15. Firewalls unter UNIX

Gliederung

- Grundlagen
- Linux: iptables
- OpenBSD: PF Toolkit
- **BSD, Solaris: IPF Toolkit**
- Löcher in Firewalls: Virtuelle Private Netze

Humboldt Universität Informatik
Systems Architecture Group
http://sar.informatik.hu-berlin.de

15.3 Firewall - Solaris

Historisches

IP Filter wurde ursprünglich von Darren Reed für BSD entwickelt. Es ist zurzeit für FreeBSD, NetBSD, OpenBSD, HU/UX, Irix, QNX, Solaris und SunOS verfügbar. IP Filter steht nicht unter GPL oder einer ähnlichen Lizenz – private Lizenz von D.Reed, die aber eine kostenlose Nutzung zuläßt.

Mit der Einführung von Solaris 10 stellte SUN den Vertrieb des eigenen Firewallproduktes SunScreen ein. IP Filter wurde Bestandteil des Betriebssystems. IP Filter kann bezüglich IP-Adressen, Ports, Protokolls, Verkehrsrichtung und Interfaces filtern. Zusätzlich ist die Möglichkeit der Bildung von Adresspools vorhanden.

15.3 Firewall - Solaris

Features

- Stateful Packet Filtering
- IPv4 und IPv6 Unterstützung
- Filtern nach: Typ, Flags, Fragmentierungs, IP-Optionen, ToS
- Logging über **syslogd**, eigenen Logdateien und **stdout**
- Statistiken
- Antwortmöglichkeiten auf eingehende Pakete
- NAT
- Unterstützung von aktivem FTP auch hinter dem Firewall
- RDR
- Unterstützt zwei Sätze von Filterregeln (aktiv und inaktiv)
- Separate Adresspool-Verwaltung

15.3 Firewall - Solaris

Aktivieren von IP Filter unter Solaris(2)

1. Eventuell **/etc/ipf/ippool.conf** erzeugen (Adresspools)
2. **/etc/ipf/ipf.conf** erzeugen (Filterregeln)
3. Eventuell **/etc/ipnat.conf** erzeugen (NAT-Regeln)
4. Forward aktivieren: **routeadm -e ipv4-forwarding**
5. IP Filter einschalten

   svcadm enable svc:/network/ipfilter:default
6. IP Filter aktivieren (scharf schalten)
   - Reboot oder
   - ifconfig ce0 unplumb

     ifconfig ce0 plumb 192.168.1.20 netmask 255.255.255.0 up

     ifconfig ce0 inet6 unplumb

     ifconfig ce0 inet6 plumb fec3:f840::1/96 up

15.3 Firewall - Solaris

Kommandos(1)

ipf - Programm zum Verwalten der aktiven und inaktiven Filterregeln. Pro Filterregelsatz werden die in- und die out-Regeln in einer separaten Liste verwaltet.

ipf [**-AdDEInoPrsvVyzZ**] [**-I block |pass |nomatch**]
[**-T** *optionlist*] [**-F i | o | a | s | S**] { **-f** *filename* }

-D - IP-Filter abschalten
-E - IP-Filter einschalten
-A - aktive Filterregeln verwalten
-I - inaktive Filterregeln verwalten
-d - Debug-Mode einschalten

Kommandos(2)

ipf [**-AdDEInoPrsvVyzZ**] [**-l block |pass |nomatch**]
[**-T** *optionlist*] [**-F i | o | a | s | S**] { **-f** *filename* }

-F [**i** | **o** | **a** | **s** | **S**] - Löschen von Informationen aus dem Filter.
Folgende Informationsgruppen können gelöscht werden:

i - Input Filter-Regeln

o - Output Filter-Regeln

a - alle Filter-Regeln

s - alle Informationen über unvollständige Verbindungen in der State-Tabelle

S - alle Informationen über Verbindungen in der State-Tabelle

Kommandos(3)

ipf [**-AdDEInoPrsvVyzZ**] [**-l block |pass |nomatch**]
[**-T** *optionlist*] [**-F i | o | a | s | S**] { **-f** *filename* }

-f *filename* - Name des Files, in dem Regeln für den IP Filter stehen.

-l block | pass | nomatch - Ändert die Default-Einstellung für das Logging von Paketen. Sinnvolle Einstellung ist **nomatch**. Hierdurch werden alle Pakete geloggt, für die keine gültige Regel in den Firewallregeln gefunden wurde.

-n - es wird keine Aktion ausgeführt (Test)

-P - Hinzufügen von Regeln als temporäre Regeln

-r - Löschen von Filterregeln aus dem File

ipf -r -f delete-rules

Kommandos(4)

ipf [**-AdDEInoPrsvVyzZ**] [**-I block |pass |nomatch**]
[**-T** *optionlist*] [**-F i | o | a | s | S**] { **-f** *filename* }

-s - Swap aktive Filterregeln gegen inaktive Filterregeln

-o - Nur die Output-Liste bearbeiten

-v - verbose

-V - Anzeigen der Versionsnummer und Zustandsinformationen von ipf

-y - manuelle Synchronisation der in-Kernel-Interfaces mit der aktuellen Interface-Status-Liste.

