

# Firewall iptables



# Table of Contents

1 Einführung.....	5
1.3 Aufgaben einer Firewall.....	5
2 Arten von Firewalls.....	7
2.1 Paketfilter Firewall .....	7
2.2 Proxy Firewall.....	7
3 IP-Adresse.....	8
3.2 Netzmaske.....	8
4 Router.....	9
5 Quality of Service.....	9
6 NAT – Masquerading.....	9
7 Ports.....	10
7.1 Port Mapping oder Port Forwarding.....	10
8 Proxy.....	11
9 IP Pakete.....	11
10 Protokolle.....	11
10.1 ICMP Internet Control Message Protocol.....	11
10.2 UDP User Datagramm Protocol.....	11
10.3 TCP Transmission Control Protocol.....	12
10.3.1 Der TCP Verbindungsaufbau.....	12
11 Angriffe.....	14
11.1 Spoofing.....	14
11.2 Source routed traffic.....	15
11.3 Denial of Service Angriffe.....	15
11.4 Flooding.....	15
11.5 Portscanning.....	16
11.6 Disconnects.....	16
11.7 Hacks und Exploits.....	16
12 Warum Linux ?.....	17
13 FIREWALL TOPOLOGIEN.....	18
13.1 Einfacher Paketfilter.....	18
13.2 Dual Homed Host.....	19
13.3 Architektur mit überwachten Hosts.....	20
13.4 Architektur mit überwachtem Teilnetz(mit zwei Paketfiltern).....	20

13.5	Geteiltes und überwachtes Teilnetz mit Dual-Homed-Host.....	21
13.6	Architektur mit zusammengelegtem inneren und äusseren Paketfilter.....	21
13.7	Architektur mit mehreren Grenznetzen.....	22
14	Verbindungsszenarien .....	23
14.1	Eingehende Tcp/Ip Verbindungen.....	23
14.2	Ausgehende Tcp/Ip Verbindungen.....	23
14.3	Ausgewählte TCP Dienste auf die diese Regeln zutreffen.....	23
14.4	UDP Problematik.....	24
14.5	Sonderfälle.....	24
14.5.1	Nameservices (domain Port 53).....	24
14.5.2	Ftp Dienst ( ftp 21 ftp-data 20).....	25
14.5.2.1	Aktiv.....	25
14.5.2.2	Passiv.....	25
15	Sicherheitspolitik.....	26
15.1	Was sollte eine Sicheheitspolitik enthalten?.....	26
15.2	Was sollte nicht Teil der Sicherheitpolitik sein?.....	27
15.3	Beispiel.....	27
15.4	Reagieren auf Zwischfälle.....	28
15.4.1	Bewerten Sie die Lage.....	28
15.4.2	Beginnen Sie mit der Dokumentation.....	28
15.4.3	Unterbrechen Sie die Verbindung oder fahren Sie den Rechner runter.....	28
15.4.4	Analyse und Reaktion.....	28
15.4.5	Ertstellen eines Schnappschuss.....	29
15.4.6	Reparatur.....	29
16	Die Verwendung von Firewalls mit iptables.....	30
17	Das Konzept .....	30
18	Die Adaption.....	33
18.1	Begriffe.....	35
18.3	Iptables Optionen.....	37
18.4	Tables.....	37
18.5	Struktur.....	37
18.11	Übereinstimmungsoptionen.....	42
18.12.1	Kontrolle des Verbindungszustandes .....	44
18.12.2	state .....	44
18.12.3	limit.....	46

18.12.4 multiport.....	46
18.12.5 Mac.....	46
18.12.6 Die Mangle.....	47
18.13 NAT Tabelle.....	48
18.14 Auflistungsoptionen.....	49
18.15 Zusätzliche Ressourcen.....	49
18.16 Bedeutung einer Logfilezeile .....	50
18.17 Kernelparameter.....	51
19 ICMP TYPES AND CODES.....	52
1 IPTABLES MANUAL.....	53
2 Hilfreiche Websites.....	62
3 Exploits.....	63

# 1 Einführung

Dieses Dokument behandelt die Einrichtung einer Paketfilter Firewall unter dem Betriebssystem Linux. Nach einer kurzen Einführung und der Erklärung wichtiger Begriffe folgt ein Überblick über mögliche Angriffe. In den weiteren Kapiteln folgt die Einrichtung der Firewall sowie weiterer Sicherheitsvorkehrungen.

## 1.1 Was ist eine Firewall ?

Das Wort "Firewall" stammt aus der englischen Sprache und bedeutet "Feuerwand". Diese Feuerwand ist eine Struktur, die die Ausbreitung eines Feuers verhindern soll. Das Wort "Firewall" stammt aus der englischen Sprache und bedeutet "Feuerwand". Diese Feuerwand ist eine Struktur, die die Ausbreitung eines Feuers verhindern soll. Gebäude haben Firewalls, die verschiedene Bereiche des Gebäudes komplett voneinander trennen.

In einem Auto übernimmt die Metallplatte, welche den Motor von der Passagierkabine trennt, die Aufgaben einer Firewall. In der Informatik wird Firewall als Bezeichnung für einen Computer benutzt, der die Schnittstelle zwischen Computernetzen darstellt, und gleichzeitig bestimmte Bereiche der Computernetze vor Angriffen und unerwünschten Zugriffen schützt.

## 1.2 Warum wird eine Firewall benötigt ?

Die Zeiten, in denen Viren, die offline per herkömmlichen Datenträgern eingeschleppt wurden, die einzige Bedrohung für Computer darstellte, sind vorbei.

Durch die zunehmende Vernetzung von Firmen, Privathaushalten, Schulen, anderen öffentlichen Einrichtungen, und deren Anbindung an das Internet, müssen diese lokalen Netze vor Angriffe aus dem Internet, geschützt werden. Doch auch Angriffe aus dem eigenen Netz sind keine Seltenheit. Der Angreifer sitzt häufig in den eigenen Reihen. Ohne eine Firewall bemerken die Netzadministratoren die Einbrüche durch Angreifer aus anderen Netzen oder dem Internet oftmals nicht. Eine Firewall kann jedoch nur vor Angriffen schützen, wenn sämtlicher Datentransfer über die Firewall läuft. Die Firewall muß als Schnittstelle zwischen den angrenzenden Netzen dienen. Ist es dem Angreifer möglich, den Firewallrechner zu umgehen, so hat diese Ihr Ziel verfehlt, da sie keinen ausreichenden Schutz darstellt.

Beim Betrieb einer Firewall muß die gesamte Sicherheitsphilosophie beachtet werden. Eine Firewall ist nur ein Glied in einer Kette von Sicherheitsvorkehrungen, und bietet allein noch keinen ausreichenden Schutz vor Angriffen.

## 1.3 Aufgaben einer Firewall

Eine Firewall soll heutzutage eine Vielzahl von Aufgaben wahrnehmen, von denen der Schutz des lokalen Netzes die Wichtigste, aber bei weitem nicht die Einzige ist. Folgende Liste erhebt keinen Anspruch auf Vollständigkeit:

Eine Firewall soll

das eigene Netz gegenüber dem Internet absichern

Angriffsversuche soweit möglich melden

Angriffe verhindern

Den Zugang ins Internet regeln (nach Zeit, Benutzer oder Rechner)

NAT bzw. Masquerading unterstützen

Zugriffe auf bestimmte Seiten sperren (z.B. Sex-Sites)

HTTP- und FTP Zugriffe zwischenspeichern

Statistiken über benutzte Dienste und Webzugriffe liefern.

sich für Mitarbeiter transparent verhalten

VPN mit mehreren Standorten ermöglichen

fernadministrierbar sein

ausfallsicher sein

sich schnell und verzögerungsfrei verhalten

einfach zu bedienen und zu warten sein

## 2 Arten von Firewalls

Man unterscheidet folgende Firewalltypen:

### 2.1 Paketfilter Firewall

Sie ist das Thema dieses Dokuments. Eine Paketfilter Firewall analysiert den gesamten Datenverkehr im Netzwerk über die Informationen im Header eines jeden Paketes. An Hand von aufgestellten Regeln (was darf sein, was nicht) wird entschieden, ob das Paket seinen Weg fortsetzen darf, oder ob es geblockt wird.

### 2.2 Proxy Firewall

Ein Proxy wird genutzt, um ausgehenden Datenverkehr zu überwachen. Alle Verbindungen werden protokolliert. So ist es jederzeit möglich zu prüfen, was für Verbindungen zu einem bestimmten Zeitpunkt aktiv waren und von welcher Quelle diese Verbindungen ausgelöst wurden. Zusätzlich bieten viele Proxies die Möglichkeit, Daten zwischenspeichern, um Übertragungsvolumen zu sparen und den Zugriff auf die Daten zu beschleunigen. Alle Daten eines per Proxy angebotenen Dienstes werden an eine Applikation auf dem Firewallrechner weitergeleitet, die dann die Verbindung zum Ziel übernimmt. So bleiben die eigentlichen Quellen der Kommunikation geschützt im Verborgenen. Alle Anfragen scheinen vom Proxyserver zu kommen.

Auf diese Art von Firewall (auch Application Level Gateway genannt) soll in diesem Dokument nicht näher eingegangen werden.

## 3 IP-Adresse

Jeder Rechner, der an das Internet angeschlossen ist, erhält eine weltweit eindeutige Kennung, die Internet Protokoll Adresse, kurz IP-Adresse.

Die IP-Adresse ist eine 32bit breite Binärzahl. Die einzelnen Bytes werden durch einen Punkt getrennt. Die vorderen Bits bezeichnen das Subnetz, in dem sich der Rechner befindet. Die restlichen Bits unterscheiden die einzelnen Rechner innerhalb dieses Subnetzes.

Damit lassen sich in der aktuellen Version des Internet Protokolls (Ipv4)  $2$  hoch  $32$ , also 4,294967296 Milliarden Rechner eindeutig bestimmen. Allerdings steht nicht der gesamte IP-Adreßbereich zur Verfügung, da einige Bereiche für andere Zwecke reserviert sind:

### 3.1.1 Privater Adreßbereich der Klasse A

Dieser Adreßbereich reicht von 10.0.0.0 bis 10.255.255.255 und ist für lokale Netze reserviert.

### 3.1.2 Privater Adreßbereich der Klasse B

Dieser Adreßbereich reicht von 172.16.0.0 bis 172.31.255.255 und ist ebenfalls für lokale Netze reserviert.

### 3.1.3 Privater Adreßbereich der Klasse C

Ein weiterer Bereich für lokale Netze, der von 192.168.0.0 bis 192.168.255.255 geht.

### 3.1.4 Multicast-Adreßbereich der Klasse D

Dieser IP-Adreßbereich ist für Multicast-Anwendungen reserviert. Dabei handelt es sich um Audio- oder Videoanwendungen mit einer Quelle und vielen Empfängern. Der reservierte Bereich reicht von 224.0.0.0 bis 239.255.255.255.

### 3.1.5 Reservierter Adreßbereich der Klasse E

Hierbei handelt es sich um einen Bereich, der für zukünftige und experimentelle Zwecke freigehalten wird. Er erstreckt sich von 240.0.0.0 bis 247.255.255.255. Es gibt noch einige weitere Blöcke, die von der IANA, Internet Assigned Numbers Authority, reserviert sind.

## 3.2 Netzmaske

Durch die Netzmaske ist festgelegt, wie viele Bits einer IP-Adresse das Subnetz bilden. Jedes Subnetzbit erhält eine Eins, jedes Hostbit eine Null. Eine Netzmaske wird wie eine IP-Adresse durch vier Bytes angegeben: 255.255.255.0 beispielsweise bedeutet, daß die ersten 24 Bit der IP-Adresse dem Subnetz zuzuordnen sind, während die letzten 8 Bit die einzelnen Rechner im Subnetz unterscheiden. Neben dieser Schreibweise ist es auch sehr verbreitet, die Netzmaske als Anzahl der Subnetz-Bits anzugeben. Bei einer IP-Adresse wie 192.168.1.15/24 beispielsweise stehen die ersten 24 Bits für das Subnetz und die letzten 8 Bits kennzeichnen den Host.

