LinuxFocus article number 289
http://linuxfocus.org

Het filteren van pakketten met Linux

door Vincent Renardias
<vincent(at)renardias.com>

Over de auteur:

Vincent Renardias, een GNU/Linux gebruiker sinds 1993, begon in 1996 deel te nemen aan de ontwikkeling: hij is Debian ontwikkelaar, vertaler naar het Frans van de Gimp en de GNOME desktop, oprichter van de Linux gebruikersgroep in Marseille (PLUG)... Nu, terwijl hij R&D manager van het EFB2 bedrijf is, blijft hij zijn steentje bijdragen tot GNU/Linux.

Vertaald naar het Nederlands door:
Samuel Derous
<cyberprophet/at/linux.be>

Kort:

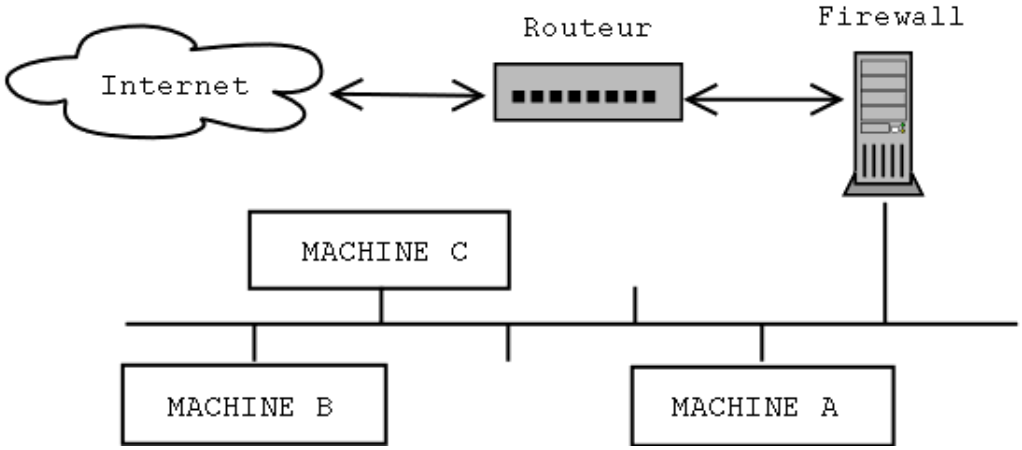
Dit artikel werd eerder gepubliceerd in een Speciale uitgave over beveiliging in een Frans Linux tijdschrift. De uitgevers, de schrijvers en de vertalers, waren zo vriendelijk om LinuxFocus toestemming te verlenen elk artikel uit dit speciaal nummer te publiceren. LinuxFocus zal ze tot jullie brengen meteen als ze naar het Engels vertaald zijn. Dank aan allen die bij dit werk betrokken zijn. Deze korte inleiding zal voor elk artikel die uit deze speciale uitgave komt, opnieuw gebruikt worden.

Een goede manier om indringers buiten te sluiten is om wat onbruikbaar is in een netwerk te filteren. Deze taak wordt over het algemeen toebedeeld aan een machine die fungeert als firewall. In dit artikel zullen we je inleiden in de vereiste kennis voor het implementeren en het configureren van zo'n systeem.

Gateway, Proxy-Arp of Ethernet Bridge ?

Het filtermechanisme kan gezien worden als een net dat ongewilde pakketten tegenhoudt. Het belangrijkste bestaat erin de juiste grootte voor de strik en de juiste plaats voor het installeren te vinden.

Firewall locatie in het netwerk



Om pakketten behoorlijk te filteren is het nodig het filtermechanisme fysiek tussen het netwerk dat het beschermt en de "rest van de wereld" te plaatsen. Praktisch gezien wordt dit verwezenlijkt met een machine die twee netwerkinterfaces (gewoonlijk ethernet) bevat, één is verbonden met het interne netwerk, en de ander met de router die zorgt voor de toegang met de buitenwereld. Op deze manier dient de communicatie door de firewall te gaan die ze, naar gelang hun inhoud, zal blokkeren of niet. De machine die het filtermechanisme zal bevatten kan op drie verschillende manieren geconfigureerd worden.

