

iptables

Gliederung:

- Firewall – Technologien (Filtertechnologien)
- Stateful Packet Inspection (SPI)
- iptables
 - was ist iptables
 - Aufbau und Funktion der filter Tabelle
 - Aufbau einer Regel
- Beispielkonfiguration anhand der Linux Firewalldistribution IPcop

Was versteht man unter dem Begriff Firewall?

Ganz allgemein gesagt, ist eine Firewall eine Zusammensetzung von verschiedenen Softwaremodulen, die den ein- und ausgehenden Datenverkehr eines Netzwerkes mittels festgelegter Regeln überwachen.

Firewall – Technologien (Filtertechnologien)

Zunächst gehe ich auf vier verschiedene Filtertechnologien ein, mit deren Hilfe man eine sehr wirksame Firewall erstellen kann.

Diese sind jedoch in 2 Hauptgruppen unterteilt, zum einen die Contentfilter und zum anderen die Paketfilter.

Wobei die Paketfilter, vorallem der Stateful Packet Inspection am wichtigsten ist um eine effektive Firewall zu erstellen (bzw. einzurichten)!

Hier nun ein kurzer Überblick zu den Filtertechnologien.

Contentfilter:

- Proxyfilter
 - initiiert stellvertretend für den anfragenden PC (Client) die Verbindung mit dem Zielsystem und leitet die Antwort des Zielsystems an den anfragenden Client weiter
 - meist auf ein bestimmtes Kommunikationsprotokoll (z.B. HTTP) festgelegt
 - speichert Antworten (z.B. Webseiten) der Zielsysteme zwischen, um diese bei mehrmaligen Anfragen verschiedener hinter dem Proxyfilter sitzender Clients aus dem Cache heraus zu beantworten. So muss die Anfrage nicht noch einmal gestellt werden
- Contentfilter
 - spezielle Form eines Proxyfilters
 - filtert anhand der Nutzerdaten (z.B. können Inhalte wie ActiveX oder JavaScript von Webseiten durch diesen Filter verändert oder herausgefiltert und dann erst dem Client übermittelt werden)

Paketfilter:

Paketfilter filtern ein- und ausgehenden Netzwerkverkehr anhand der Headerinformationen auf der OSI Schicht 3 Network Layer (IP-Adressen) und der OSI Schicht 4 Transport Layer (Ports)

- Stateless Packet Inspection (zustandsloser Paketfilter)
 - benötigt 2 Regeln für eine Kommunikation (quasi einen Hin- und Rückkanal)
 - Quell- und Zielsystem können somit eine Verbindung initiieren -
Das ist aber in den meisten Fällen unerwünscht, da normalerweise der Verbindungsaufbau am Client (Quellsystem) gestartet wird
 - Beispiel für einen zustandslosen Paketfilter ist ipchains, welches bis zur Linux Kernelversion 2.2 mit ausgeliefert wurde
- Stateful Packet Inspection (zustandsorientierter Paketfilter)
 - Es wird nur eine Regel für eine Kommunikation benötigt
Die zweite Regel (quasi die für den Rückkanal) wird beim Verbindungsaufbau temporär bzw. dynamisch erstellt
 - UDP Verbindungen werden genauso wie TCP Verbindungen gehandhabt, auch wenn UDP eigentlich als zustandsfrei betrachtet wird
 - Verbindungen haben zugewiesene Timeouts -
Grund dafür ist, das bei UDP nicht erkennbar ist, wann eine Verbindung beendet worden ist und bei TCP, weil es durchaus vorkommen kann, dass eine Verbindung nicht korrekt abgebaut wird bzw. abbricht
 - Beispiele hierfür sind iptables (seit Linux Kernel 2.4), ipfw unter FreeBSD bzw. MacOS

Was ist „iptables“?

Iptables ist ein Frontend zum editieren der Filtertabellen des Kernels und ist Teil des Netfilter Softwareprojekts.

Iptables basiert auf der Stateful Packet Inspection Filtertechnologie.

Das Netfilter Softwareprojekt stellt unter anderem Paketfilter wie iptables, NAT, PAT und weitere für Firewalls relevante Werkzeuge bereit.

Zu finden unter: <http://www.netfilter.org/>

Es können drei verschiedene Filtertabellen mit Hilfe von iptables bearbeitet werden:

- filter
- nat
- mangle

Maschinen, die keine Routing Tätigkeit ausüben, verwenden im Normalfall nur die filter Tabelle. Ich werde im Folgenden nur auf die filter Tabelle eingehen.

Aufbau und Funktionsweise der „filter“ Tabelle

Es gibt mindestens drei chains (Ketten) in der filter Tabelle, die jeweils unter bestimmten Umständen durchlaufen werden:

- INPUT (Ziel: lokaler Prozess)
- FORWARD (Ziel: nicht lokal, Paket hat externen Empfänger)
- OUTPUT (Quelle: lokaler Prozess)

Eine Grafik hierzu ist in der Präsentation auf Seite 7 zu sehen.

Es können aber auch noch manuell weitere chains erstellt werden.

Eine chain besteht aus mehreren Regeln, diese werden von IP Paketen sequentiell von oben nach unten durchlaufen. Je nachdem welches Ziel bzw. Quelle das IP Paket hat, wird dann eben eine der chains wie oben beschrieben abgearbeitet. Sollte beim Durchlauf einer chain das IP Paket auf eine Regel stoßen, dessen Beschreibung auf das IP Paket zutrifft (ein sogenannter match) wird das Paket an ein bestimmtes target weitergeleitet.