## 4 Router

Router haben die Aufgabe Netzwerkpakete weiterzuleiten. Ein IP-Paket erreicht sein Ziel über eine Reihe von Routern, die das Paket an einem Netzwerkinterface entgegennehmen und nach den Angaben ihrer Routertabellen an ein anderes Interface weiterleiten.

## 5 Quality of Service

Jedes IP-Paket enthält ein Feld, das die Priorität des Pakets angibt, das sogenannte Quality of Service Feld (abgekürzt QoS). Erreichen einen Router mehr Pakete als er gleichzeitig verarbeiten kann, wird die Verarbeitungsreihenfolge durch Hilfe dieses QoS-Feldes festgelegt: Als erstes werden Pakete mit hoher Priorität abgearbeitet, dann folgen die niedrigeren Prioritäten. Das QoS-Feld eines Pakets kann modifiziert werden, so lassen sich z.B. bestimmte Pakete bevorzugt behandeln.

## 6 NAT – Masquerading

Haben die Rechner des eigenen Netzes nicht registrierte IP-Adressen (aus einer der für lokale Netze reservierten Klasse A, B oder C), sind direkte Verbindungen ins Internet nicht möglich, da diese von den Routern nicht weitergeleitet werden.

Diese nur lokal gültigen Adressen müssen also durch offizielle IP-Adressen ersetzt werden. Dieser Vorgang nennt sich Network Address Translation (NAT) oder Masquerading.

Hier ersetzt der Rechner mit der Verbindung ins Internet alle ankommenden Pakete aus dem LAN mit seiner eigenen, gültigen IP-Adresse. Informationen über die einzelnen maskierten Verbindungen werden in Routingtabellen gespeichert, um anschließend ankommende Pakete, die für das interne Netz bestimmt sind (von aus dem LAN initiierten Verbindungen), wieder zu demaskieren.

Während beim normalen NAT  $m$  lokale Adressen auf  $n$  offizielle maskiert werden ( $m:n$  Zuordnung) ist das Masquerading ein Sonderfall von NAT. Hier werden alle lokalen Adressen auf eine gültige IP-Adresse maskiert ( $m:1$  Zuordnung).

Die Vorteile von NAT:

Internetanbindung des gesamten lokalen Netzes über eine IP-Adresse.

Zusätzlicher Schutz, da Rechner im LAN für das Internet nicht sichtbar sind.

## 7 Ports

Viele Server im Internet bieten gleichzeitig mehrere Netzwerkdienste an. Erreicht ein Datenpaket den Server, muß dieser herausfinden, für welchen Dienst die Daten bestimmt sind. Dies erreicht man durch den Einsatz von Ports. Jedem Netzwerkdienst ist ein spezieller Port zugeordnet.

Hier nun einige wichtige Portnummern:

Portnummer Dienst

20,21 FTP

23 Telnet

25 SMTP

53 DNS

80 HTTP

110 POP3

119 NNTP

143 IMAP

443 HTTPS

Ports mit Nummer unter 1024 werden als privilegierte Ports bezeichnet. Diese Ports werden von den verschiedenen Servern benutzt. Die Server warten hier auf Verbindungen aus dem Netz. Ports zwischen 1024 und 65535 werden als unprivilegierte Ports bezeichnet. Sie werden von Clients für Verbindungen genutzt.

Zusätzlich sind Ports zwischen 1024 und 49151 von der IANA registriert.

Sie können entweder als normale unprivilegierte Ports benutzt werden, oder hier läuft ein bestimmter Dienst, wie z.B. das X-Windows System, die grafische Benutzeroberfläche unter Linux. Die ursprüngliche Idee war, auf diesen höheren Ports Dienste laufen zu lassen, die nicht über root-Privilegien verfügen.

### 7.1 Port Mapping oder Port Forwarding

Port Mapping um eigene Netzwerkdienste hinter einer Firewall zur Verfügung zu stellen. Alle auf der Firewall eingehenden Anfragen auf einen bestimmten Port werden an einen bestimmten Rechner im LAN weitergereicht (deswegen auch Port Forwarding).

Beispielsweise werden alle auf der Firewall eingehenden Pakete auf dem Port 80 an den Webserver im lokalen Netz weitergeleitet, alle Pakete auf Port 110 gehen auf den POP3 Mailserver, der sich ebenfalls im LAN befindet. Die Server im LAN genießen so einen zusätzlichen Schutz.

Durch Port Mapping lassen sich auch transparente Proxies realisieren.

## 8 Proxy

Ein Proxy ist ein Rechner, der für einen Client eine Verbindung zu einem bestimmten Dienst aufbaut. Proxies existieren für unterschiedliche Dienste, wie z.B. HTTP und FTP. Die Daten werden vom Proxy angefordert und an den Client weitergereicht. Dadurch bleibt der Client im Verborgenen. Sämtliche Anfragen scheinen vom Proxyserver zu kommen. Zusätzlich dient ein Proxy als großer Zwischenspeicher: Daten, die von Clients angefordert werden und bereits auf dem Proxy liegen, müssen von diesem nicht noch einmal angefordert werden. Dadurch kann die Geschwindigkeit der Übertragung gesteigert werden.

Viele Internetprovider bieten ihren Kunden einen eigenen Proxyserver für FTP und HTTP an.

## 9 IP Pakete

Das IP Protokoll definiert eine Nachricht, die zwischen zwei Computern im Netzwerk gesendet wird. Eine solche Nachricht wird Paket genannt. Ein Paket ist also eine einzelne Nachricht, die im Netz gesendet wird.

Ein IP Paket besteht aus einem Paketkopf (message header) und dem Nachrichtenkörper (message body). Der Körper enthält die eigentlichen Daten, die ausgetauscht werden.

## 10 Protokolle

### 10.1 ICMP Internet Control Message Protocol

ICMP Pakete werden zu Kontrollzwecken und für Statusmeldungen benutzt. Sie werden lediglich zwischen zwei Computern ausgetauscht, nicht zwischen Diensten, die auf dem Computer laufen. Der Header enthält

Quell- und Zieladresse

Art der Kontrollnachricht

ICMP Nachrichtentyp (Kommando, Statusinformation, Fehlermeldung)

### 10.2 UDP User Datagramm Protocol

UDP Pakete werden zwischen zwei Netzwerkdiensten ausgetauscht. Bei UDP Paketen gibt es keine Benachrichtigung darüber, ob ein Paket erfolgreich empfangen wurde. UDP ist also ein verbindungsloses Protokoll, und damit ist nicht sichergestellt, daß ein UDP Paket sein Ziel erreicht. Dies läßt sich am Besten mit dem Versand einer Postkarte vergleichen. Auch hier ist nicht sichergestellt, daß diese auch jemals ihr Ziel erreicht. Der daraus entstehende Vorteil ist, daß UDP Pakete um ein vielfaches schneller sind als TCP Pakete. Der Header enthält:

Quell- und Zieladresse

Quell- und Zielporntnummer

UDP Protokolltyp

## 10.3 TCP Transmission Control Protocol

TCP ist das am meisten genutzte Protokoll für Netzwerkdienste, da bei TCP eine Verbindung zwischen den Computern während der gesamten Dauer des Datenaustausches sichergestellt ist. Für jedes empfangene TCP Paket wird eine Empfangsbestätigung an den Absender geschickt. So werden Übertragungsfehler, verlorene Datenpakete oder Duplikate erfolgreich vermieden. Der Header enthält

Quell- und Zieladresse

Quell- und Zielportnummer

TCP protocol message type

Sequence acknowledgement number

Control flags (SYN, ACK, FIN)

Eine TCP Verbindung lässt sich am Besten mit einem Telefongespräch vergleichen. Eine Verbindung zum Informationsaustausch ist hier sichergestellt. Kommen am anderen Ende der Leitung nicht alle gewünschten Informationen an (z.B. durch Rauschen in der Leitung) wird einfach erneut um diese gebeten.

### 10.3.1 Der TCP Verbindungsaufbau

Der Verbindungsaufbau bei TCP läuft in folgenden Schritten ab:

1) Ein Webserver läuft auf einem Computer im Internet und wartet auf Verbindungen auf dem TCP-Port 80. Ein Benutzer greift auf den Webserver zu, in dem er eine Webseite dieses Servers abrufen will. Er gibt dazu die URL in seinen Browser ein. Die URL wird durch einen Nameserver in die IP Adresse des Hosts aufgelöst. Der Browser belegt nun einen unprivilegierten Port, z.B. 24000, und sendet eine Verbindungsanfrage an den Webserver. Dieses TCP Paket hat z.B. folgenden Header:

Protokoll: TCP  
Quelladresse: 114.115.12.11  
Quellport: 24000  
Zieladresse: 195.20.202.1  
Zielport: 80  
Flags: SYN (connection synchronization request)

Das SYN Flag wird immer bei einem Verbindungsaufbau gesendet. Diesem SYN Flag folgt eine Synchronisations Sequenz Nummer, die vom Client vergeben wird (hier beispielsweise 14000).

2) Der Webserver empfängt das Paket des Clients. Er antwortet auf das SYN Flag mit einer Bestätigung, dem ACK Flag. Der Server teilt dem Client dadurch mit, daß er seine Verbindungsanfrage empfangen hat. Diesem ACK Flag folgt die Synchronisations Sequenz Nummer des Clients, die vom Server um eins erhöht wurde (14001). Außerdem sendet der Server nun ein weiteres SYN Flag, mit dem er ebenfalls einen Verbindungsaufbau bekanntgeben will. Der Server gibt diesem Flag seine eigene Synchronisations Sequenz Nummer (z.B. 34008). Die Verbindung ist nun halb geöffnet.

Dies ist der Header des vom Server zurückgesendeten TCP Pakets:

Protokoll: TCP  
Quelladresse: 195.20.202.1  
Quellport: 80  
Zieladresse: 114.115.12.11  
Zielport: 24000  
Flags: ACK (acknowledgement), SYN (connection synchronization request)

3. Der Client Rechner empfängt das Paket des Servers und antwortet auf das SYN Flag des Servers mit einem ACK Flag und einer um eins erhöhten Synchronisations Sequenz Nummer, die dem Flag vom Server zugewiesen wurde (34009). Von nun an werden keine weiteren SYN Flags mehr benötigt, die Server verständigen sich nur noch über ACK Flags.

Der Header des vom Client gesendeten Pakets:

Protokoll: TCP  
Quelladresse: 114.115.12.11  
Quellport: 24000  
Zieladresse: 195.20.202.1  
Zielport: 80  
Flags: ACK

Die beiden Flags SYN und ACK sind sehr wichtig für das Aufsetzen einer Firewall. Das SYN Flag wird gesetzt, wenn ein Client und ein Server die ersten beiden Pakete beim Verbindungsaufbau austauschen. Alle weiteren Pakete haben nur das ACK Flag gesetzt. Beim Beenden einer Verbindung gibt es noch weitere Flags, die jedoch für eine Paketfilter Firewall nicht zur Auswertung zur Verfügung stehen und deshalb nicht von Interesse sind.

# 11 Angriffe

An das Internet angeschlossene Computer sind einer Vielzahl von Angriffen ausgesetzt, die verschiedene Absichten verfolgen. Manche Angriffe versuchen den Zielrechner zu beschäftigen, daß dieser keinen anderen Aufgaben mehr nachgehen kann, und somit für andere Computer im Netzwerk unerreichbar ist. Andere haben das Ziel, auf dem angegriffenen Computer Lese- und Schreibrechte zu erhalten, um Daten zu stehlen, manipulieren oder zu löschen. Wieder Andere sammeln lediglich Informationen über das Zielsystem, die sich später bei weiteren Angriffen vorteilhaft auswirken können.

In diesem Kapitel sollen Techniken und häufig auftretende, bekannte Angriffe kurz vorgestellt werden.

## 11.1 Spoofing

Beim IP-Spoofing wird die Quelladresse eines Pakets modifiziert und gefälscht (gespoofed). Das Paket erhält eine falsche Identität. Es gibt keinen verlässlichen Schutz gegen IP-Spoofing. Man kann sich nie sicher sein, ob ein Paket das ist, was es vorgibt zu sein.

Ein gefälschtes Paket läßt sich dadurch leicht in ein geschütztes lokales Netz einschleusen.