- "Simple" gateway: dit is de meest algemene configuratie. De machine wordt als een gateway tussen twee netwerken of subnetwerken gebruikt. De computers in het lokale netwerk zullen geconfigureerd worden om gebruik te maken van de firewall in plaats van de routers als hun standaard route naar buiten.

- "Proxy-ARP" gateway: De voorgaande configuratie gaat ervan uit dat het netwerk in twee subnetwerken is verdeeld, wat ervoor zorgt dat de helft van de beschikbare IP adressen verloren gaan. Dat is vervelend. In het voorbeeld van een subnetwerk van 16 adressen (met een 28 bit netmask), zijn er maar 14 beschikbaar, daar de netwerk- en de broadcastadressen beide in gebruik zijn. Door maar 1 bit aan het subnetmask toe te voegen, daalt het aantal van 14 naar maar 6 beschikbare adressen (8 IP’s min degene die voor het netwerk en de broadcast gebruikt worden). Als je je het niet kan veroorloven om de helft van de beschikbare IP’s te verliezen, kan je gebruik maken van deze techniek die wat later in dit artikel uitgelegd zal worden. Verder dient nog gezegd te worden dat deze techniek geen veranderingen in de configuratie van de bestaande machines, noch van de router, noch van de beschermde computers, vereist.

- Ethernet bridge: door het installeren van een Ethernet Gateway (geen IP gateway) wordt het filtreer-mechanisme voor andere machines onzichtbaar. Zo’n configuratie kan zonder het toewijzen van IP adressen aan de Ethernet interfaces verwezenlijkt worden. De machine is dan niet langer traceerbaar met ping, traceroute, etc... Laten we hierbij wel opmerken dat een pakket filtreerimplementatie in zo’n configuratie een 2.2.x kernel vereist daar de port van dit kenmerk voor de 2.4.x kernels nog niet afgewerkt is.

Basisregels bij het filteren

Nu we weten waar we onze filter dienen te installeren, moeten we nog definiëren wat het moet blokkeren, en wat het moet accepteren.
Er zijn twee manieren om zo'n filter te configureren:

- de goede manier: geen enkel pakket behalve die welke expliciet zijn toegestaan worden doorgelaten
- de slechte manier: (helaas wordt deze manier vaak gebruikt) alle pakketten die nadrukkelijk verboden werden, worden geblokkeerd, alle andere worden toegelaten.

Het is eenvoudig uit te leggen: ten eerste, het vergeten van een regel kan resulteren in een service die slecht of helemaal niet werkt. Gewoonlijk wordt dit snel opgemerkt en wordt dit simpelweg opgelost door de juiste regel toe te voegen om alles terug aan te praat te krijgen.
Ten tweede, het vergeten van een regel kan in een potentieel zwak punt resulteren, die moeilijker te vinden is... als we dat zwakke punt al vinden.

Netfilter

De meest gebruikte software voor pakketfiltering met Linux 2.4 is netfilter; het vervangt keurig 'ipchains' die met de linux kernel 2.2 gebruikt werd. Netfilter bestaat uit twee delen: de ondersteuning voor de kernel die in de kernel gecompileerd dient te worden, en het 'iptables' commando dat op je systeem beschikbaar zou moeten zijn.

Voorbeeld van setup

Daar een voorbeeld dat voorzien is van de nodige commentaar beter is dan een lang theoretisch relaas, zullen we hier beschrijven hoe je een filtreermechanisme dient te installeren en op te zetten. Eerst zullen we de machine configureren als een gateway door gebruik te maken van Proxy-ARP om het aantal IP adressen te reduceren, daarna zullen we het filtersysteem opzetten.

De auteur heeft een een voorkeur voor de Debian distributie om zo'n systeem op te zetten, maar met elke andere disttibutie zou het ook moeten gaan.