-z - Statistiken für Regeln im Eingabefile zurücksetzen

-Z - Statistiken für aktive Regeln im Kern zurücksetzen

15.3 Firewall - Solaris

Kommandos(5)

ipf [**-AdDEInoPrsvVyzZ**] [**-I block |pass |nomatch**]
[**-T** *optionlist*] [**-F i | o | a | s | S**] { **-f** *filename* }

Beispiele:

- ipf -E # Einschalten des IP-Filters
- ipf -f /etc/ipf/ipf.conf # Aktivieren der Paketfilterung
- ipf -Fa # Löschen des aktiven Regelsatzes im Kern
- ipf -Fi # Löschen der aktiven Regeln für Input im Kern
- ipf -Fo # Löschen der aktiven Regeln für Output im Kern
- ipf -I -f inakt.ipf.conf # Laden von Regeln in den inaktiven Regelsatz
- ipf -s # Umschalten zwischen aktiven und inaktivem Regelsatz
- ipf -I -Fa # Löschen des inaktiven Regelsatzes im Kern

15.3 Firewall - Solaris

Kommandos(6)

ipf [**-AdDEInoPrsvVyzZ**] [**-I block |pass |nomatch**]
[**-T** *optionlist*] [**-F i | o | a | s | S**] { **-f** *filename* }

Beispiele:

- ipf -Fa -D # Abschalten des IP-Filters, alles darf durch
- ipf -FS # Eintragungen in der State-Tabelle löschen

Kommandos(7)

ippool - Programm zur Verwaltung von IP-Adressen im IP-Adressen-Pool. IP-Adressen-Pools sind Mengen von IP-Adressen, die auf verschiedene Weise gebildet werden können. Diesen Mengen werden IDs zugeordnet, über die dann in den Regeln auf die IP-Adressen zugegriffen werden kann. Das Programm kann wie folgt aufgerufen werden:

ippool -a [**-dnv**] **-m** *num* [**-r** *role*] **-i** *ipaddr*[*Inetmask*]

ippool -A [**-dnv**] **-m** *num* [**-r** *role*] [**-S** *seed*] [**-t** type]

ippool -f *file* [**-dnv**]

ippool -F [**-dv**] [**-r** *role*] [**-t** *type*]

ippool -l [**-dv**] [**-m** *num*] [**-t** *type*]

ippool -r [**-dnv**] [**-m** *num*] [**-r** *role*] **-i** *ipaddr*[*Inetmask*]

ippool -R [**-dnv**] [**-m** *num*] [**-r** *role*] [**-t** *type*]

ippool -s [**-dnv**] [**-M** *core*] [**-N** *namelist*]

Kommandos(8)

Allgemeine Optionen für **ippool**

-d - Debugging Informationen mit ausgeben.

-n - nur Test des Kommandos, keine Aktion über dem Kern

-v - verbose

ippool -a [**-dnv**] **-m** *num* [**-r** *role*] **-i** *ipaddr*[*/netmask*]

-a - Hinzufügen neuer IP-Adressen zu einem bestehenden ID

-m *num* - num ist der Name (Nummer) des ID

-r *role* - Funktion, für die der IP-Pool benutzt werden kann:
ipf, auth, count (z.Z. nicht unterstützt)

-i *ipaddr*[*/netmask*] - Menge der IP-Adressen(141.20.20.20,
141.20.20.0/255.255.255.0, 141.20.20.0/24)

15.3 Firewall - Solaris

Kommandos(9)

ippool -A [**-dnv**] **-m** *num* [**-r** *role*] [**-S** *seed*] [**-t** type]

-A - Erzeugen eines neuen leeren IDs im Kern

-S *seed* - Hashwert für Hash-Pools

-t *type* - Pooltype: **pool**, **hash**, **group-map**

-m, **-r** - siehe Option bei **-a**

ippool -f *file* [**-dnv**]

-f *file* - Laden des IP-Pools aus einem File

ippool -F [**-dv**] [**-r** *role*] [**-t** *type*]

-F - Löschen der Pools.

15.3 Firewall - Solaris

Kommandos(11)

ippool -l [**-dv**] [**-m** *num*] [**-t** *type*]

-l - Anzeigen des Inhaltes des Pools

ippool -r [**-dnv**] **-m** *num* [**-r** *role*] **-i** *ipaddr*[*Inetmask*]

-r - Löschen von IP-Adressen aus einem ID *num*

ippool -R [**-dnv**] **-m** *num* [**-r** *role*] [**-t** *type*]

-R - Löschen einer ID *num*

ippool -s [**-dtv**] [**-M** *core*] [**-N** *namelist*]

-s - Anzeigen von Statistikinformationen

Kommandos(12)

Beispiele:

```
# Anzeigen des Inhaltes des Pools
ippool -l
# Löschen des Inhaltes des Pools
ippool -F
# Laden des Pools vom File
ippool -f /etc/ipf/ippool.conf
# Laden einer Adresse in den Pool 10
Ippool -a -m 10  -i 192.168.1.1/24
```