IP-Spoofing wird beispielsweise von Angreifern verwendet werden, um von sich abzulenken, und einem Anderen den Angriff anzulasten.

Außer dem IP-Spoofing gibt es noch weitere Arten von Spoofing, nämlich

DNS- und Web-Spoofing. Auf diese soll jedoch hier nicht näher eingegangen werden.

## 11.2 Source routed traffic

Normalerweise wird der Weg (die Route), die ein IP-Paket zurücklegt, um von Punkt A nach Punkt B zu gelangen, durch die Router zwischen den beiden Endpunkten festgelegt. Das Paket gibt lediglich das gewünschte Ziel vor, nicht welchen Weg es gehen will.

Es ist möglich, Informationen über die Route, die das Paket gehen will, im Paket zu speichern. Die Router werten diese Informationen aus und leiten das Paket entsprechend weiter.

Hiermit lassen sich von einem Angreifer Pakete generieren, die vorgeben aus dem eigenen lokalen Netz (hinter der Firewall) zu stammen. Liegt das Ziel innerhalb des lokalen Netzes wird die Firewall diese Pakete auch dorthin senden. Dem Angreifer wäre es dadurch gelungen, die Firewall zu umgehen.

Source routed traffic sollte deshalb grundsätzlich nicht zugelassen werden.

Anmerkung: Source routed traffic läßt sich schon auf Kernelebene deaktivieren.

## 11.3 Denial of Service Angriffe

Denial of Service (DoS) Angriffe binden Systemressourcen des Zielrechners. Dies kann im schlimmsten Fall dazu führen, daß das System nicht mehr erreichbar ist. DoS Angriffe lassen sich in Floods (binden Bandbreite, nutzen Rechenzeit des Ziels) und Disconnects unterteilen.

## 11.4 Flooding

Beim Flooding (Fluten) wird das Ziel mit Unmengen von ICMP- (gewöhnlich) oder auch UDP-Paketen beschossen (überflutet). Die Folgen gehen bis zur Unerreichbarkeit des Ziels, da dieses vollständig mit der Bearbeitung der Pakete ausgelastet ist.

Bsp. Ping Flooding: Das Ziel wird mit Ping-Paketen (ICMP Message Type 8) beschossen und wird damit ausgelastet, die Ping-Pakete mit Echo-Reply-Paketen (ICMP Message Type 0) zu beantworten.

Bsp. Identification Flooding (Inetd): Dieser Angriff ähnelt einem gewöhnlichen ICMP Flood, es werden aber zusätzlich Informationen von TCP Port 113 angefordert. Dieser Angriff verbraucht mehr Rechenzeit als ein gewöhnlicher ICMP Flood, da die Antwort generiert werden muß.

Bsp. TCP SYN Flooding: Das SYN Flooding macht sich dem gewöhnlichen dreistufigen TCP Verbindungsaufbau zunutze. Das Ziel wird mit gespoofen IP-Paketen beschossen, die eine Verbindung zu einem TCP basierten Dienst (z.B HTTP) aufbauen wollen. Es wird ein Paket mit gesetztem SYN Flag gesendet. Das Ziel beantwortet dieses Paket mit einer Nachricht, die SYN- und ACK-Flag gesetzt hat. Da die IP-Adresse gespoof ist, kommt es nie zu einer Antwort mit gesetztem ACK-Flag des Angreifers. Die TCP Verbindung bleibt in Ihrem halb geöffneten Zustand und verbraucht Systemressourcen, bis es zu einem Timeout der Verbindung kommt. Da neue Pakete schneller eintreffen, als alte Verbindungen durch den Timeout beendet werden, sind nach kurzer Zeit sämtliche Ressourcen des Ziels belegt. Es können keine weiteren Verbindungen beantwortet werden. Das Ziel ist damit ausgeschaltet.

Bsp. SMTP Session Hijacking: Hier wird der Mailserver des Ziels mit Mails überflutet, bis dieser für andere Computer nicht mehr erreichbar ist.

## 11.5 Portscanning

Beim Portscanning werden die Ports des Ziels auf vorhandene Dienste geprüft. Dadurch erhält der Angreifer Informationen über mögliche Angriffspunkte auf dem Zielsystem. Portscans dienen der Informationsgewinnung, verbrauchen aber auch Systemressourcen. Sie sind damit ein weiterer DoS Angriff, wenn auch weniger effizient wie andere DoS Formen. Portscans gehen häufig wirklichen Angriffen voraus.

Portscans beschränken sich häufig auf einige wenige, ausgesuchte Zielports, auf denen verbreitete Netzwerkdienste laufen. Diese Dienste haben in der Vergangenheit oft Schwachstellen in der Sicherheit aufgezeigt und sind ein idealer Ausgangspunkt für Angriffe. Portscanning gilt trotz dieser für die Informationsgewinnung wichtigen Funktion als klassischer DoS Angriff, da eine entsprechende Anzahl an Scans gleichzeitig den Zielrechner blockieren kann.

## 11.6 Disconnects

Durch Disconnects wird der Zugriff auf den eigenen Computer oder auf andere Rechner im Netzwerk verhindert. Bsp. ICMP Destination Unreachable: Bei diesem DoS Angriff wird ein gespooftes Destination-Unreachable-Paket (ICMP Message Type 3) an das Ziel gesendet. Das Ziel erhält die Nachricht, daß der Quellrechner nicht mehr erreichbar ist und beendet alle Verbindungen mit diesem Computer.

## 11.7 Hacks und Exploits

Ein Hack ist eine Anwendung oder ein Paket, welches Schwächen in einem Betriebssystem, einer Anwendung oder einem Protokoll ausbeutet (engl.: to exploit). Die Folgen reichen vom Absturz des Ziels über Datendiebstahl bis zum vollständigen Datenverlust.

Bsp. Boink: Der Boink Hack ähnelt anderen Hacks, wie Bonk, Teardrop oder New Tear. Durch ungültige Paketfragmente, die nicht mehr korrekt zusammengefügt werden können, wird das Ziel zum Absturz gebracht.

Bsp. Land: Beim Land Hack versucht ein gefälschtes, angeblich vom Ziel stammendes Paket eine Verbindung zum Ziel aufzubauen. Das Ziel versucht mit sich selber eine Verbindung aufzubauen und stürzt ab.

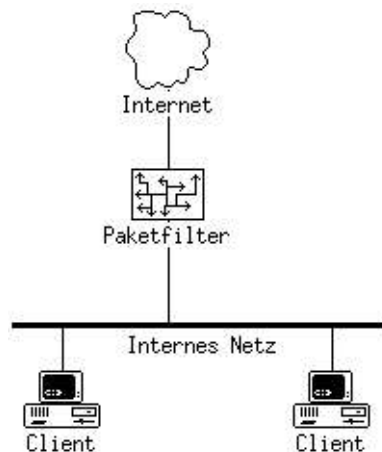
Bsp. Winnuke: Winnuke nutzt einen Fehler im TCP/IP Stack der bekannten Windows Versionen (Win3.x, Win9x, WinNT 3.51, WinNT 4) aus. Spezielle Daten (out of bound data) werden an TCP Port 139 des Ziels geschickt. Als Folge davon stürzt das TCP/IP Protokoll ab und der Computer muß neu gebootet werden, will man die Funktionalität des Protokolls wieder herstellen.

## 12 Warum Linux ?

- \* Gründe für den Einsatz von Linux als Firewall:
- \* Verfügbarkeit. Für kein anderes Betriebssystem existieren derart viele Kernel Features und Tools Verfügbarkeitsstudien zeigen die herausragende Stabilität von Linux.
- \* Ausgereifter Netzwerkteil, vergleichbar mit dem anderer Unix-Derivate.
- \* Keine weitere Software neben den Bordmitteln notwendig.
- \* Fast die gesamte Hardware ist unter Linux lauffähig.
- \* Im Gegensatz zu kommerziellen Betriebssystemen entfällt bei Linux der Support des Herstellers. Dafür aber Newsgroups, Mailing Lists und OnlineDokumentation
- \* Das gesamte Betriebssystem, inklusive Dienstprogramme und wichtiger Anwendungen, unterliegt der GPL und sind somit kostenlos und frei im Quellcode zugänglich.

## 13 FIREWALL TOPOLOGIEN

### 13.1 Einfacher Paketfilter



Einfacher Paketfilter wird das Interne Netz mit einem Router an ein öffentliches Netz angebunden. Der Router arbeitet aber selektiv. Das bedeutet, dass nur bestimmte Arten von Paketen das Passieren erlaubt wird, anderen wird es nicht gewährt. Paketfilter arbeitet auf den untersten 3 Schichten der TCP/IP-Architektur.

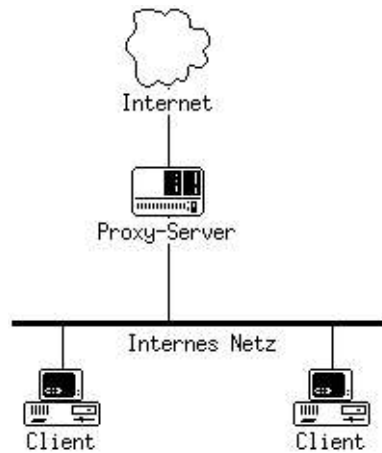
Vorteile der Paketfilterung :

- \* kann ganzes Netzwerk schützen
- \* extrem effektiv
- \* weit verbreitet

Nachteile:

- \* Filterbeschreibungssprachen sind nicht perfekt
- \* Router wird belastet
- \* Nicht alle Sicherheitrichtlinien lassen sich durchsetzen

## 13.2 Dual Homed Host



Auf einem Dual Homed Host laufen sogenannte Proxys (Stellvertreter). Ein Proxy arbeitet auf allen 4 Schichten der TCP/IP-Architektur. Das bedeutet, dass sie mit dem Anwendungsprotokoll in Berührung kommen. Wobei man zwischen 2 verschiedenen Arten unterscheiden kann:

### \* Application Level Proxys

Diese Proxyarten kennen das Anwendungsprotokoll genau. Beispielsweise squid, aber auch ein cache-only Nameserver kann man als Application Level Proxy bezeichnen. (~Dolmetscher in der realen Welt)

### \* Circuit Level Proxys

Diese Proxyarten kennen das Anwendungsprotokoll nicht. Sie „quatschen“ alles nur nach oder setzen es um. Beispielsweise socks oder delegated sind Circuit Level Proxys. (~Übersetzungsprogramm in der realen Welt)

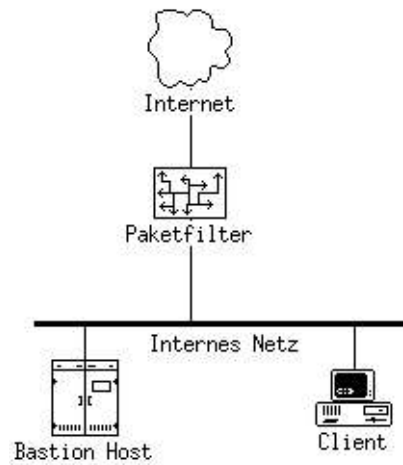
Vorteile:

- \* Gutes Protokollieren
- \* Caching
- \* Anwendungsspezifische Filterung
- \* Authentifikation auf UserEbene

Nachteile:

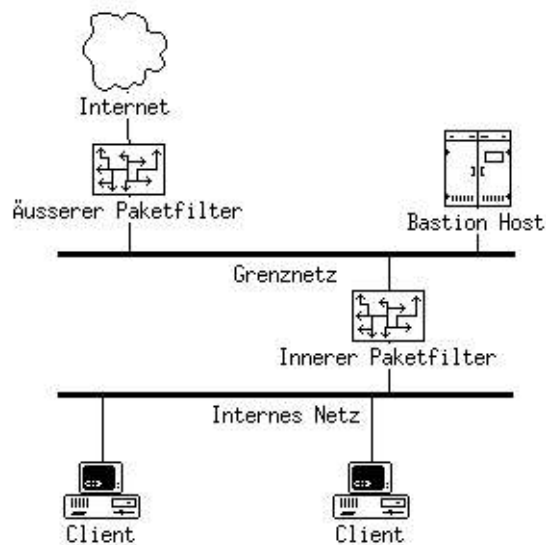
- \* Neuere Protokolle werden nicht unterstützt
- \* Verschiedene Proxyserver für verschiedene Dienste notwendig
- \* Client müssen Verhalten ändern

### 13.3 Architektur mit überwachten Hosts



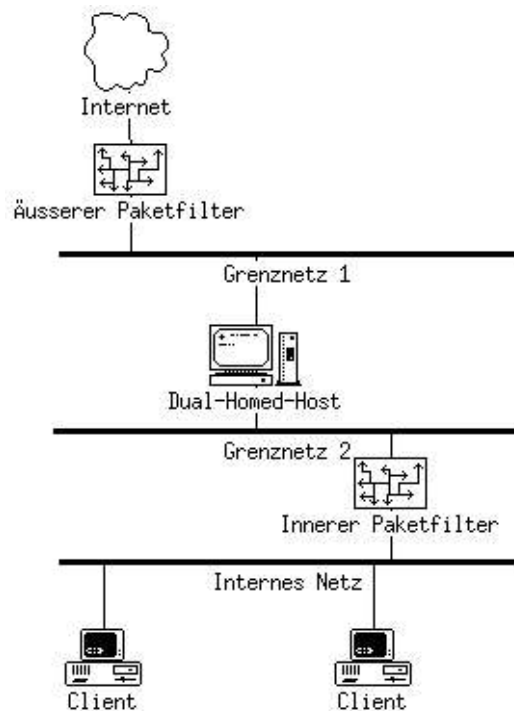
Der Paketfilter lässt nur Traffic zu dem Bastion Host zu, die Clients kommunizieren nur mit den Proxys auf dem Bastion Host.