Controleer eerst of je kernel Netfilter ondersteunt. Als de kernel dit doet, zouden de opstartberichten de volgende regels moeten bevatten:

```
ip_conntrack (4095 buckets, 32760 max)
ip_tables: (c)2000 Netfilter core team
```

Anders zul je de kernel opnieuw moeten compileren nadat je de Netfilter ondersteuning hebt geactiveerd. De bijhorende opties kunnen gevonden worden in het "Network Packet filtering" submenu van het "Networking Options" menu. Vanuit de "Netfilter Configuratie" sectie dien je de opties die je nodig hebt te selecteren. Als je twijfelt kun je ze altijd allemaal selecteren. Verder is het beter om

Netfilter in de kernel bij te sluiten, dan gebruik te maken van modules. Als om één of andere reden één van de Netfilter modules niet zou worden geladen, of niet te vinden zou zijn, zou filtering niet werken, en laten we het maar niet hebben over de risico's dat dat zou kunnen inhouden.

Je zou ook het 'iproute2' pakket moeten installeren (Dit is niet vereist, maar ons voorbeeld zal het gebruiken omdat het ons toelaat het configuratiescript simpeler te maken). Met Debian is het voldoende om de 'apt-get install iproute' commando in te typen.
Bij andere distro's dien je het bijhorende pakket op te halen. Eventueel kun je de software vanuit broncode te installeren, welke je kunt downloaden van het volgende adres:
ftp://ftp.inr.ac.ru/ip-routing/

Nu moeten de twee ethernetkaarten geconfigureerd worden. We moeten hier opmerken dat de linuxkernel, wanneer hij automatisch naar netwerkkaarten op zoek gaat, meteen stopt wanneer hij er één gevonden heeft. Op deze manier wordt alleen de eerste kaart gevonden.
Een simpele oplossing voor dit probleem bestaat erin om de volgende regel aan het lilo.conf bestand toe te voegen:
append="ether=0,0,eth1"

Nu moeten we de Ethernet interfaces configureren. De methode die we toepassen laat ons toe om hetzelfde ip-adres te gebruiken voor beide kaarten, zodat we één adres uitsparen.
Laten we aannemen dat we een 10.1.2.96/28 subnetwerk hebben, dat zijn adressen starten vanaf 10.1.2.96 en dat 10.1.2.111 erbij hoort. De router zal het adres 10.1.2.97 adres hebben en ons filtreermachine het adres 10.1.2.98. De eth0 interface zal met de router verbonden zijn door een cross-connect RJ45 kabel als beide kaarten direct met elkaar verbonden zijn, zonder gebruik te maken van een hub of een switch; de eth1 interface zal verbonden worden met de hub/switch en vandaaruit verbonden worden met de lokale netwerkmachines.

Hierdoor zullen beide interfaces met de volgende parameters geconfigureerd worden:

```
adres address  : 10.1.2.98
netmask  : 255.255.255.240
netwerk  : 10.1.2.96
broadcast: 10.1.2.111
gateway  : 10.1.2.97
```

Hierna dienen we gebruik te maken van het volgende script dat opgestart moet worden na de initiële configuratie van de netwerkkaarten om de setup af te sluiten.

```
net.vars: configuration variables
PREFIX=10.1.2
DMZ_ADDR=$PREFIX.96/28
# Interface definities
BAD_IFACE=eth0
DMZ_IFACE=eth1
ROUTER=$PREFIX.97


net-config.sh: network configuration script

#!/bin/sh
# Haal het hekje van de volgende regel weg om de commando's bij het
# uitvoeren te tonen
# set -x
```

```
# We lezen de variabelen die in het voorgaande bestand werden gedefinieerd in
source /etc/init.d/net.vars

# We verwijderen de huidige routes van het lokale netwerk
ip route del $PREFIX.96/28 dev $BAD_IFACE
ip route del $PREFIX.96/28 dev $DMZ_IFACE

# We definiëren dat het lokale netwerk door eth1 bereikt kan worden
# en de router door eth0
ip route add $ROUTER dev $BAD_IFACE
ip route add $PREFIX.96/28 dev $DMZ_IFACE

# We activeren Proxy-ARP voor beide interfaces
echo 1 > /proc/sys/net/ipv4/conf/eth0/proxy_arp
echo 1 > /proc/sys/net/ipv4/conf/eth1/proxy_arp

# We activeren IP forwarding door toe te laten dat de pakketten op
# één kaart toekomen om daarna naar de andere kant geleid
# te worden door de router.
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Laten we teruggaan naar het Proxy-ARP mechanisme dat vereist is voor onze configuratie.
Wanneer een computer met een andere computer van hetzelfde netwerk communiceert, dient hij het ethernet adres (of MAC adres of hardware adres) dat met zijn IP adres overeenkomt, te weten. Dan zendt de bronmachine de vraag uit "wat is het MAC adres van de interface die het IP adres 1.2.3.4 heeft?", en de doelmachine moet antwoorden.

Hier is een voorbeeld van zo'n "gesprek", gemaakt met tcpdump:

- De vraag: de machine 172.16.6.72 vraagt het MAC adres dat met IP adres IP 172.16.6.10 correspondeert.