Kommandos(13)

ipfstat - Ausgabe von IP Filterregeln und Statistiken bei laufendem IP-Filter.
Das Programm kann wie folgt aufgerufen werden:

ipfstat [**-aACdfghlilnostv**] [**-D** *addrport*] [**-P** *protocol*] [**-S** *addrport*] [**-T** *refreshtime*]

-a - Anzeigen der Accounting Filter Liste (optional)

-A - Anzeigen der Authentikation-Statistik

-C - Anzeigen der Statistik für offene Verbindungen, zusammen mit **-t**

-t - Wiederholt anzeigen

-d - Debugging on

-D *addrport* - Protokoll nur für angegebene Zieladresse (*ipaddr*[**,***port*) anzeigen

-P *protocol* - Ausgaben nur für das angegebene Protokoll.

15.3 Firewall - Solaris

Kommandos(14)

ipfstat [**-aACdfghlilnostv**] [**-D** addrport] [**-P** protocol] [**-S** addrport]
[**-T** refreshtime]

-S *addrport* - Protokoll nur für angegebene Quelladresse (*ipaddr*[,*port*]) anzeigen

-f - IP-Fragment Status anzeigen

-g - Gruppen Anzeigen (wenn vorhanden)

-h - Trefferanzeige pro Regel (**-hi**)

-I (grosses i) - Statistik für inaktive Regeln

-i - input-Regeln

-l (kleines l) - zusammen mit **-s** anzeigen der aktiven State-Eintragungen

-n - Regelnummern mit ausgeben.

15.3 Firewall - Solaris

Kommandos(15)

ipfstat [**-aACdfghlilnostv**] [**-D** addrport] [**-P** protocol] [**-S** addrport]
[**-T** refreshtime]

-o - output-Regeln anzeigen

-s - Statistik für Statusinformationen anzeigen

-v - verbose

-T refreshtime - Zusammen mit **-t**, Statistiken wiederholt anzeigen. refreshtime ist das Zeitintervall in Sekunden

Kommandos(16)

ipfstat [**-aACdfghlilnostv**] [**-D** addrport] [**-P** protocol] [**-S** addrport]
[**-T** refreshtime]

Beispiele:

ipfstat -ai
ipfstat -aio
ipfstat -A
ipfstat -Ct
ipfstat -C
ipfstat -C -S 141.20.20.67,22 -t
ipfstat -C -D 141.20.20.22 -t
ipfstat -f

15.3 Firewall - Solaris

Kommandos(17)

ipfstat [**-aACdfghlilnostv**] [**-D** addrport] [**-P** protocol] [**-S** addrport]
[**-T** refreshtime]

Beispiele:

ipfstat -g
ipfstat -hi
ipfstat -s
ipfstat -ls
ipfstat -in
ipfstat -in
ipfstat -Ct -T 3

Humboldt Universität Informatik
Systems Architecture Group
http://sar.informatik.hu-berlin.de

Kommandos(18)

ipnat - Programm zum Laden der NAT-Regeln in den IP-Filter. Das Programm kann wie folgt aufgerufen werden:

ipnat [**-lhnrsvCF**] **-f** *filename*

-C - löschen aller NAT Regeln

-F - löschen aller z.Z. aktiven NAT-Eintragungen (vorhandene Verbindungen)

-f filename - Laden der NAT-Regeln vom File filename (ohne Option **-r**)

-h - Anzeigen der Treffer der NAT-Regeln

-l (kleines l) - Anzeigen der aktuellen NAT-Regeln

-n - nichts machen, nur Syntax prüfen

-s - Statistiken anzeigen

-r - löschen von NAT-Regeln, die im File spezifiziert sind.

15.3 Firewall - Solaris

Kommandos(19)

ipnat [**-lhnrsvCF**] **-f** *filename*

Beispiele:

```
# Liste der NAT-Regeln anzeigen
ipnat -l
# Statistik anzeigen lassen
ipnat -s
# Treffer anzeigen lassen
ipnat -h
```

Kommandos(20)

ipfs - Speichern und laden von NAT-Regeln und Statusinformationen. Damit ist es möglich Statusinformationen über ein „Reboot“ zu retten. Verbindungen werden dadurch nicht unterbrochen. Das Programm kann wie folgt aufgerufen werden:

ipfs [**-nv**] **-l**

ipfs [**-nv**] **-u**

ipfs [**-nv**] [**-d** *dirname*] **-R**

ipfs [**-nv**] [**-d** dirname] **-W**

ipfs [**-nNSv**] [**-f** *filename*] **-r**

ipfs [**-nNSv**] [**-f** *filename*] **-w**

ipfs [**-nNSv**] **-f** *filename* **-i** *interface1*,*interface2*

15.3 Firewall - Solaris

Kommandos(21)

ipfs [**-nv**] **-l**

-n - nichts tun, nur testen

-v - verbose

-l - sperren der State-Tabelle im Kern

ipfs [**-nv**] **-u**

-u - entsperren der State-Tabelle im Kern

ipfs [**-nv**] [**-d** *dirname*] **-R**

-R - Rückspeichern der Informationen aus der angegebenen Direktory (-d dirname) in den Kern. Standard: /var/db/ipf