### 13.4 Architektur mit überwachtem Teilnetz(mit zwei Paketfiltern)



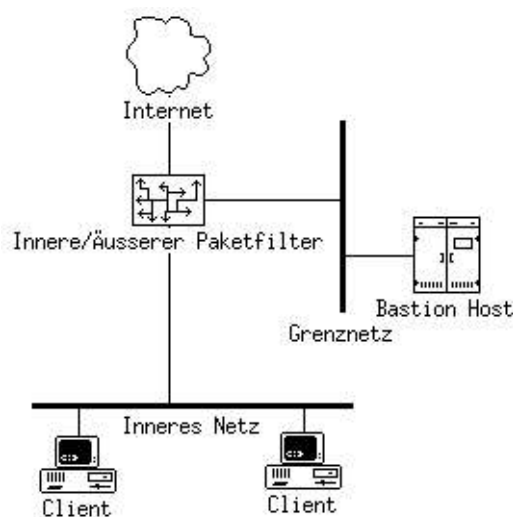
Äussere Paketfilter lässt nur Traffic zu dem Bastion Host zu. Innerere Paketfilter lässt nur Traffic zu dem Bastion Host zu die Clients kommunizieren nur mit den Proxys auf dem Bastion Host

## 13.5 Geteiltes und überwachtes Teilnetz mit Dual-Homed-Host



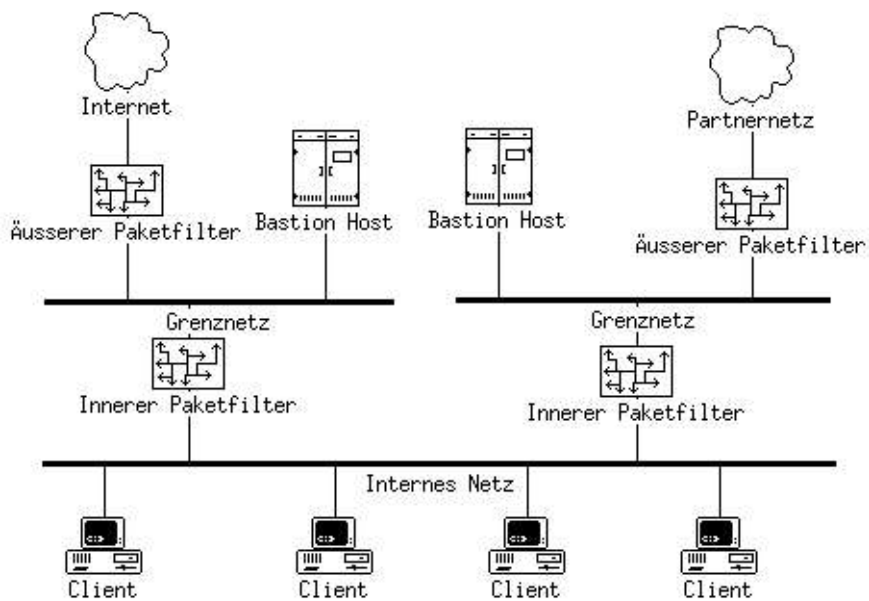
Äussere Paketfilter lässt nur Traffic zu dem Dual-Home-Host zu. Innerere Paketfilter lässt nur Traffic zu dem Dual-Home-Host zu die Clients kommunizieren nur mit den Proxys auf dem Bastion Host. Kein Forwarding auf dem Dual-Home-Host

## 13.6 Architektur mit zusammengelegtem inneren und äusseren Paketfilter



Häufig haben kommerzielle Produkte diese Layout

## 13.7 Architektur mit mehreren Grenznetzen



Äussere Paketfilter lässt nur Traffic zu dem Bastion Host zu. Innerere Paketfilter lässt nur Traffic zu dem Bastion Host zu die Clients kommunizieren nur mit den Proxys auf dem Bastion Host Das Partnernetz (z.B. Vertriebspartner) wird genauso angebunden.

## 14 Verbindungszenarien

### 14.1 Eingehende Tcp/Ip Verbindungen

Die Regel für eingehende Pakete muss alle Kombinationen von SYN und ACK Flags im TCP Header akzeptieren, das heisst, es muss auch das Paket, welches nur ein gesetztes SYN Flag hat, durchlassen.

Die Regel für ausgehende Pakete muss explizit Pakete, die lediglich das SYN Flag im TCP Header gesetzt haben, ausschliessen um zu verhindern das von innen nach aussen eine Verbindungsaufnahme erfolgen kann.

### 14.2 Ausgehende Tcp/Ip Verbindungen

Die Regel für ausgehende Pakete muss alle Kombinationen von SYN und ACK Flags im TCP Header akzeptieren. Das heisst, es muss auch das Paket, welches nur das SYN Flag gesetzt hat, durchlassen. Die Regel für eingehende Pakete muss explizit Pakete, die lediglich das SYN Flag im TCP Header gesetzt haben, ausschliessen, um zu verhindern, dass von aussen nach innen eine Verbindungs- aufnahme erfolgen kann.

### 14.3 Ausgewählte TCP Dienste auf die diese Regeln zutreffen.

Tabelle

<i>Port</i>	<i>Name</i>	<i>Beschreibung</i>
22	ssh	Secure Shell (gesicherte Fernwartung , Datenübertragung)
23	telnet	Telnet (ungesicherte Fernwartung)
25	smtp	Simple Message Transfer Protokoll (Mailserver)
80	http	Hyper Text Transfer Protokoll (Webserver)
110	pop3	Post Office Protokoll (Mailempfangsystem)
113	auth	authentication tap client ident (Authentifikations Dienst)
119	nntp	Net News Tranfer Protokoll (Newsserver)
143	imap2	Internet Message Access Protokoll (Mailempfangsystem)
443	https	Hyper Text Transfer Protokoll Secure (Webserver verschlüsselt)
515	lpd	Line Printer Daemon (Druckserver)
993	imaps	Internet Message Access Protokoll Secure (Mailempfangsystem)
995	pop3s	Post Office Protokoll Secure (Mailempfangsystem)
1080	socks	Socks Proxyserver (universeller Proxyserver)
3128	squid	Squid Proxyserver (Web Proxyserver) nicht offiziell
3130	icpv2	Internet Cache Protokoll (zur Kommnikation von Squid Proxys)

<i>Port</i>	<i>Name</i>	<i>Beschreibung</i>
3306	mysql	MySQL (Structure Query Language)
8080	webcache	Alternativer Proxydienst (WWW Caching Service)

## 14.4 UDP Problematik

Da UDP basierende Dienste verbindungslos sind, ergibt sich das Problem, dass man mit herkömmlichen Mitteln nicht bestimmen kann, welche Seite die Verbindung initiiert hat. Ausgewählte UDP Dienste auf die diese Regeln zutreffen.

<i>Port</i>	<i>Name</i>	<i>Beschreibung</i>
7	echo	Liefert das zurück was gesendet wurde.
67	bootps	Vorläufer DHCP -> hier läuft der DHCP Server
68	bootpc	Vorläufer DHCP -> Anfragen der Clients
69	tftp	Trivial File Transfer Protocoll (Dateiübertragungen ohne AUTH)
111	sunrpc	Portmapper Dienst (Basis von NFS) sehr unsicher
123	ntp	Network Time Protokoll
161	snmp	Simple Network Managment Protokoll

## 14.5 Sonderfälle

Die Nachfolgenden Dienste fallen nicht in die obengenannten Kategorien und werden deshalb gesondert behandelt.

### 14.5.1 Nameservices (domain Port 53)

Client <--> Server

Bei Anfrage eines Clients an einen Nameserver verwendet der Client als Source Port einen Wert ab 1024 und als Destination Port 53 als Protokoll wird UDP verwendet. Bei Verbindungsproblemen oder wenn das Paket grösser als 512 Byte ist erfolgt die Anfrage auf TCP Basis. Manche Unixe verwenden ausschliesslich TCP.

Server <--> Server

Bei Lookupanfrage eines Nameservers an einen Nameserver verwendet der Client als Source Port einen Wert 53, als Destination Port 53 und als Protokoll wird UDP verwendet. Seit BIND 8 verhält sich der anfragende Nameserver wie ein normaler Client (kann per Konfiguration geändert werden).

Bei Zonentransfer verwendet der Client (meist sekundäre Nameserver) den Source Port ab 1024 und der Server (meist primärer Nameserver) den Port 53 als Protokoll wird TCP verwendet.

## 14.5.2 Ftp Dienst ( ftp 21 ftp-data 20)

Beim Ftp Dienst werden zwei Kanäle zwischen Client und Server benutzt. Der Verbindungsaufbau wird wie eine ganz normale TCP Verbindung initiiert. Als Source Port wird ein Wert grösser gleich 1024 benutzt und als Destinationport 21. Diese Verbindung bleibt bestehen und wird als Steuerkanal benutzt. Danach gibt es zwei Möglichkeiten :

### 14.5.2.1 Aktiv

Beim **aktiven FTP** teilt der Client dem Server über den Steuerkanal einen Port oberhalb 1023 mit. Der Server geht darauf eine Verbindung zum Client auf diesem Port ein. Über diesen Kanal werden die Daten ausgetauscht.

### 14.5.2.2 Passiv

Beim **passiven FTP** teilt der Client dem Server über den Steuerkanal das Schlüsselwort PASV mit. Der Server überträgt darauf einen Port ab 1024 der für den Datenkanal bereit steht. Daraufhin konnektiert der Client den Server auf diesem Port. Über diesen Kanal werden die Daten ausgetauscht.

# 15 Sicherheitspolitik

Man sollte die höchstmögliche Sicherheitsstufe anstreben wobei folgende Punkte zu beachten sind:

- \* Preis                                      Wieviel Geld können/wollen sie für Sicherheit ausgeben?
- \* Funktionalität                            Können Sie Ihre Computer immer noch benutzen?
- \* Akzeptanz                                 Stören die Sicherheitseinrichtungen die Art und Weise wie die Benutzer Ihres Standorts gewöhnlich untereinander und mit der Aussenwelt kommunizieren?
- \* Juristisches                               Entsprechen die Sicherheitseinrichtungen Ihren juristischen Anforderungen?