```
19:46:15.702516 arp who-has 172.16.6.10 tell 172.16.6.72
- Het antwoord: de machine 172.16.6.10 voorziet hem van zijn kaartnummer.
19:46:15.702747 arp reply 172.16.6.10 is-at 0:a0:4b:7:43:71
```

Dit leidt ons tot het slot van deze korte uitleg: Het ARP verzoek wordt verzorgd door broadcasting, maar gelimiteerd tot één fysiek netwerk. Daardoor zou het verzoek van een beveiligde machine om het MAC adres van de router te vinden, geblokkeerd moeten worden door de filtreermachine. Het activeren van de Proxy-ARP kenmerken zou dit probleem moeten oplossen, daar het het ARP verzoek zal doorzenden.

Op dit punt zou je over een werkend netwerk moeten beschikken met een machine die het volledige verkeer tussen het lokale netwerk en de buitenwereld regelt.

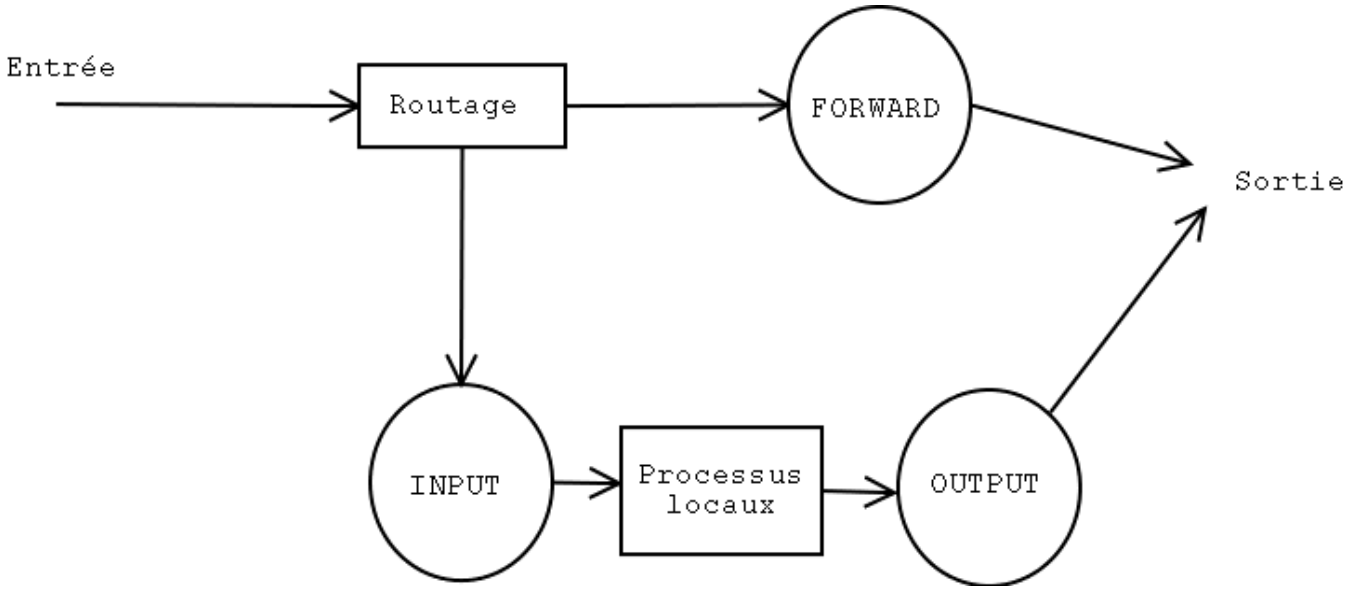
Nu dienen we de filtering op te zetten door gebruik te maken van Netfilter.
Netfilter laat toe om meteen op de pakketstroom te reageren. In de basisconfiguratie worden de pakketten door drie ketens van regels geregeld:
- INPUT: voor de pakketten die door een interface binnenkomen,
- FORWARD: voor alle pakketten die van één interface naar een andere dienen doorgezonden te worden
- OUTPUT: voor de pakketten die naar buiten gaan via een interface.

Het 'iptables' commando laat toe om in elke van deze ketens het filtergedrag aan te te passen, regels toe te voegen of te verwijderen.
Verder heeft elke keten een standaardwaarde, hiermee wil men zeggen dat het weet wat het dient te doen