15.3 Firewall - Solaris

Kommandos(22)

ipfs [**-nv**] [**-d** dirname] **-W**

-W - Schreiben der ipfstate-Tabelle in das angegebene Direktory

ipfs [**-nNSv**] [**-f** *filename*] **-r**

-r - Lesen der Informationen vom File und in den Kern speichern

ipfs [**-nNSv**] [**-f** *filename*] **-w**

-w - Schreiben der Informationen aus dem Kern in das File

ipfs [**-nNSv**] **-f** *filename* -i *interface1*,*interface2*

Kommandos(23)

ipmon - Anzeigen der Protokoll-Informationen, die über /dev/ipl ausgegeben werden. ipmon kann wie folgt aufgerufen werden:

ipmon [**-abDFhnpstvxX**] [**-N** *device*] [[**-o**] [**N** |**S** | **I**]] [**-O** [**N** | **S** | **I**]] [**-P** *pidfile*] [**-S** *device*] [**-f** *device*] [*filename*]

-a - Alle Logfiles anzeigen

-b - erzeugen von HEX-Kode für den Inhalt von Paketen

-D - starten von ipmon als Daemon

-f *device* - alternatives Input-File für LOG-Informationen

-F - löschen von Pufferinformationen

-h - Help-Informationen ausgeben

-n - Namensauflösung für Hosts und Ports

Kommandos(23)

ipmon [**-abDFhnpstvxX**] [**-N** *device*] [[**-o**] [**N** | **S** | **I**]] [**-O** [**N** | S | **I**]]
[**-P** *pidfile*] [**-S device**] [**-f** *device*] [*filename*]

-n - Namensauflösung für Hosts und Ports

-N *device* - NAT-Informationen für das Gerät auslesen

-o [**N** | **S**| **I**] - Art des Logfiles, das ausgewertet werden soll:
N -NAT, **S** - Statusfile, **I** - IP-Filter

-O [**N** | **S** | **I**] - Art der Loginformationen, die nicht ausgegeben werden sollen

-p - Portnummern immer als Nummern anzeigen.

-s - Ausgabe der Informationen über den **syslogd**

15.3 Firewall - Solaris

Kommandos(24)

ipmon [**-abDFhnpstvxX**] [**-N** *device*] [[**-o**] [**N** | **S** | **I**]] [**-O** [**N** | **S** | **I**]]
[**-P** *pidfile*] [**-S** *device*] [**-f** *device*] [*filename*]

-t - tail-Form der Auswertung
-v - verbose (TCP-Window,ack, sequence-fields)
-x - Anzeigen in HEX-Format
-X - nur Header in HEX-Format

filename – Ausgabe von **ipmon**, wenn nicht angegeben erfolgt die Ausgabe auf STDOUT

15.3 Firewall - Solaris

Kommandos(25)

ipmon [**-abDFhnpstvxX**] [**-N** *device*] [[**-o**] [**N** | **S** | **I**]] [**-O** [**N** | **S** | **I**]]
[**-P** *pidfile*] [**-S** *device*] [**-f** *device*] [*filename*]

Beispiele:

ipmon -a
ipmon -ao N
ipmon -ao I
ipmon -an
cat /dev/ipl > /tmp/logfile
ipmon -f /tmp/logfile

Konfigurationsfiles(1)

/etc/ipf/ippool.conf - Konfigurationsfile für IP-Address-Pool beim starten des Betriebssystems. Ein Eintrag hat folgende einfache Syntax:

table role = *role-name* **type =** *storage-format* **number =** *id-number*

{ *ipaddr*/*maskbits* {, *ipaddr*/*maskbits*} **};**

Dabei bedeutet:

role-name - Rolle im Filter, z.Z. nur **ipf**

storage-format - Speicherformat für die IP-Adressen, z.Z. gibt es die Formate **hash** und **tree** .

id-number - Regelnummer, diese Zahl wird zur Referenzierung der Tabelle in den Regelwerken benötigt. z.B

pass in from pool/33 **to any**

ipaddr/maskbists - beschreibt die IP-Adressen, die zu der Tabelle gehören, z.B. 141.20.20.0/24, 141.20.23.13/32

Konfigurationsfiles(2)

/etc/ipf/ippool.conf

Beispiel:

```
table role = ipf type = tree number = 10
   { 141.20.20.18/32, 141.20.20.20/32, 141.20.20.22/32, 141.20.20.50/32, \
                141.20.20.51/32, 141.20.20.78/32 , 141.20.28.18/32 };
table role = ipf type = tree number = 20
       { 141.20.20.0/24 };
table role = ipf type = tree number = 100
       { 141.20.20.0/24, 141.20.21.0/24, 141.20.22.0/24, 141.20.23.0/24, \
         141.20.24.0/24, 141.20.25.0/24, 141.20.26.0/24, 141.20.27.0/24, \
         141.20.28.0/24, 141.20.30.0/24, 141.20.31.0/24, 141.20.32.0/24, \
         141.20.33.0/24, 141.20.34.0/24, 141.20.35.0/24, 141.20.36.0/24, \
         141.20.37.0/24, 141.20.38.0/24, 141.20.39.0/24  };
```