## 15.1 Was sollte eine Sicherheitspolitik enthalten?

- \* Begründung                                Es ist wichtig das begründet wird warum Entscheidungen so getroffen wurden. - Nachvollziehbarkeit -
- \* Verantwortung                            Man sollte explizit Zuständigkeitsbereiche festlegen so das es keine Unklarheiten entstehen wer verantwortlich ist. Aber auch die Benutzer sollten auf Sicherheit achten und sich nicht nur auf die Administration verlassen.
- \* Verständlichkeit                         Man sollte die Sicherheitspolitik verständlich formulieren nur so ist gewährleistet das sie auch eingehalten wird. Was man nicht versteht kann man auch nicht einhalten.
- \* Durchsetzung                             - Der Verwalter bestimmter Dienste ist befugt Zugang zu verwehren.  
- Der Vorgesetzte kümmert sich um Überschreitungen.  
- Einrichtungen die Standards nicht erfüllen werden ausgeschlossen
- \* Berücksichtigung von Ausnahmen    Was geschieht wenn Ausnahmesituationen eintreten?
- \* Skalierbarkeit                            Was passiert wenn Ihre Netzwerk wächst?
- \* Spezielle Fragen                         - Wer erhält Zugang? Gibt es Gastzugänge?  
- Gibt es Gemeinsame Accounts? (Bsp. Email)  
- Wann verliert man einen Account?  
- Wer darf sich per Modem einwählen.  
- Was ist vor der Inbetriebnahme eines Computers zu tun?  
- Wie sicher müssen diese sein?  
- Wie werden Personaldaten geschützt?  
- Welche Passörter sollten verwendet werden? (Änderung)  
- Wer darf ausführbare Programme besorgen/installieren?  
- Welche Vorkehrungen werden gegenüber Viren getroffen?  
- Was ist mit Verbindungen mit Geschäftspartnern?

## 15.2 Was sollte nicht Teil der Sicherheitspolitik sein?

- \* Technische Details Je einfacher desto verständlicher. Es nutzt nichts wenn nur ausgebildete Techniker die Sicherheitspolitik verstehen.
- \* Probleme Anderere Abgrenzungen zu anderen Standorten <-> Leute
- \* Probleme die nichts mit Sicherheit zu tun haben. Anschauen von Pornobilder ist ein juristisches – und keine Sicherheits Problem. Leute die den ganzen Tag spielen sind ein Problem für Personalabteilung :-)

## 15.3 Beispiel

Schlecht:

Für alle eingehende Verbindungen ist OTP nach IETF basierend auf S/Key zu benutzen.

Besser

Eingehende Verbindungen von der Aussenwelt müssen mit nicht wiederverwendbaren Passwörtern authentifiziert werden, um zu verhindern, daß ein Angreifer durch Überwachen solcher Verbindungen ein wiederverwendbares Passwort aufschnappt.

Noch besser

Da normale Passwörter oft gestohlen werden, benutzen wir für unser Netz, Einmalpasswörter.

## 15.4 Reagieren auf Zwischenfälle

- \* **Regel 1**                    **Keine Panik**
- \* **Regel 2**                    **Dokumentieren Sie alles**

### 15.4.1 Bewerten Sie die Lage

- \* **Hat es der Angreifer geschafft in Ihr System einzudringen?**
- \* **Läuft der Angriff noch ?**

### 15.4.2 Beginnen Sie mit der Dokumentation

Ab dem Erkennen eines Versuches sollten die weiteren Schritte und Erkenntnisse dokumentiert werden.

### 15.4.3 Unterbrechen Sie die Verbindung oder fahren Sie den Rechner runter

Es ist immer besser die betroffene Maschinen zu trennen da durch das herunterfahren Informationen über den Angreifer zerstört werden können. Des weiteren können Sie eine ausgeschaltete Maschine nicht analysieren.

### 15.4.4 Analyse und Reaktion

- \* Man sollte Ruhe bewahren und erstmal Tief durchatmen. Hektisches handeln verschlimmert die Lage eventuell.
- \* Informieren Sie andere über den Vorfall

(INTERN)

Ein sicherlich schwere wenn auch notwendiger Schritt. Viele Administratoren füllen sich schuldig und verschweigen den Vorfall. Einem Problem sollte man sich stellen und eine vernünftige Geschäftsleitung wird das auch so sehen. Man hat danach Argumente für ein höheres Budget.

(EXTERN)

Polizei, Hersteller, Community, IETF (Internet Engineering Task Force)

## 15.4.5 Erstellen eines Schnappschusses

Gründe

- \* Wenn die Reparatur schief geht
- \* Beweissicherung
- \* Spätere Recherche wenn das Original System wieder läuft.

## 15.4.6 Reparatur

- \* Update des Systems

Ersetzen der kompromitierten Teil ist sehr schwierig da hier Kernel und Bibliotheken betroffen sein können.

- \* Neuinstallation

Oftmal die sichere und einfacherer Wahl

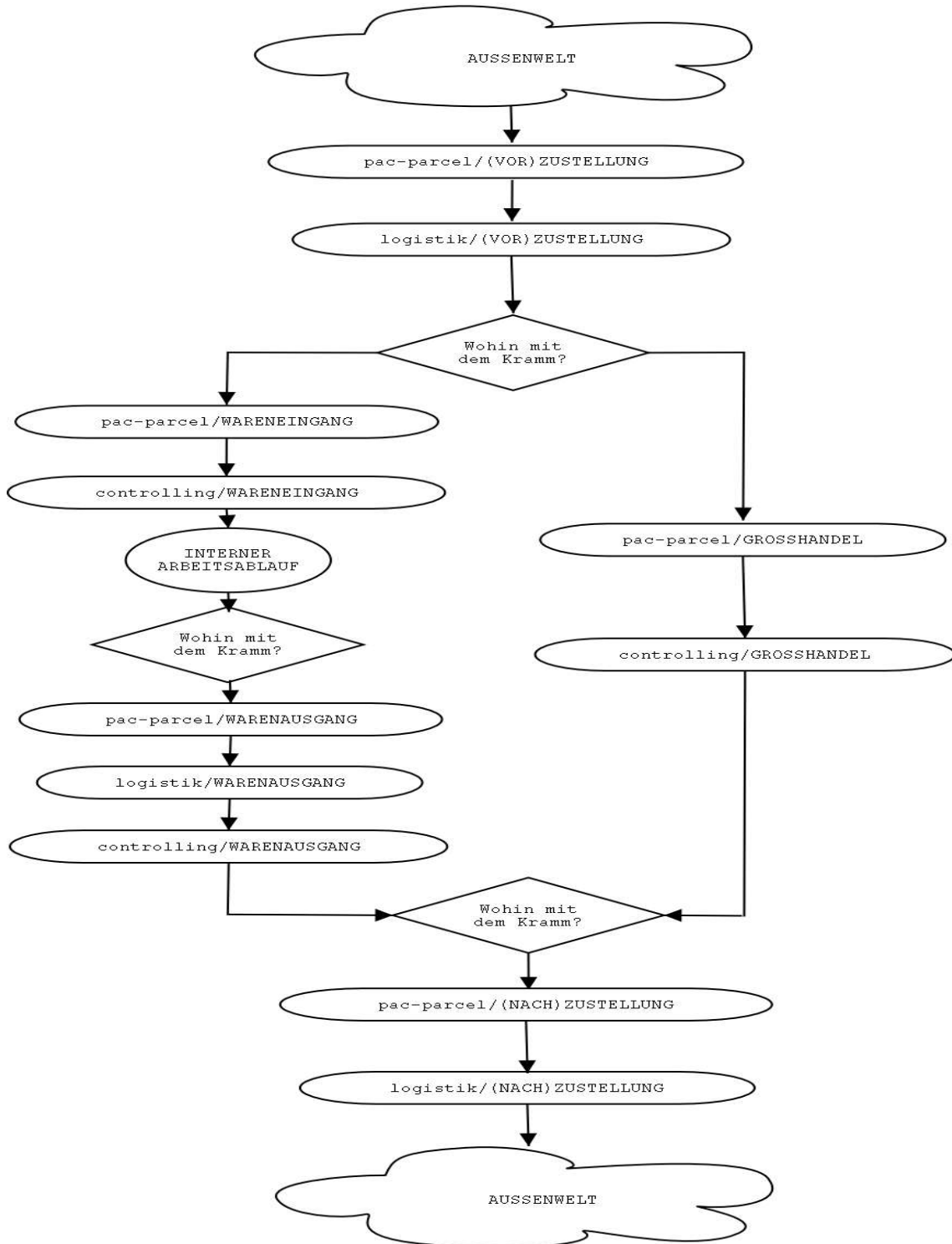
## 16 Die Verwendung von Firewalls mit iptables

Der Linux-Kernel enthält fortgeschrittene Tools für Packet-Filtering, der Prozess für die Kontrolle von Netzwerkpaketen bei ihrem Versuch einzudringen, durch und aus dem System hinaus zu dringen. Kernels vor der 2.4 Version konnten Pakete mit ipchains manipulieren, die Listen von Regeln verwendeten, die für Pakete in jeder Phase des Filterungsprozesses angewandt werden. Die Einführung des 2.4-Kernels hat iptables mit sich gebracht, die den ipchains gleichen, aber deren Wirkungsbereich und Kontrollmöglichkeiten bei der Filterung von Paketen erweitern.

# 17 Das Konzept

Am besten kann man IPTABLES mit den inneren Abläufen einer Firma vergleichen. Wir nehmen mal eine fiktive Firma von denen wir 3 Abteilungen betrachten

- \* pac-parcel -> Ver-und Entpacken
- \* logistik -> Verantwortlich für Adressierungen
- \* controlling > Kontrollieren der Ware



Die Abteilungen haben weiterer Unterabteilungen:

controlling       ->   WAREINEINGANG  
                     ->   GROSSHANDEL  
                     ->   WARENAUSGANG

Hauptaufgabe:   Kontrolle der Ware

logistik           ->   VORZUSTELLUNG  
                     ->   WARENAUSGANG  
                     ->   NACHZUSTELLUNG

Hauptaufgabe:   Wohin soll die Ware <-> Woher kam die Ware (Anbringen von Hersteller)

pac-pacel         ->   VORZUSTELLUNG  
                     ->   WAREINEINGANG  
                     ->   WARENAUSGANG  
                     ->   NACHZUSTELLUNG  
                     ->   GROSSHANDEL

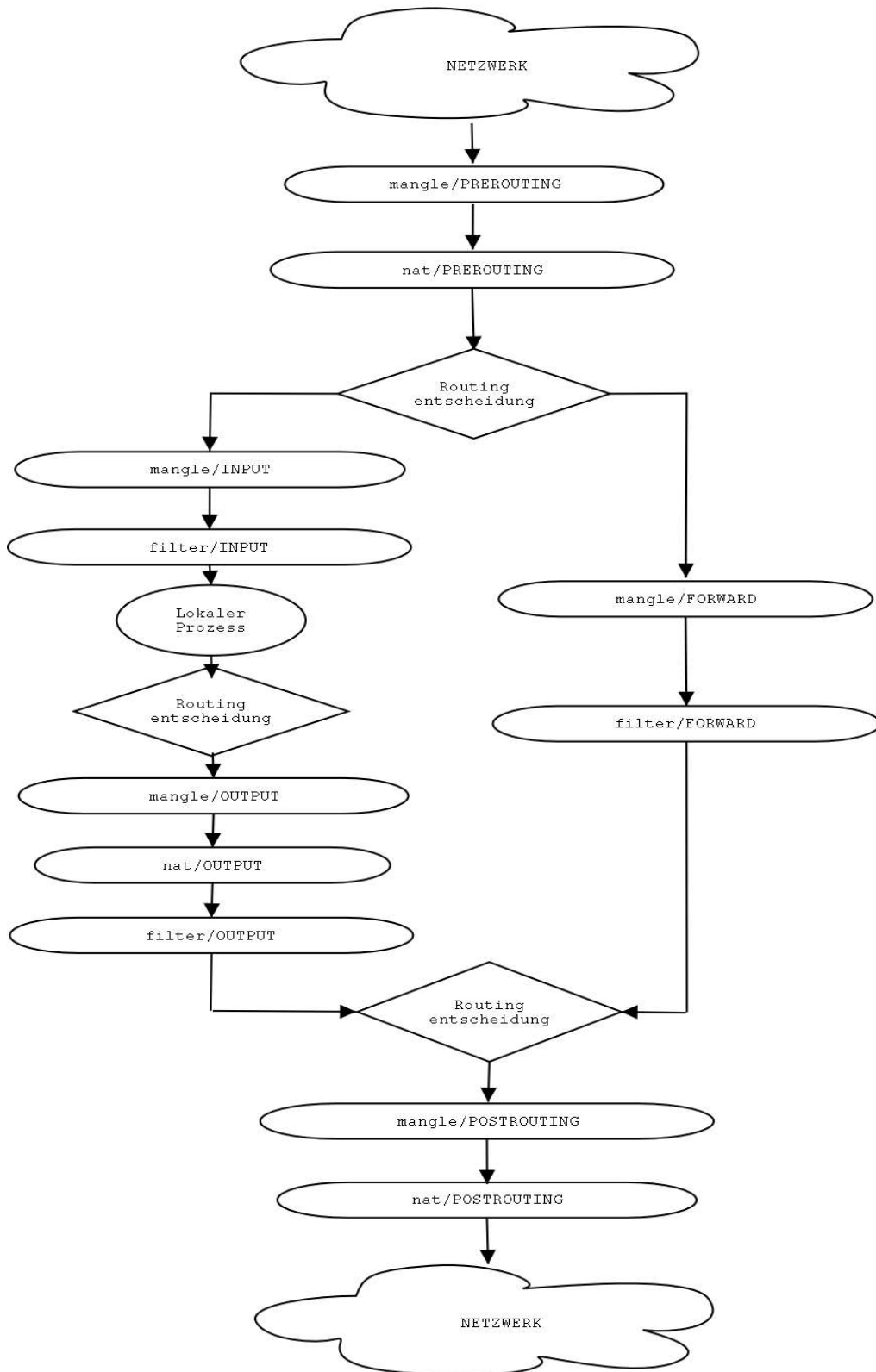
Hauptaufgabe:   Sonstige Einflussnahme auf die Ware

Genau wie bei einer normalen Firma sind die Abteilungen wieder unterteilt. Diese

Unterabteilungen müssen nicht nebeneinander liegen, sondern können durchaus an verschiedenen Orten der Firma liegen.