wanneer geen enkele regel met een pakket in de keten overeenkomt.

Verder zijn de meest huidige opties:
- ACCEPT: het pakket wordt toegelaten
- REJECT: het pakket wordt geweigerd en het error pakket die hiermee verband houdt wordt verzonden (ICMP poort is onbereikbaar, TCP RESET, hangt van de zaak af).
- LOG: schrijft een pakket-opmerking in het syslog
- DROP: het pakket wordt genegeerd en geen antwoord wordt verzonden

Pakketstromen door standaard ketens



Hier volgen de hoofdopties van iptables die toelaten om gehele ketens te manipuleren. We zullen hen later gedetailleerder bespreken:

-N: Maakt een nieuwe keten aan.
-X: verwijdert een lege keten.
-P: verandert de default waarde van een keten.
-L: heeft een lijst van de regels in een keten weer.
-F: verwijdert alle regels in een keten.
-Z: Zet de bytes- en pakkettellers die door een keten zijn gegaan terug op nul.

Om een keten aan te passen zijn volgende commando's beschikbaar:
-A: voegt regels aan het einde van een keten toe
-I: Voegt een nieuwe regel op een gegeven positie in een keten in.
-R: Vervangt een gegeven regel in een keten.
-D: verwijdert een regel in een keten, ofwel door zijn nummer te gebruiken, of door het beschrijven van de regel.

Laten we nu eens een praktisch voorbeeld bekijken: We zullen de PING-antwoorden die van een gegeven machine komen, blokkeren (dat is de 'echo-reply' type ICMP pakketten).
Laten we eerst controleren of we de gegeven machine kunnen "pingen":

