rusty hoef je je geen zorgen te maken tot 2004 als je ipfwadm of ipchains gebruikt.

9 NAT en een pakket filter tegelijk gebruiken

NAT staat voor Network Address Translation en het stelt je in staat om pakketjes een ander doel of bestemming te geven. Meer details hierover vind je in de NAT HOWTO. Je kan NAT en een pakket filter goed combineren.

Als je een pakket filter opzet, kan je gewoon je NAT negeren: de bestemming van je regel is de 'echte' bestemming en wordt niet beïnvloed door NAT. Als je bijvoorbeeld je NAT zo hebt ingesteld dat pakketjes die naar 1.2.3.4 poort 80 gaan, naar 10.1.1.1 poort 8080 gestuurd worden, ziet je pakket filter de pakketjes naar 10.1.1.1 poort 8080 gaan en niet naar 1.2.3.4. De pakketjes gaan tenslotte niet naar 1.2.3.4, dus heeft het ook geen zin hier regels voor te maken.

Bij masquerading (SNAT) komen de pakketjes dus van hun interne netwerkadres (192.168.1.1) en gaan naar een extern adres (www.linux.org). Ingaande pakketjes worden eerst omgeschreven, dan gefilterd, uitgaande pakketjes worden eerst gefilterd, dan omgeschreven.

Je kan de 'state' uitbreiding gebruiken zonder dat het extra moeite kost voor het pakket filter, want die moet het toch al bijhouden om NAT te kunnen doen. Om inkomende connecties tegen te houden en masquerading te gebruiken kan je het volgende doen (dit staat ook deels in de NAT howto):

```
# Masquerade uitgaand ppp0
iptables -t nat -A POSTROUTING -o ppp0 -j MASQUERADE

# Houd NEW en INVALID inkomende of doorgestuurde connecties tegen.
iptables -A INPUT -i ppp0 -m state --state NEW,INVALID -j DROP
iptables -A FORWARD -i ppp0 -m state --state NEW,INVALID -j DROP

# IP forwarding aanzetten
echo 1 > /proc/sys/net/ipv4/ip_forward
```

10 Verschillen tussen ipchains en iptables

- De namen van de ingebouwde reeksen hebben nu namen in hoofdletters. Ze heten dus geen 'input', 'forward' en 'output' meer maar 'INPUT', 'FORWARD' en 'OUTPUT'. De INPUT en OUTPUT reeksen krijgen respectievelijk alleen pakketjes die voor deze computer bedoeld zijn of die door deze computer gemaakt zijn. Bij ipchains kwam al het verkeer door deze reeksen.
- De '-i' optie betekend nu de interface van het inkomende verkeer, en werkt alleen in de INPUT en FORWARD reeksen. Sommige regels in de FORWARD reeks en alle regels in de OUTPUT reeks moeten aangepast worden: '-i' moet vervangen worden door '-o', de optie voor de uitgaande interface.
- De TCP en UDP opties om een poort te specificeren zijn nu –source-port of –sport (of –destination-port/–dport) en deze moeten na '-p tcp' of '-p udp' komen, omdat deze opties de TCP en UDP uitbreidingen laden.
- De TCP '-y' optie is veranderd naar '–syn' en moet na '-p tcp' komen.
- Het DENY doel is veranderd naar DROP.
- Je kan reeksen leegmaken terwijl je de informatie ervan bekijkt.
- Het leegmaken van ingebouwde reeksen leegt nu ook de beleidstellers.
- FIXME: Listing chains gives you the counters as an atomic snapshot.
- REJECT en LOG zijn nu uitbreidingen, wat betekend dat ze apart geladen moeten worden.
- Reeksen kunnen namen hebben tot 31 tekens.
- MASQ is veranderd in MASQUERADE en werkt ook anders. REDIRECT heeft dezelfde naam gehouden maar werkt ook anders. Meer informatie hierover kan je vinden in de NAT HOWTO.

- De '-o' optie betekend nu de uitgaande interface in plaats van pakketjes naar 'userspace' te sturen. Pakketjes worden nu naar 'userspace' gestuurd via het QUEUE doel.
- Duizenden dingen meer.

11 Pakket filter advies

Het is vaak een goed idee voor maximale veiligheid om alles tegen te houden, en dan selectief sommige dingen door te laten. Iets onbekends wordt dan tegengehouden.

Draai geen daemons, servers en services als je ze niet nodig hebt, zelfs als je denkt dat je ze geblockt hebt.

Het kan geen kwaad veilige dingen te combineren. Je kan beter teveel veiligheid hebben dan te weinig. Gebruik tcp-wrappers, proxies, route verificatie en een pakket filter. Route verificatie is pakketjes blokken die een IP hebben op een verkeerde interface. Als je interne netwerk addressen heeft als 10.1.1.1 en er komt een pakketje vanaf IP 10.1.1.2 in je computer via je externe interface, kan je hem beter tegenhouden. Doe je dat niet, dan denkt je computer dat het een pakketje van je netwerk is, dus van mensen die vertrouwd kunnen worden. Dit kan als volgt ingesteld worden:

```
# echo 1 > /proc/sys/net/ipv4/conf/ppp0/rp_filter
#
```

Of voor alle interfaces:

```
# for f in /proc/sys/net/ipv4/conf/*/rp_filter; do
#     echo 1 > $f
# done
#
```

Debian doet dit standaard al, maar bij sommige routers moet dit uitgezet worden omdat ze pakketjes uit rare richtingen horen te krijgen.

Logging is nuttig om meer informatie te krijgen, maar als je firewall hevig gebruikt wordt kan je log wel eens overvol raken. Dit kan zelfs resulteren tot een DoS situatie. Om dit te voorkomen kan je 'limit' gebruiken, zie hierboven.

Rusty raad dringend aan om connectie tracking in te stellen. Het maakt de boel iets langzamer, maar het is erg nuttig om de toegang tot je netwerk te bepalen. Als je connectie tracking niet hebt ingebouwd moet je de module 'ip_conntrack.o' laden. Als je ingewikkelde protocolen wilt tracken, dan moet je hulp modules laden, zoals 'ip_conntrack_ftp.o'. Hier een voorbeeldje van wat je kan doen met connectie tracking:

```
# iptables -N no-conns-from-ppp0
# iptables -A no-conns-from-ppp0 -m state --state ESTABLISHED,RELATED -j ACCEPT
# iptables -A no-conns-from-ppp0 -m state --state NEW -i ! ppp0 -j ACCEPT
# iptables -A no-conns-from-ppp0 -i ppp0 -m limit -j LOG --log-prefix "Bad packet from ppp0:"
# iptables -A no-conns-from-ppp0 -i ! ppp0 -m limit -j LOG --log-prefix "Bad packet not from ppp0:"
# iptables -A no-conns-from-ppp0 -j DROP

# iptables -A INPUT -j no-conns-from-ppp0
# iptables -A FORWARD -j no-conns-from-ppp0
```

Een goede firewall bouwen is niet het onderwerp van deze HOWTO. Zie daarvoor de Security HOWTO. Een goede tip is altijd zo min mogelijk door te laten.