15.3 Firewall - Solaris

Konfigurationsfiles(3)

/etc/ipf/ipf.conf - Konfigurationsfile für den IP-Filter für die Filterregeln. Dieses File enthält die Anfangskonfiguration des IP-Filters bezüglich der Filterregeln, die beim Booten des Systems geladen werden. Die Syntax des Regelwerkes entspricht denen von **PFCTL** bei **OpenBSD**. Allerdings werden keine Makros unterstützt. Dafür wurde der IP-Pool-Mechanismus integriert. Die Abarbeitung des Regelwerkes durch den IP-Filter erfolgt ebenfalls ähnlich wie bei **OpenBSD**, alle Regeln werden der Reihe nach ausgewertet. Die Aktion der letzten passenden Regel wird ausgeführt, es sei denn, die Option **quick** wurde spezifiziert. Die Eintragungen im File haben folgendes Format:

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

Konfigurationsfiles(3)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

action steht dabei für:

- **block** - Paket blockieren
- **pass** - Paket passieren lassen
- **log** - Protokoll des Paketes für ipmon
- **count** - zählen des Paketes für die Statistik für ipfstat
- **skip** *number* - *number* Regeln übergehen (Sprung im Regelwerk)
- **auth** - Authentifizierung notwendig

und stellt die Aktion dar, die ausgeführt werden soll, wenn die nachfolgenden Bedingungen passen.

Konfigurationsfiles(4)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

Die Schlüsselwörter **in** bzw. **out** charakterisieren eingehende bzw. ausgehende Pakete.

option steht für zusätzliche Aktionen bzw. Bedingungen.

log - Protokollieren des Paketes, wenn dies die letzte passende Regel ist. Kann durch **ipmon** angezeigt werden.

quick - Es wird keine weitere passende Regel gesucht, wenn diese Regel passt. Die zugehörige Aktion wird ausgeführt.

on *interfacename* - Spezifiziert diese Regel nur für das angegebene Interface *interfacename*.

15.3 Firewall - Solaris

Konfigurationsfiles(5)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

Weitere Möglichkeiten für *option*

dup-to *interfacename* - Kopieren des Paketes zu Ausgabe zu dem angegeben Interface.

to *interfacename* - Umlenken des Paketes zur Ausgabe an das angegeben Interface.

Konfigurationsfiles(6)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

Nach den Optionen folgen eine Reihe von Bedingungen *keyword*, die entweder erfüllt sein müssen, damit die Regel passt und die zugehörige Aktion ausgeführt werden kann, wenn dies die letzte passende Regel ist (Ausnahme: **quick**) oder zusätzliche Aktionen im Filter auslösen - z.B. keep. Folgende Bedingungen sind möglich:

tos	**ttl**	**proto**
from	**to**	**all**
with	**flags**	**keep**
head	**group**	**icmp-type**

15.3 Firewall - Solaris

Konfigurationsfiles(7)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

keyword:

tos *number* - Das Paket hat den Type-of-Service Wert *number* (hex. oder dez. Zahl).

ttl value - Das Paket hat den TTL-Wert value (Zahl).

proto *protocol* - Das Paket gehört zum Protokoll *protocol*. Folgende Angaben sind für *protocol* zulässig:

tcp/udp, **tcp**, **udp**, **icmp**, dez. Nummer des Protokolls

15.3 Firewall - Solaris

Konfigurationsfiles(8)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

from [**!**] *ipaddress* **port** *ports* - Paket hat die Quelladresse.

ipaddress kann folgende Werte annehmen:

- *Hostname*[*IMaskenbits*]
- *IP-Addresse*[/Maskenbits]
- **any oder <thishost>**
- **pool***/pool-id* - ID eines Pools von IP-Adressen.

ports kann folgende Werte annehmen:

- [=|<|>|<=|>=] *portnummer*
- *portnummer* [<>|><] portnummer

Konfigurationsfiles(9)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

to [**!**] *ipaddress* **port** *ports* - Paket hat die Zieladresse.

ipaddress kann folgende Werte annehmen:

- *Hostname*[*IMaskenbits*]
- *IP-Addresse[*/Maskenbits]
- **any oder <thishost>**
- **pool***/pool-id* - ID eines Pools von IP-Adressen.

ports kann folgende Werte annehmen:

- [=|<|>|<=|>=] *portnummer*
- *portnummer* [<>|><] portnummer

Konfigurationsfiles(10)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

keyword:

all - Das Paket kann von beliebigen IP-Adressen und Ports kommen und zu beliebigen IP-Adressen und Ports weitergeleitet werden

with [**not**|**no**] [**ipopts** | **short** | **fraq** | **opt** *optname*] - Paket erfüllt die angegeben Attribute. Dient zum Erkennen von fehlerhaften IP-Paketen.