```
# ping -c 1 172.16.6.74
PING 172.16.6.74 (172.16.6.74): 56 data bytes
64 bytes from 172.16.6.74: icmp_seq=0 ttl=255 time=0.6 ms

--- 172.16.6.74 ping statistics ---

1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.6/0.6/0.6 ms
```

Nu voegen we een regel toe aan de INPUT keten die de ICMP-Reply zal onderscheppen ('-p icmp --icmp-type echo-reply') die van de machine 172.16.6.74 komen ('-s 172.16.6.74'). Deze pakketten zullen genegeerd worden ('-j DROP').

```
# iptables -A INPUT -s 172.16.6.74 -p icmp --icmp-type echo-reply -j DROP
```

Laten we de machine nogmaals PINGen:

```
# ping -c 3 172.16.6.74
PING 172.16.6.74 (172.16.6.74): 56 data bytes

--- 172.16.6.74 ping statistics ---

3 packets transmitted, 0 packets received, 100% packet loss
```

Zoals we konden verwachten worden de antwoorden niet toegelaten. We kunnen controleren dat de drie antwoorden geblokkeerd werden (3 pakketen van 252 bytes):

```
# iptables -L INPUT -v
Chain INPUT (policy ACCEPT 604K packets, 482M bytes)
 pkts bytes target     prot opt in     out     source               destination
   3   252   DROP       icmp --  any    any     172.16.6.74          anywhere
```

Om terug te keren naar de oorspronkelijke situatie dienen we enkel de eerste regel uit de INPUT keten te verwijderen:

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

Nu zou PING weer naar behoren moeten werken:

```
# ping -c 1 172.16.6.74
PING 172.16.6.74 (172.16.6.74): 56 data bytes
64 bytes from 172.16.6.74: icmp_seq=0 ttl=255 time=0.6 ms

--- 172.16.6.74 ping statistics ---

1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.6/0.6/0.6 ms
#
```

Het werkt!

Je kunt andere ketens aan de drie bestaande (die je toch niet kan verwijderen) toevoegen en ervoor zorgen dat er wat verkeer door hen wordt afgehandeld. Dit kan nuttig zijn om bijvoorbeeld te verhinderen dat regels in de verschillende ketens meerdere keren voorkomen.

Laten we nu de vereiste regels opzetten voor een minimale firewall. Het zal ssh, domain(DNS), http en smtp services toelaten en niets anders.
De setup commando's zijn geschreven in een shell script om de configuratie gemakkelijker te maken. De script begint met het verwijderen van de huidige configuratie voordat hij de nieuwe opzet. Deze kleine truc laat het script toe om uitgevoerd te worden wanneer de configuratie actief is zonder het risico te lopen dezelfde regels te krijgen.

rc.firewall

```
#!/bin/sh

# De regels verwijderen
iptables -F
iptables -F INPUT
iptables -F OUTPUT
iptables -F FORWARD


# De keten is gebouwd volgens de richting
# bad = eth0 (de slechte buitenwereld)
# dmz = eth1 (binnenkant)
iptables -X bad-dmz
iptables -N bad-dmz
iptables -X dmz-bad
iptables -N dmz-bad
iptables -X icmp-acc
iptables -N icmp-acc
iptables -X log-and-drop
iptables -N log-and-drop

# specifieke keten die gebruikt wordt voor het loggen van pakketten
# alvorens ze geblokkeerd worden
iptables -A log-and-drop -j LOG --log-prefix "drop "
iptables -A log-and-drop -j DROP

# De pakketten die de TCP vlaggen geactiveerd hebben, worden verwijderd
# hetzelfde voor de pakketten die helemaal geen vlaggen hebben
# (wordt vaak gebruikt met Nmap scans)
iptables -A FORWARD -p tcp --tcp-flags ALL ALL -j log-and-drop
iptables -A FORWARD -p tcp --tcp-flags ALL NONE -j log-and-drop

# De pakketten die van gereserveerde adressen komen worden weggelaten
# hetzelfde geldt voor multicast pakketten
iptables -A FORWARD -i eth+ -s 224.0.0.0/4 -j log-and-drop
iptables -A FORWARD -i eth+ -s 192.168.0.0/16 -j log-and-drop
iptables -A FORWARD -i eth+ -s 172.16.0.0/12 -j log-and-drop
iptables -A FORWARD -i eth+ -s 10.0.0.0/8 -j log-and-drop

# De pakketten die behoren tot een connectie die al tot stand is gebracht,
# worden geaccepteerd
iptables -A FORWARD -m state --state INVALID -j log-and-drop
iptables -A FORWARD -m state --state RELATED,ESTABLISHED -j ACCEPT

# De bijhorende keten wordt verzonden volgens de oorsprong van het pakket
iptables -A FORWARD -s $DMZ_ADDR -i $DMZ_IFACE -o $BAD_IFACE -j dmz-bad
iptables -A FORWARD -o $DMZ_IFACE -j bad-dmz

# Al de rest wordt genegeerd
iptables -A FORWARD -j log-and-drop
```

```
# geaccepteerde ICMP’s
iptables -A icmp-acc -p icmp --icmp-type destination-unreachable -j ACCEPT
iptables -A icmp-acc -p icmp --icmp-type source-quench -j ACCEPT
iptables -A icmp-acc -p icmp --icmp-type time-exceeded -j ACCEPT
iptables -A icmp-acc -p icmp --icmp-type echo-request -j ACCEPT
iptables -A icmp-acc -p icmp --icmp-type echo-reply -j ACCEPT
iptables -A icmp-acc -j log-and-drop