Standard targets (Ziele):

- ACCEPT (Paket darf passieren)
- DROP (Paket wird verworfen)
- REJECT (Paket wird verworfen, aber dem Absender mitgeteilt, das es verworfen wurde)

Es können aber auch selbst targets erstellt werden.

Nachdem ein match erreicht wurde, durchläuft das Paket keine Regel dieser Kette mehr.

Sollte kein match erreicht werden, greift am Ende jeder chain die policy.

Diese sollte bei den 3 standard chains idealerweise das Paket verwerfen, also eine DROP Policy sein.

Aufbau einer Regel

Auf der Seite 8 der Präsentation sehen Sie ein Beispiel für den Aufbau einer Regel. Das Beispiel aus der Präsentation besagt, dass zur INPUT chain eine Regel hinzugefügt wird, die alle auf dem Netzwerkinterface eth0 eingehenden Pakete vom Typ ICMP (Ping) mit der Quelladresse 192.168.0.0/24 und der Zieladresse 127.0.0.1 akzeptiert.

Die Syntax und alle verfügbaren Parameter können in den man pages von iptables nachgelesen werden.

Diese finden Sie im Internet z.B. unter <http://ipset.netfilter.org/iptables.man.html> oder Sie geben auf ihrem Linux System „man iptables“ ein.

Beispielkonfiguration der Linux Firewalldistribution Ipcop

Um eine ähnliche Ausgabe zu den Filtertabellen zu erhalten, müssen Sie „iptables -n -v -L“ auf ihrem Linux System eingeben.

```
Chain INPUT (policy DROP 68 packets, 5895 bytes)
pkts bytes target      prot opt in      out     source        destination
2266 251K ipac~o      all  --  *      *       0.0.0.0/0     0.0.0.0/0
2266 251K BADTCP    all  --  *      *       0.0.0.0/0     0.0.0.0/0
2266 251K CUSTOMINPUT all  --  *      *       0.0.0.0/0     0.0.0.0/0
2266 251K GUIINPUT  all  --  *      *       0.0.0.0/0     0.0.0.0/0
995 88471 ACCEPT     all  --  *      *       0.0.0.0/0     0.0.0.0/0     state RELATED,ESTABLISHED
1253 161K IPSECVIRTUAL all  --  *      *       0.0.0.0/0     0.0.0.0/0
1253 161K OPENSSELVIRTUAL all  --  *      *       0.0.0.0/0     0.0.0.0/0
4 277 ACCEPT     all  --  lo     *       0.0.0.0/0     0.0.0.0/0     state NEW
0 0 DROP      all  --  *      *       127.0.0.0/8    0.0.0.0/0     state NEW
0 0 DROP      all  --  *      *       0.0.0.0/0     127.0.0.0/8    state NEW
772 92166 ACCEPT     !icmp --  eth0    *       0.0.0.0/0     0.0.0.0/0     state NEW
477 68388 DHCPBLUEINPUT all  --  *      *       0.0.0.0/0     0.0.0.0/0
477 68388 IPSECPHYSICAL all  --  *      *       0.0.0.0/0     0.0.0.0/0
477 68388 OPENSSELPHYSICAL all  --  *      *       0.0.0.0/0     0.0.0.0/0
477 68388 WIRELESSINPUT all  --  *      *       0.0.0.0/0     0.0.0.0/0     state NEW
68 5895 REDINPUT  all  --  *      *       0.0.0.0/0     0.0.0.0/0
68 5895 XTACCESS  all  --  *      *       0.0.0.0/0     0.0.0.0/0     state NEW
10 1120 LOG      all  --  *      *       0.0.0.0/0     0.0.0.0/0     limit: avg 10/min burst 5 LOG flags 0 level 4 prefix 'INPUT '

Chain FORWARD (policy DROP 0 packets, 0 bytes)
pkts bytes target      prot opt in      out     source        destination
93 7812 ipac~fi     all  --  *      *       0.0.0.0/0     0.0.0.0/0
93 7812 ipac~fo     all  --  *      *       0.0.0.0/0     0.0.0.0/0
93 7812 BADTCP    all  --  *      *       0.0.0.0/0     0.0.0.0/0
0 0 TCPMSS     tcp  --  *      *       0.0.0.0/0     0.0.0.0/0     tcp flags:0x06/0x02 TCPMSS clamp to PMTU
93 7812 CUSTOMFORWARD all  --  *      *       0.0.0.0/0     0.0.0.0/0
17 1428 ACCEPT     all  --  *      *       0.0.0.0/0     0.0.0.0/0     state RELATED,ESTABLISHED
76 6384 IPSECVIRTUAL all  --  *      *       0.0.0.0/0     0.0.0.0/0
76 6384 OPENSSELVIRTUAL all  --  *      *       0.0.0.0/0     0.0.0.0/0
0 0 ACCEPT     all  --  lo     *       0.0.0.0/0     0.0.0.0/0     state NEW
0 0 DROP      all  --  *      *       127.0.0.0/8    0.0.0.0/0     state NEW
0 0 DROP      all  --  *      *       0.0.0.0/0     127.0.0.0/8    state NEW
22 1848 ACCEPT     all  --  eth0    *       0.0.0.0/0     0.0.0.0/0     state NEW
54 4536 WIRELESSFORWARD all  --  *      *       0.0.0.0/0     0.0.0.0/0     state NEW
0 0 REDFORWARD all  --  *      *       0.0.0.0/0     0.0.0.0/0
0 0 PORTFWACCESS all  --  *      *       0.0.0.0/0     0.0.0.0/0     state NEW
0 0 LOG      all  --  *      *       0.0.0.0/0     0.0.0.0/0     limit: avg 10/min burst 5 LOG flags 0 level 4 prefix 'OUTPUT '
```

Regeln der INPUT chain