15.3 Firewall - Solaris

Konfigurationsfiles(11)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

keyword:

flags *flags*[/*flagmask*] - Im Paket sind die angegeben Flags flags gesetzt. Wird nur bei TCP benutzt. flagmask dient zum definieren von Testmasken. Folgende Flags sind erlaubt:

F - FIN, S - SYN, R - RST,

A - ACK, U -URG

Beispiele:

flags S - Flag S ist gesetzt

flags S/SA - Flag S ist gesetzt und Flag A nicht!

Systems Architecture Group
http://sar.informatik.hu-berlin.de

Konfigurationsfiles(12)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

keyword:

icmp-type *icmptype* [**code** *number*] - Nur gültig für ICMP-Pakete. Das Paket ist vom angegeben Type *icmptype*. Für icmptype sind folgende Angaben möglich:

unreach, **echo**, **echorep**, **squench**, **redir**, **timex**,**inforeq**, **paramprob**, **timest**, **timestrep**, **inforep**, **maskreq**, **maskrep**

number gibt den Untertype für den *icmptype*.

Konfigurationsfiles(13)

/etc/ipf/ipf.conf

action [**in** | **out**] {*option*} *keyword*{,*keyword*}

keyword:

keep state - Wenn das Paket den Filter erfolgreich passiert, wird der Status der Verbindung gespeichert und weitere Pakete, die zu dieser Verbindung gehören, können den Filter ebenfalls passieren.

keep frags - Wenn das Paket ein Fragment ist, werden Informationen über dieses Fragment gespeichert und bei späteren Fragmenten berücksichtigt.

Konfigurationsfiles(14)

/etc/ipf/ipnat.conf - Konfigurationsfile für den IP-Filter für die NAT-Regeln. Dieses File enthält die Anfangskonfiguration des IP-Filter bezüglich der NAT-Regeln, die beim Booten geladen werden. NAT-Regeln haben folgenden prinzipiellen Aufbau:

map *interface-name old-ip* **->** *new-ip-mask*

für NAT, neue Adresse wird zufällig, round-robin Verfahren bestimmt

bitmap *interface-name old-ip* **->** *new-ip-mask*

für BINAT, bidirektionales NAT, feste Adresszuordnung

rdr *interface-name old-ip-mask* **port** *old-port* **->** \
new-ip-mask **port** *new-port protocol*

für RDR, Umlenkung einzelner Ports.

Humboldt Universität Informatik
Systems Architecture Group
http://sar.informatik.hu-berlin.de

Konfigurationsfiles(15)

/etc/ipf/ipnat.conf

Beispiele:

1. rdr le0 141.20.20.40/32 -> 201.1.2.3/32
2. rdr le0 203.1.2.3/32 port 80 -> 203.1.2.3/32,203.1.2.4/32 port 80 \
   tcp round-robin
   rdr le0 203.1.2.3/32 port 80 -> 203.1.2.5/32 port 80
3. map ce0 from 10.1.0.0/16 to any -> 201.1.2.3/32
4. map eri1 from 192.168.199.0/24 to any -> 141.20.20.120/32
   map ppp0 from 10.0.0.0/8 -> 203.1.2.0/24 portmap tcp/udp \
   3000:65000
   map ppp0 from 10.0.0.0/8 -> 203.1.2.0/24 portmap tcp/udp auto

Beispiele(1)

1. Reihenfolge

```
# alles kann passieren
block in all
pass in all

# alles wird blockiert
pass in all
block in all

# alles wird blockiert
block in quick all
pass in all
```

Beispiele(2)

2. Filtern nach IP-Adressen - blockieren privater Adress-Räume

```
block in quick from 192.168.0.0/16 to any
block in quick from 172.16.0.0/12 to any
block in quick from 10.0.0.0/8 to any
pass in all
```

3. Interface erlauben

```
pass in quick on bge0 to any keep state
block in all
```

15.3 Firewall - Solaris

Beispiele(3)

4. Interface und IP-Adressen

```
block in quick on bge0 from 192.168.4.0/24 to any
pass in quick on bge1 from 192.168.3.0/24 to any keep state
block in all
```

5. Eigenes Routen von privaten Netzen verhindern - etwas für die Allgemeinheit tun

```
block out quick on bge0 from any to 192.168.0.0/16
block out quick on bge0 from any to 172.168.0.0/12
block out quick on bge0 from any to 10.0.0.0/8
block out quick on bge0 from any to 127.0.0.0/8
pass in all keep state
pass out all keep state
```

Beispiele(4)

6. Pakete von privaten Netzen verbieten und protokollieren - Schutz für uns

```
block in log quick on bge0 from any to 192.168.0.0/16
block in log quick on bge0 from any to 172.168.0.0/12
block in log quick on bge0 from any to 10.0.0.0/8
block in log quick on bge0 from any to 127.0.0.0/8
pass in all keep state
```