# 18 Die Adaption

DAS IPTABLES Konzept ist ähnlich. Abteilungen entsprechen hier den sogenannten Tabellen  
Die Unterabteilungen heißen jetzt Ketten.



Die Tabellen bestehen aus Ketten

filter           ->  INPUT  
                 ->  FORWARD  
                 ->  OUTPUT

Hauptaufgabe:  Darf ein Paket passieren oder nicht

nat              ->  PREROUTING  
                 ->  OUTPUT  
                 ->  POSTROUTING

Hauptaufgabe:  Manipulation von Adressen und/oder Ports

mangle          ->  INPUT  
                 ->  FORWARD  
                 ->  OUTPUT  
                 ->  PREROUTING  
                 ->  POSTROUTING

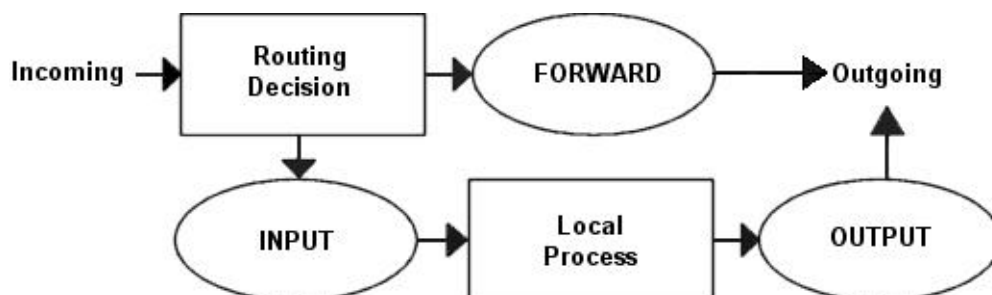
Hauptaufgabe:  Sonstige Einflussnahme auf die Pakete

Die verschiedenen Ketten werden logisch zu Tabellen zusammengefasst, wobei das unerheblich ist für die Reihenfolge der Abarbeitung

## 18.1 Begriffe

Tabelle	In Tabellen werden Ketten zusammen gefasst. Es gibt drei Arten von Tabellen filter (default), nat und mangle. Jedes Paket durchläuft verschiedene Tabellen aber nicht zwangsläufig alle. filter bedeutet das die pakete anhand einer Schablone untersucht und dann abgewiesen oder akzeptiert werden. nat steht für network address translation. Das bedeutet die Adressen des Paketes werden irgendeiner Form verändert. mangle heißt sonstige veränderungen der Kenndaten.
Kette	Ketten sind Listen von Filterregeln(Schablonen). Jedes Paket wandert von oben nach unten durch eine Kette. Sobald eine Regel exakt passt wird anhand des Zieles entschieden was mit dem Paket zu tun ist.
Ziel	In Zielen landen die Pakete nach einem Treffer. Die Ziele entscheiden über das weitere Schicksal des Paketes. Einige Standardziele sind: ACCEPT(Paket wird akzeptiert) , DROP(Paket wird verworfen) und REJECT (wie DROP es wird jedoch eine Nachricht an den Absender geschickt) Ein Ziel kann auch eine selbstdefinierte Kette sein.
Default Policy	Wenn ein Paket eine Kette komplett passiert hat entscheidet die Default Policy über das Schicksal des Paketes
Regel	Regeln kann man Schablonen vergleichen. Jedes Paket das eine Kette durchläuft wird von oben nach unten nach und nach mit jeder Regel verglichen (zB. Eine Regel kann sein alle tcp Pakete oder alle die Port 80 gesetzt haben usw.) Sobald eine Regel passt wird anhand des Zieles entschieden was mit dem Paket zu tun ist.

Ketten der Filtertabelle	
INPUT	Alle Pakete deren Ziel eine IP-Adresse des lokalen Systems ist, durchlaufen diese Kette.
OUTPUT	Alle Pakete die von einem lokalen Prozesse erzeugt worden sind, das heist die von dem lokalen System stammen durchlaufen diese Kette.
FORWARD	Alle Pakete die nicht von lokal erzeugtem Prozess stammen und deren Ziel nicht das lokale System ist durchlaufen Kette.



## 18.2 Ablauf der filter Ketten

<i>Input</i>	<i>Forward</i>	<i>Output</i>
1. REGEL -> ZIEL	1. REGEL -> ZIEL	1. REGEL -> ZIEL
2. REGEL -> ZIEL	2. REGEL -> ZIEL	2. REGEL -> ZIEL
3. REGEL -> ZIEL	3. REGEL -> ZIEL	3. REGEL -> ZIEL
.....	.....	.....
.....	.....	.....
n. REGEL -> ZIEL	n. REGEL -> ZIEL	n. REGEL -> ZIEL
DEFAULT POLICY	DEFAULT POLICY	DEFAULT POLICY

Eine Regel wird immer nach dem Schema

**iptables -A KETTE [EIGENSCHAFTEN] -j ZIEL** eingefügt. Die Eigenschaften kann man sich als Schablonen vorstellen die man feinkörnig oder grobkörnig wählen kann. Die Regeln werden von oben nach unten abgearbeitet trifft passt ein Paket in eine Schablone wird es zum Ziel zur geschickt.

Beispiele:

```
iptables -A INPUT -s 127.0.0.1 -j ACCEPT
```

Alles von der Adresse 127.0.0.1 in zu Rechner hin ist erlaubt

```
iptables -P INPUT DROP
```

Setzen der DefaultPolicy auf DROP

## 18.3 Iptables Optionen

Regeln, die es ermöglichen, dass Pakete vom Kernel gefiltert werden, werden durch Ausführen des iptables-Befehls mit einer Anzahl von nachgestellten Optionen erstellt, die den Typ der zu filternden Pakete, den Ursprung oder die Bestimmung dieser Pakete und was mit dem Paket getan werden soll wenn es der Regel unterliegt. Die in Zusammenhang mit einer bestimmten iptables-Regel verwendeten Optionen müssen auf logische Weise gruppiert werden, und zwias nach dem Zweck und den Bedingungen der allgemeinen Regel, damit die Regel gültig ist.

## 18.4 Tables

Eine leistungsstarke Eigenschaft von iptables ist, dass mehrere Tabellen verwendet werden können, um zu entscheiden, wie mit einem Paket vorgegangen werden soll, je nach dem Typ des Pakets, das kontrolliert wird und je nach dem, was mit dem Paket geschehen soll. Dank der erweiterbaren Struktur von iptables, können spezielle Tabellen erstellt und im Verzeichnis `/etc/modules/<Kernel-Version>/kernel/net/ipv4/netfilter` abgelegt werden, um bestimmten Zwecken zu entsprechen. Mit iptables können mehrfache Sätze von ipchains-Regeln in definierten Ketten enthalten, wobei jeder Satz eine bestimmte Regel erfüllt.

Die standardmäßige Tabelle, filter, enthält die standardmäßigen eingebauten INPUT-, OUTPUT und FORWARD-Ketten. Dies ist ähnlich den standardmäßigen Ketten, die mit ipchains verwendet werden. Trotzdem, enthalten iptables standardmäßig auch zwei zusätzliche Tabellen, die spezifische Paketfilterungsvorgänge ausführen.

Die Tabelle nat kann verwendet werden, um die in Paketen aufgezeichneten Ursprungs- und Bestimmungsadressen zu verändern. Mit der Tabelle mangle können Sie Pakete auf besondere Weise verändern.

Jede Tabelle enthält standardmäßige Ketten, die gemäß dem Zweck der Tabelle nötige Aufgaben ausführen, aber Sie können in jeder Tabelle auf einfache Weise neue Ketten erstellen.

## 18.5 Struktur

Viele iptables-Befehle haben die folgende Struktur:

```
iptables [-t <Tabellenname>] <Befehl> <Kettenname> <Parameter1> \  
  
<Option1> <Parameter-n> <Option-n>
```

In diesem Beispiel ermöglicht die Option <Tabellenname> dem Benutzer, eine andere Tabelle als die standardmäßige filter-Tabelle auszuwählen, die mit dem Befehl verwendet werden soll.

Die Option <Befehl> ist der Kern des Befehls, der einen bestimmten auszuführenden Vorgang hervorruft, wie zum Beispiel das Anhängen an oder Entfernen einer Regel an oder von einer Kette, die in der Option <Kettenname> spezifiziert ist. Nach dem <Kettenname> befinden sich

Parameterpaare und Optionen, die im Endeffekt bestimmen, wie die die Regel angewandt wird und was passiert, wenn ein Paket einer Regel entspricht.

Sehen wir uns die Struktur eines iptables-Befehls an, ist es wichtig zu beachten, dass im Gegensatz zu dem meisten anderen Befehlen, die Länge und Komplexität eines iptables-Befehls dessen Zweck ändern kann. Ein einfacher Befehl für das Entfernen einer Regel von einer Kette kann sehr kurz sein, während ein Befehl für das Filtern von Paketen eines bestimmten Unternetzes mithilfe verschiedener spezifischer Parameter und Optionen ziemlich lang sein kann. In iptables-Befehlen sind im Zusammenhang mit gewissen Parametern und Optionen weitere Parameter und Optionen nötig, welche die Anfrage der vorhergehenden Option erläutern. Um eine gültige Regel zu erstellen, muss diese fortfahren, bis jedes Parameter und jede Option, die eine andere Reihe von Optionen verlangt, erfüllt ist.

Geben Sie iptables -h für eine vollständige Liste der iptables-Befehlsstrukturen ein.