# buitenste -> binnenste keten
# mail, DNS, http(s) en SSH worden geaccepteerd
iptables -A bad-dmz -p tcp --dport smtp -j ACCEPT
iptables -A bad-dmz -p udp --dport domain -j ACCEPT
iptables -A bad-dmz -p tcp --dport domain -j ACCEPT
iptables -A bad-dmz -p tcp --dport www -j ACCEPT
iptables -A bad-dmz -p tcp --dport https -j ACCEPT
iptables -A bad-dmz -p tcp --dport ssh -j ACCEPT
iptables -A bad-dmz -p icmp -j icmp-acc
ptables -A bad-dmz -j log-and-drop

# binnenste -> buitenste keten
# mail, DNS, http(s) en telnet worden geaccepteerd
iptables -A dmz-bad -p tcp --dport smtp -j ACCEPT
iptables -A dmz-bad -p tcp --sport smtp -j ACCEPT
iptables -A dmz-bad -p udp --dport domain -j ACCEPT
iptables -A dmz-bad -p tcp --dport domain -j ACCEPT
iptables -A dmz-bad -p tcp --dport www -j ACCEPT
iptables -A dmz-bad -p tcp --dport https -j ACCEPT
iptables -A dmz-bad -p tcp --dport telnet -j ACCEPT
iptables -A dmz-bad -p icmp -j icmp-acc
iptables -A dmz-bad -j log-and-drop

# Ketens voor de machine zelf
iptables -N bad-if
iptables -N dmz-if
iptables -A INPUT -i $BAD_IFACE -j bad-if
iptables -A INPUT -i $DMZ_IFACE -j dmz-if

# Externe interface
# SSH wordt enkel op deze machine geaccepteerd
iptables -A bad-if -p icmp -j icmp-acc
iptables -A bad-if -p tcp --dport ssh -j ACCEPT
iptables -A bad-if -p tcp --sport ssh -j ACCEPT
ipchains -A bad-if -j log-and-drop

# Interne interface
iptables -A dmz-if -p icmp -j icmp-acc
iptables -A dmz-if -j ACCEPT
```

Enkel woorden over de kwaliteit van de service (QOS, Quality Of Service). Linux kan het ToS ("Type of Service") veld aanpassen en zijn waarde veranderen om het pakket een andere prioriteit te geven. Het volgende commando verandert bijvoorbeeld de uitgaande SSH pakketten om het antwoord van de verbinding te verbeteren.

```
iptables -A OUTPUT -t mangle -p tcp --dport ssh -j TOS --set-tos Minimize-Delay
```

Op dezelfde manier kan je voor een FTP verbinding gebruik maken van de ’--set-tos Maximize-Throughput’ optie, om de overzendgraad ten nadele van de interactiviteit van de sessie te verbeteren.

Dat is het. Nu ken je de basis om een efficiënt pakketfiltersysteem op te zetten. Houd echter in gedachten dat een firewall geen wondermiddel is als het op veiligheid aankomt. Het is enkel wèèr een andere voorzorgsmaatregel. Het opzetten van een firewall houdt je niet tegen om ook gebruik te maken van goede wachtwoorden, de laatste veiligheidspatches, indringingsdetectie-systemen, etc.

Referenties

- Proxy-ARP Mini-HOWTO: http://www.linuxdoc.org/HOWTO/mini/Proxy-ARP-Subnet/index.html
- Netfilter: http://netfilter.samba.org/

Site onderhouden door het LinuxFocus editors team © Vincent Renardias "some rights reserved" see linuxfocus.org/license/ http://www.LinuxFocus.org	Vertaling info: fr --> -- : Vincent Renardias <vincent(at)renardias.com> fr --> en: Georges Tarbouriech <gt(at)linuxfocus.org> en --> nl: Samuel Derous <cyberprophet/at/linux.be>

2005-01-14, generated by lfparser_pdf version 2.51