Beispiele(5)

7. Vollständiger einfacher Regelsatz (bge0 - extern, bge1 - intern) (1)

```
block in all
block out all
pass out quick on lo0 keep state
pass in quick on lo0 keep state
block out quick on bge0 from any to 192.168.0.0/16
block out quick on bge0 from any to 172.168.0.0/12
block out quick on bge0 from any to 10.0.0.0/8
block out quick on bge0 from any to 127.0.00/8
```

Beispiele(6)

7. Vollständiger einfacher Regelsatz (bge0 - extern, bge1 - intern) (2)

```
block in log quick on bge0 from any to 192.168.0.0/16
block in log quick on bge0 from any to 172.168.0.0/12
block in log quick on bge0 from any to 10.0.0.0/8
block in log quick on bge0 from any to 127.0.00/8
pass out quick on bge1 keep state
pass in on quick bge1 keep state
#
pass out on bge0 proto tcp/udp from any to any keep state
```

Beispiele(7)

8. ICMP-Filtern

```
# kein ICMP (kein ping)
block in log quick on bge0 icmp from any to any
# ordentliches Ping erlauben, böses Ping verbieten (Spoofing verbieten)
block in quick  on bge0 from 192.168.0.0/16 to any
   ...
block in quick on bge0 from 127.0.0.0/8 to any
pass in quick on bge0 proto icmp from any to <thishost> icmp-type 0
pass in quick on bge0 proto icmp from any to <thishost> icmp-type 11
block in log quick on bge0 prot icmp from any to any
pass in all keep state
```

Beispiele(8)

9. Portspezifisches Blockieren von unliebsamen Paketen (neugieriges Window)

```
block in on ce0 proto udp from 141.20.20.16 port = 520 to 141.20.20.255/32
block in on ce0 proto udp from any port 137 to <thishost>
block in on ce0 proto udp from any port 137 to 141.20.20.255/32
block in on ce0 proto udp from any port 138 to <thishost>
block in on ce0 proto udp from any port 137 to 141.20.20.255/32
```

Beispiele(9)

10. Personal Firewall für einen Server mit NFS

Der Server stellt folgende Dienste zur Verfügung:

- Mail mit SMTP,POP,IMAP
- Radius-Server
- DNS-Server
- NFS-Server für spezielle Rechner
- NIS-Server für spezielle Rechner

Problem: NFS-Ports sind variable und können erst nach dem Start des Dienstes in das Regelwerk des Firewalls integriert werden!!!

Beispiele(10)

/etc/ipf/ippool.conf

```
table role = ipf type = tree number = 10
      { 141.20.20.18/32, 141.20.20.20/32, 141.20.20.22/32, 141.20.20.50/32,  \
              141.20.20.67/32, 141.20.20.78/32, 141.20.28.18/32 };
table role = ipf type = tree number = 20
      { 141.20.20.0/24, 141.20.21.0/24, 141.20.253.0/24, 141.20.1.0/24 };
table role = ipf type = tree number = 100
      { 141.20.20.0/24, 141.20.21.0/24, 141.20.22.0/24, 141.20.23.0/24, \
        141.20.24.0/24, 141.20.25.0/24, 141.20.26.0/24, 141.20.27.0/24, \
        141.20.28.0/24, 141.20.30.0/24, 141.20.31.0/24, 141.20.32.0/24, \
        141.20.33.0/24, 141.20.34.0/24, 141.20.35.0/24, 141.20.36.0/24, \
         141.20.37.0/24, 141.20.38.0/24, 141.20.39.0/24  };
```

Beispiele(11)

/etc/ipf/ipf.conf

```
# My IP: 141.20.20.67  <thishost>
# Block any packets which are too short to be real
# Interface: ce0
block in log quick all with short
# drop and log any IP packets with options set in them.
block in log all with ipopts
# Allow all traffic on loopback.
pass  in quick on lo0 all keep state
pass out quick on lo0 all keep state
# Public Network.   Block everything not explicity allowed.
block  in log on ce0 all
block out log on ce0 all
```

Beispiele(12)

```
# Allow pings out.
pass out quick on ce0 proto icmp all keep state
#
# Allow outbound state related packets.
pass out quick on ce0 proto tcp/udp from any to any keep state
#
# for testing, allow pings from bellus,bellus1 and bellus3
pass in quick on ce0 proto icmp from pool/10 to <thishost> keep state
#
# Actually, allow ssh only from  pool 10 (bellus, bellus3, ....)
pass in quick on ce0 proto tcp from pool/10 to <thishost> port = 22 keep state
```

Beispiele(13)

```
#       mail (alle)
#   smtp, imap, imaps, pop2, pop3, pop3s
pass in quick on ce0 proto tcp from any to <thishost> port = 25 keep state
pass in quick on ce0 proto tcp from any to <thishost> port = 143 keep state
pass in quick on ce0 proto tcp from any to <thishost> port = 993 keep state
pass in quick on ce0 proto tcp from any to <thishost> port = 109 keep state
pass in quick on ce0 proto tcp from any to <thishost> port = 110 keep state
pass in quick on ce0 proto tcp from any to <thishost> port = 995 keep state
```