## 18.6 Befehle

Befehle weisen iptables an, einen bestimmten Vorgang auszuführen. Nur ein Befehl pro iptables-Befehlszeichenkette ist zugelassen. Mit Ausnahme des Hilfebefehls sind alle Befehle in Großbuchstaben geschrieben. iptables-Befehle sind:

	Bedeutung
-A	Hängt die iptables-Regel an das Ende der spezifizierten Kette an. Dies ist der Befehl, um eine Regel einfach hinzuzufügen, wenn die Reihenfolge der Regeln in der Kette nicht ausschlaggebend ist.
-C	Kontrolliert eine bestimmte Regel bevor sie in die benutzerspezifizierte Kette hinzugefügt wird. Dieser Befehl kann Ihnen helfen, komplizierte iptables-Regeln zu erstellen, in denen zusätzliche Parameter und Optionen verlangt werden.
-D	Entfernt eine Regel in einer bestimmten Kette nach Nummer (wie zum Beispiel 5 für die fünfte Regel in einer Kette). Sie können ebenfalls die gesamte Regel eingeben und iptables entfernt die entsprechende Regel aus der Kette.
-E	Benennt eine benutzerdefinierte Kette um. Dies hat überhaupt keine Auswirkung auf die Struktur der Tabelle, sondern erspart Ihnen nur das Entfernen der Kette, indem Sie mit einem neuen Namen erstellt wird, und alle Regeln für diese Kette neu konfiguriert werden.
-F	Löscht die gewählte Kette, und es wird effektiv jede Regel in der Kette entfernt. Wenn keine Kette angegeben ist, löscht dieser Befehl jede Regel jeder Kette.
-h	Liefert eine Liste von nützlichen Befehlsstrukturen, sowie eine kurze Zusammenfassung der Befehlsparameter und -Optionen.
-I	Fügt eine Regel an einem bestimmten Punkt in eine Kette ein. Ordnen Sie der einzufügen-den Regel eine Nummer zu und iptables wird sie dorthin verschieben. Wenn keine Nummer angegeben ist, fügt iptables den Befehl am Anfang der Regelliste ein.
-L	Listet alle Regeln in der nach dem Befehl spezifizierten Kette auf. Um alle Regeln in allen Ketten in der Standardtabelle filter aufzulisten, spezifizieren Sie nicht eine Kette oder eine Tabelle. Ansonsten sollte die folgende Syntax verwendet werden, um die Regeln in einer spezifischen Kette in einer bestimmten Tabelle aufzulisten: iptables -L <Kettenname> -t <Tabellename> Leistungsstarke Optionen für den -L-Befehl, die Regelnummern liefern und ausführlichere Regelbeschreibungen ermöglichen, sind unter anderem in Abschnitt namens Auflistungsoptionen erläutert.

	Bedeutung
-N	Erstellt eine neue Kette mit einem benutzerdefinierten Namen.
-P	Setzt die standardmäßige Politik für eine bestimmte Kette, damit wenn Pakete eine ganze Kette durchqueren, ohne dass sie mit einer Regel übereinstimmen, diese, wie bei ACCEPT oder DROP, an ein bestimmtes Ziel weitergeliefert werden.
-R	Ersetzt eine Regel in einer bestimmten Kette. Sie müssen eine Regelnummer nach dem Namen der Kette verwenden, um die Regel zu ersetzen. Die erste Regel einer Kette bezieht sich auf die Regelnummer 1.
-X	Entfernt eine benutzerdefinierte Kette. Das Entfernen einer eingebauten Kette für eine Tabelle ist nicht zugelassen.
-Z	Stellt Byte- und Paketzähler in allen Ketten für eine bestimmte Tabelle auf null.

**Caution:**

Achten Sie darauf, welche Option (-A oder -I) Sie beim Hinzufügen von Regeln verwenden. Die Reihenfolge der Regeln kann sehr wichtig sein, wenn Sie bestimmen, ob ein bestimmtes Paket dieser oder jener Regel entspricht. Stellen Sie sicher, dass beim Hinzufügen einer Regel am Anfang oder am Ende einer Kette, diese nicht andere Regeln in der Kette beeinflusst.

## 18.7 Parameter

Sobald gewisse iptables-Befehle spezifiziert worden sind, jene zum Hinzufügen, Anhängen, Entfernen, Einfügen oder Ersetzen von Regeln innerhalb einer Regel inbegriffen, müssen Sie Parameter definieren, um mit dem Erstellen einer Paketfilterungsregel zu beginnen.

	<i>Bedeutung</i>
-c	Stellt die Zähler einer Regel zurück. Dieser Parameter akzeptiert die Optionen PKTS und BYTES für die Spezifizierung der rückzustellenden Zähler.
-d	Setzt Ziel-Host-Namen, IP-Adresse, oder -Netzwerk eines Pakets, das einer Regel entspricht. Wenn es einem Netzwerk entspricht, können Sie zwei verschiedene Methoden für die Kennzeichnung der Netzmasken verwenden, wie zum Beispiel 192.168.0.0/255.255.255.0 oder 192.168.0.0/24.
-f	Wendet diese Regel nur für fragmentierte Pakete an. Durch Verwendung der !-Option nach diesem Parameter werden nur unfragmentierte Parameter abgefangen.
-i	Setzt die Eingangsnetzwerkschnittstelle, wie zum Beispiel eth0 oder ppp0, die für eine bestimmte Regel verwendet werden soll. Mit iptables können diese zusätzlichen Parameter nur mit INPUT- und FORWARD-Ketten in Verbindung mit der filter-Tabelle und der PREROUTING-Kette mit den nat- und mangle-Tabellen verwendet werden. Dieses Parameter besitzt verschiedene nützliche Optionen, die verwendet werden können, bevor der Name einer Schnittstelle festgelegt wird: ! - Weist dieses Parameter an, keine entsprechende Übereinstimmungen zu suchen, bzw. jede spezifizierte Schnittstelle wird von dieser Regel ausgeschlossen. + - Ein Platzhalterzeichen, das verwendet wird, um alle Schnittstellen zu kontrollieren, die einer bestimmten Zeichenkette entsprechen. Zum Beispiel das Parameter -i eth+ würde diese Regel für jegliche Ethernet-Schnittstellen auf Ihrem System anwenden, aber alle anderen Schnittstellen, wie ppp0, auslassen. Wenn der -i-Parameter verwendet wird, aber keine Schnittstelle bestimmt wurde, wird die Regel für alle Schnittstellen angewandt.

	<i>Bedeutung</i>
-j	weist iptables an, auf ein bestimmtes Ziel überzugehen, wenn ein Paket einer bestimmten Regel entspricht. Gültige Ziele, die nach der -j-Option verwendet werden können, sind unter anderem die Standardoptionen, ACCEPT, DROP, QUEUE und RETURN, sowie erweiterte Optionen, die über Module verfügbar sind, die standardmäßig mit dem Red Hat Linux iptables-RPM-Paket geladen werden, wie LOG, MARK und REJECT. Weitere Informationen über diese und andere Ziele, sowie Regeln zu deren Verwendung (da viele Ziele nur mit einer bestimmten Tabelle verwendet werden können), finden Sie auf der man-Seite iptables. Außer dem Spezifizieren von Zielvorgängen können Sie auch ein Paket, das dieser Regel entspricht an eine benutzerdefinierte Kette außerhalb der aktuellen Kette weiterleiten. Dies ermöglicht es, weitere Regeln in Übereinstimmung mit diesem Paket anzuwenden und letzteres nach spezifischeren Kriterien zu filtern. Wenn kein Ziel festgelegt ist, fließt das Paket an der Regel vorbei, ohne dass ein Vorgang hervorgerufen wird. Trotzdem springt der Zähler für diese Regel um 1 Stelle weiter, da das Paket der festgelegten Regel entspricht.
-o	Setzt die Ausgangsnetzwerkschnittstelle für eine bestimmte Regel und kann nur mit OUTPUT- und FORWARD-Ketten in der Tabelle filter und mit der POSTROUTING-Kette in den nat- und mangle-Tabellen verwendet werden. Die Optionen dieses Parameters sind dieselben, als jene des Parameters der Eingangsnetzwerkschnittstelle (-i).
-p	Setzt das IP-Protokoll für die Regel, die entweder icmp, tcp, udp oder all sein kann, um allen möglichen Protokollen zu entsprechen. Außerdem können weniger verwendete Protokolle, die in /etc/protocols aufgelistet sind, ebenfalls verwendet werden. Wenn diese Option beim Erstellen einer Regel ausgelassen wird, ist all die standardmäßige Option.
-s	-s Setzt den Ursprung eines bestimmten Pakets mithilfe derselben Syntax des (-d) Zielparameters.

```
18.8 <p><font color="#cc0000">iptables -N syn-flood</font> <br><font color="#cc0000">iptables -A INPUT -i $IFACE -p tcp --syn -j syn-flood</font> <br><font color="#cc0000">iptables -A syn-flood -m limit --limit 1/s --limit-burst 4 -j RETURN</font> <br><font color="#cc0000">iptables -A syn-flood -j DROP</font> </p>
```

## 18.9 Zieloptionen

Wenn ein Paket einer bestimmten Regel entspricht, kann die Regel das Paket an eine Reihe von verschiedenen Ziele senden, an denen dann eventuell weitere Vorgänge erfolgen, wie zum Beispiel das Protokollieren des Vorgangs. Außerdem hat jede Kette ein standardmäßiges Ziel, das verwendet wird, wenn ein Paket keiner Regel in der Kette entspricht oder wenn in der Regel, dem das Paket entspricht, ein Ziel angegeben ist. Es stehen nur einige Standardziele zur Verfügung, die entscheiden, was mit dem Paket geschehen soll:

<i>Ziel</i>	<i>Bedeutung</i>
benutzerdefinierte Kette	Der Name einer an einem früheren Zeitpunkt in dieser Tabelle erstellten und definierten Kette mit Regeln, die in Übereinstimmung mit diesem Paket überprüft werden, zusätzlich zu anderen Regeln in anderen Ketten, die in Übereinstimmung mit diesem Paket überprüft werden müssen. Diese Art von Ziel ist nützlich, um das Paket weiter zu analysieren, bevor man entscheidet, was damit geschehen soll oder bevor die Informationendes Pakets protokolliert werden.
ACCEPT	Das Paket kann erfolgreich bis an sein Ziel gelangen (oder eine andere Kette, wenn eine konfiguriert ist, um die erfolgreiche Kette zu verfolgen).
DROP	Das Paket wird fallen gelassen. Das System, welches das Paket gesendet hat, wird nicht über das Durchfallen des Pakets benachrichtigt. Das Paket wird einfach von der Regel entfernt, welche die Kette kontrolliert, und in den Papierkorb verschoben.
QUEUE	Das Paket wird in die Warteschlange für die Bearbeitung im Benutzerraum hinzugefügt, wo es (von einem Benutzer oder einer Anwendung, zum Beispiel) verwendet werden kann.
RETURN	Hält die Überprüfung der Übereinstimmung des Pakets mit Regeln in der aktuellen Kette an. Wenn das Paket mit einem RETURN-Ziel mit einer Regel in einer Kette übereinstimmt, die von einer anderen Kette aufgerufen wurde, wird das Paket an die erste Kette zurückgesendet, damit die Überprüfung wiederaufgenommen werden kann, wo sie unterbrochen wurde. Wenn die Regel RETURN in einer eingebauten Kette verwendet wird, und das Paket nicht an die vorherige Kette zurück kann, entscheidet das Standardziel der aktuellen Kette, was damit geschehen soll.
LOG	Protokolliert alle Pakete, die dieser Regel entsprechen. Da die Pakete vom Kernel protokolliert werden, bestimmt die Datei /etc/syslog.conf, wo diese Protokolldateien geschrieben werden. Standardmäßig werden sie in der Datei /var/log/messages gesp.
REJECT	Sendet ein Fehlerpaket zurück an das System, welches das Paket gesendet hat, und lässt dann das Paket fallen (DROP). Dieses Ziel ist nützlich, wenn Siedem System Bescheid sagen möchten, indem Sie das übereinstimmende Problem-paket senden. Mit dem REJECT-Ziel kann die --reject-with <Typ>-Option verwendet werden, um mehrere Details zusammen mit dem Fehlerpaket zu senden. Die Meldung port-unreachable ist der standardmäßige <Typ>-Fehler, der angezeigt wird, wenn keine andere Option angewandt wurde.Diese sind: icmp-net-unreachable, icmp-host-unreachable; icmp-port-unreachable,icmp-proto-unreachable, icmp-net-prohibitedor icmp-host-prohibited, Die Option option echo-reply ist ebenso erlaubt bei ICMP Paketen reply. Und tcp-reset bei TCP Paketen

Achten Sie darauf, welche Option (-A oder -I) Sie beim Hinzufügen von Regeln verwenden. Die Reihenfolge der Regeln kann sehr wichtig sein, wenn Sie bestimmen, ob ein bestimmtes Paket dieser oder jener Regel entspricht. Stellen Sie sicher, dass beim Hinzufügen einer Regel am Anfang oder am Ende einer Kette,diese nicht andere Regeln in der Kette beeinflusst.