Beispiele(14)

```
#      NIS-Server: mail, mailslv1, knecht, bellus
pass in quick on ce0 proto udp from 141.20.20.18 to <thishost> keep state
pass in quick on ce0 proto udp from 141.20.20.50 to <thishost> keep state
pass in quick on ce0 proto udp from 141.20.20.51 to <thishost> keep state
pass in quick on ce0 proto udp from 141.20.20.52 to <thishost> keep state
#
#      DNS-Server (alle)
pass in quick on ce0 proto tcp/udp from any to <thishost> port = 53 keep state
#
# ident-server
pass in quick on bge0 proto tcp from  pool/100 to <thishost> port = 113 \
          keep state
```

Beispiele(15)

```
#       Radius-Server (bekannte radius-clienten)
pass in quick on ce0 proto tcp/udp from pool/20 to <thishost> port = 1645 \
            keep state
pass in quick on ce0 proto tcp/udp from pool/20 to <thishost> port = 1646 \
            keep state
pass in quick on ce0 proto tcp/udp from pool/20 to <thishost> port = 1647 \
            keep state
#
#     ntp (Rechner des Instituts)
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 123 \
            keep state
pass in quick on ce0 proto tcp/udp from pool/100 to 141.20.20.255/32  \
            port = 123 keep state
```

Beispiele(16)

```
#   nfs-server  (Rechner des Instituts)
pass in quick on ce0 proto tcp/udp from pool/20 to 141.20.20.255/32 \
        port = 111 keep state
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> \
        port = 111 keep state
pass in quick on ce0 proto tcp/udp from pool/100 port = 2049 to <thishost> \
        port 0 >< 1024 keep state
#
# Sperren ohne Protokoll
block in on ce0 proto udp from 141.20.20.16 port = 520  to 141.20.20.255/32
block in on ce0 proto udp from any port = 137  to 141.20.20.255/32
block in on ce0 proto udp from any port = 137  to <thishost>
block in on ce0 proto udp from any port = 138  to 141.20.20.255/32
block in on ce0 proto udp from any port = 138  to <thishost>
```

Beispiele(17)

Erzeugen der Regeln, die von RPC abhängig sind.

Skript: genipf

```
#!/bin/sh
rpcinfo -p | grep status     | awk '{ print $4;}' > /tmp/netlog
rpcinfo -p | grep ypbind     | awk '{ print $4;}' >> /tmp/netlog
rpcinfo -p | grep ypserv     | awk '{ print $4;}' >> /tmp/netlog
rpcinfo -p | grep nlockmgr   | awk '{ print $4;}' >> /tmp/netlog
rpcinfo -p | grep rstatd     | awk '{ print $4;}' >> /tmp/netlog
rpcinfo -p | grep rusersd    | awk '{ print $4;}' >> /tmp/netlog
rpcinfo -p | grep rquotad    | awk '{ print $4;}' >> /tmp/netlog
rpcinfo -p | grep mountd     | awk '{ print $4;}' >> /tmp/netlog
rpcinfo -p | grep nfs        | awk '{ print $4;}' >> /tmp/netlog
rpcinfo -p | grep bootparam | awk '{ print $4;}' >> /tmp/netlog
```

Beispiele(18)

```
rm -fr /tmp/netlog-dir
mkdir /tmp/netlog-dir
cd /tmp/netlog-dir
for i in `cat /tmp/netlog`
  do
    touch $i
done
rm -f /etc/ipf/ipf.conf.add ]
for i in *
do
   echo "# `/bin/rpcinfo -p | /bin/grep $i | /bin/head -1" >> /etc/ipf/ipf.conf.add
   echo "pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> \
                                          port = $i keep state" >> /etc/ipf/ipf.conf.add
done
```

15.3 Firewall - Solaris

Beispiele(19)

```
cd /etc/ipf
/usr/sbin/ipf -f /etc/ipf/ipf.conf.add
#
# der nachfolgende Befehl schaltet den Firewall ab!!!!!
# /usr/sbin/ipf -Fa
#
```

/etc/ipf.conf.add enthält jetzt die für RPC notwendigen zusätzlichen Filter-Regeln.

15.3 Firewall - Solaris

Beispiele(20) /etc/ipf/ipf.conf.add

```
#    100024    1   tcp  32771  status
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 32771 keep state
#    100024    1   udp  32772  status
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 32772 keep state
#    100007    3   udp  32775  ypbind
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 32775 keep state
#    100002    2   tcp  32786  rusersd
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 32786 keep state
#    100001    2   udp  32788  rstatd
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 32788 keep state
#    100002    2   udp  32789  rusersd
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 32789 keep state
#    100011    1   udp  32790  rquotad
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 32790 keep state
#    100021    1   udp   4045  nlockmgr
pass in quick on ce0 proto tcp/udp from pool/100 to <thishost> port = 4045 keep state
```