## 18.10 Weitere Ziele

### 18.10.1 LOGGING

Zusätzlich zu diesen Standardzielen, können verschiedene weitere Ziele mit Erweiterungen verwendet werden, die Zielmodule heißen und die auf ähnliche Weise funktionieren, um Optionsmodulen zu entsprechen.

Weitere Informationen über Übereinstimmungsoptionsmodule finden Sie in Abschnitt namens Module mit Zusätzlichen Übereinstimmungsoptionen.

Es gibt viele erweiterte Zielmodule, von denen sich die meisten auf bestimmte Tabellen oder Situationen beziehen.

Nach dem LOG-Ziel können verschiedene Optionen verwendet werden, um die Protokollierart zu bestimmen:

<i>Option</i>	<i>Bedeutung</i>
--log-level	Bestimmt das Prioritätsniveau eines Protokollierereignisses. Auf der man-Seite syslog.conf finden Sie eine Liste der Prioritätsniveaus, deren Namen können als Optionen nach der Option --log-level verwendet werden.
--log-ip-options	Alle in den Kopfzeilen eines IP-Paket enthaltenen Optionen werden protokolliert.
--log-prefix	Fügt beim Schreiben einer Protokollzeile eine Zeichenkette vor der Protokollzeile ein. Es werden bis zu 29 Zeichen nach der --log-prefix-Option akzeptiert. Dies ist auch beim Schreiben von syslog-Filtern nützlich, die in Verbindung mit Paketprotokollierung verwendet werden.
--log-tcp-options	Alle in den Kopfzeilen eines TCP-Pakets enthaltenen Optionen werden protokolliert.
--log-tcp-sequence	Schreibt eine TCP-Sequenznummer für das Paket in der Protokolldatei.

## 18.11 Übereinstimmungsoptionen

Verschiedene Netzwerkprotokolle ermöglichen spezielle Übereinstimmungsoptionen, die auf spezifische Weise gesetzt werden können, um ein bestimmtes Paket mithilfe dieses Protokolls zu kontrollieren. Das Protokoll muss natürlicherweise zuerst im iptables-Befehl spezifiziert werden, wie zum Beispiel mit Verwendung von

-p <Protokollname>, um die Optionen für das Protokoll verfügbar zu machen.

	<i>///Bedeutung</i>
TCP	
-p tcp	Diese Übereinstimmungsoptionen sind für das TCP-Protokoll verfügbar:
--dport --destination-port	Setzt den Ziel-Port für das Paket. Für die Konfiguration dieser Option können Sie entweder einen Netzwerkdienstnamen (wie zum Beispiel www oder smtp), die Port-Nummer oder eine Reihe von Port-Nummern verwenden. Um die Namen und Alias-Namen der Netzwerkdienste und die Port-Nummern, die sie verwenden, nachzulesen, sehen Sie sich bitte die Datei /etc/services an. Sie können auch verwenden, um diese Übereinstimmungsoption zu spezifizieren. Um eine spezifische Reihe von Port-Nummern anzugeben, trennen Sie die zwei Nummern durch einen Doppelpunkt (:), wie in -p tcp --dport3000:3200. Die größtmögliche Reihe ist 0:65535. Sie können auch ein Ausrufezeichen (!) als Flag nach der Option --dport verwenden, damit iptables alle Pakete, die nicht diesen Netzwerkdienst oder Port verwenden, zu kontrollieren.
--sport --source-port	Setzt den Ursprungs-Port des Pakets unter Verwendung derselben Optionen wie --dport.
--syn	Kontrolliert alle TCP-Pakete, die eine Kommunikation beginnen, SYN-Pakete genannt, auf Übereinstimmung mit dieser Regel. Pakete, die einen Daten-Payload enthalten, werden nicht bearbeitet. Wird ein Ausrufezeichen (!) als Flag vor der Option --syn verwendet, werden alle Nicht-SYN-Pakete kontrolliert.
--tcp-option	Versucht mithilfe von TCP-spezifischen Optionen zu überprüfen, die innerhalb eines bestimmten Pakets aktiviert werden können. Diese Übereinstimmungsoption kann ebenfalls mit dem Ausrufezeichen (!) umgekehrt werden.
--tcp-flags	Ermöglicht die Verwendung von TCP-Paketen mit bestimmten Bits oder Flags, damit Sie der Regel entsprechen. Die Übereinstimmungsoption --tcp-flags akzeptiert nachstehend zwei Parameter, die Flags für verschiedene Bits in einer Liste mit Kommatrennung sind. Das erste Parameter ist die Maske, welche die zu untersuchenden Flags des Pakets bestimmt. Das zweite Parameter bezieht sich auf Flags, die im Paket gesetzt werden müssen, um eine Übereinstimmung zu erhalten. Mögliche Flags sind ACK, FIN, PSH, RST, SYN und URG. Zusätzlich können ALL und NONE ebenfalls verwendet werden, um Übereinstimmung eines jeden oder keines Flags zu erhalten. Zum Beispiel, eine iptables-Regel, die -p tcp --tcp-flags ACK,FIN,SYN SYN enthält überprüft nur TCP-Pakete, in denen das SYN-Flag aktiviert und die ACK- und FIN-Flags deaktiviert sind. Wie bei vielen anderen Optionen wird die Auswirkung der Überprüfungs-option durch Einfügen eines Ausrufezeichens (!) nach --tcp-flags umgekehrt, so dass für deren Überprüfung die Flags des zweiten Parameters nicht in Reihenfolge gesetzt werden müssen.
UDP	
-p udp	Diese Übereinstimmungsoptionen sind für das UDP-Protokoll verfügbar:
--dport --destination-port	Bestimmt den Ziel-Port des UDP-Pakets, unter Verwendung von Dienstnamen, Port-Nummer oder Reihe von Port-Nummern. Die Option kann anstatt --dport verwendet werden. In --dport match option in Abschnitt namens TCP-Protokoll werden verschiedene Methoden für die Anwendung dieser Option beschrieben.
--sport --source-port	Bestimmt den Ursprungs-Port des UDP-Pakets unter Verwendung von Dienstnamen, Port-Nummer oder Reihe von Port-Nummern. Die Überprüfungsoption kann anstelle von --sport verwendet werden. In --dport match option in Abschnitt namens TCP-Protokoll werden verschiedene Methoden für die Anwendung dieser Option beschrieben.
ICMP	
-p icmp	Pakete, welche das Internet Control Message Protocol (ICMP) verwenden, werden mithilfe der Option überprüft,

	<i>///Bedeutung</i>
--icmp-type	Bestimmt den Namen oder die Nummer des ICMP-Typs, mit der Regel übereinstimmen soll. Durch Eingabe des Befehls iptables -p icmp -h wird eine Liste aller gültigen ICMP-Namen angezeigt. Mögliche Werte sind: echo-reply, destination-unreachable, network-unreachable, host-unreachable, protocol-unreachable, port-unreachable, fragmentation-needed, source-route-failed, network-unknown, host-unknown, network-prohibited, host-prohibited, TOS-network-unreachable, TOS-host-unreachable, communication-prohibited, host-precedence-violation, precedence-cutoff, source-quench, redirect, network-redirect, host-redirect, TOS-network-redirect, TOS-host-redirect, echo-request (ping), router-advertisement, router-solicitation, time-exceeded (ttl-exceeded), ttl-zero-during-transit, ttl-zero-during-reassembly, parameter-problem, ip-header-bad required-option-missing, timestamp-request, timestamp-reply address-mask-request, address-mask-reply

## 18.12 Module mit Zusätzlichen Übereinstimmungsoptionen

**Zusätzliche Übereinstimmungsoptionen, die sich nicht spezifisch auf ein Protokoll beziehen, sind ebenfalls anhand von Modulen verfügbar, die geladen werden, wenn der Befehl iptables sie verwendet. Um ein Übereinstimmungsoptionmodul anzuwenden, müssen Sie das Modul mit dessen Namen laden, indem beim Erstellen einer Regel**

-m <Modulname> in den iptables-Befehl eingegeben wird.

Standardmäßig steht eine große Anzahl von Modulen zur Verfügung, von denen jedes eigene besondere Übereinstimmungsoptionen besitzt. Es ist sogar möglich, Ihre eigenen Module zu erstellen, um zusätzliche Übereinstimmungsoptionen zu erhalten, wie zum Beispiel für besondere Netzwerkanforderungen. Es gibt sehr viele Module, hier werden jedoch nur die bekanntesten beschrieben.

### 18.12.1 Kontrolle des Verbindungszustandes

Bei iptables wurde das sogenannte Connection tracking eingeführt. Das ermöglicht den Zustand von Verbindungen zu kontrollieren. Dabei werden die Daten ausgewertet die das Connection - Tracking Modul sammelt. Dieses Modul wird auch bei Network Address Translation benötigt mit der Option -m state wird das Modul bei Bedarf geladen. Der Name des Kernelmodules lautet ip\_conntrack

### 18.12.2 state

Das state-Modul, welches die --state-Übereinstimmungsoption definiert, kann ein Paket auf folgende Verbindungsstatus überprüfen:

-m state

<i>Option</i>	<i>Übereinstimmung</i>	
--state	ESTABLISHED	Ein übereinstimmendes Paket wird anderen Paketen in einer bestimmten Verbindung zugeordnet.
--state	INVALID	Ein übereinstimmendes Paket kann nicht mit einer bekannten Verbindung verknüpft werden.
--state	NEW	Ein übereinstimmendes Paket stellt entweder eine neue Verbindung her oder ist Teil einer vorher nie begegneten Zwei-Wegverbindung.
--state	RELATED	Pakete die in einer Beziehung zu offenen Verbindungen stehen, aber nicht direkt zur Verbindung gehören. Dazu gehören Protokolle die mehrere Kanäle benötigen wie FTP oder auch Fehler-nachrichten über ICMP. Ausser bei ICMP sind dazu jedoch weitere Module notwendig. Beispielsweise: ip_conntrack_ftp

Die Verbindungsstati können in Zusammenhang mit anderen verwendet werden, indem sie durch Kommas getrennt werden, wie zum Beispiel `-m state --state INVALID,NEW`.

### Beispiele

#Alle Pakete die ausgehend auf Port 80 konnektieren wollen bzw zu einer besteh. Verbindung #gehören.

```
iptables -A OUTPUT -p tcp --dport 80 -o eth0 -m state --state NEW,ESTABLISHED \
-j ACCEPT
```

#### FTP-Steuerkanal

# alle Pakete die ausgehend auf Port 21 konnektieren wollen bzw zu einer besteh. Verbindung #gehören.

```
iptables -A OUTPUT -p tcp --dport 21 -o eth0 -m state --state NEW,ESTABLISHED \
-j ACCEPT
```

# FTP-Datenkanal - Aktiv (ip\_conntrack\_ftp muss geladen sein)

```
iptables -A INPUT -p tcp -i eth0 --sport 20 -m state --state ESTABLISHED,RELATED \
-j ACCEPT
```

```
iptables -A OUTPUT -p tcp -o eth0 --dport 20 -m state --state ESTABLISHED -j ACCEPT
```

# FTP-Datenkanal - Passiv (ip\_conntrack\_ftp muss geladen sein)

```
iptables -A OUTPUT -p tcp -o eth0 --dport 1024: -m state \
--state ESTABLISHED,RELATED -j ACCEPT
```

# Alle Pakete die schon zu einer bestehenden Verbindung gehören oder in einer Beziehung dazu #stehen.

```
iptables -A INPUT -i eth0 -m state --state ESTABLISHED,RELATED -j ACCEPT
```

### 18.12.3 limit

Mit dem limit-Modul können Sie eine Grenze setzen für die Anzahl der in Übereinstimmung mit einer Regel zu überprüfenden Pakete. Dies ist besonders nützlich, wenn Regelübereinstimmungen protokolliert werden. Auf diese Weise vermeiden Sie, dass die zahlreichen übereinstimmenden Pakete Ihre Protokolldateien nicht mit wiederholten Nachrichten überfüllen, oder dass eine übermäßige Systemressourcennutzung zustande kommt. -m limit

<i>Option</i>	<i>Bedeutung</i>
--limit	Bestimmt die Anzahl der Übereinstimmungen für einen bestimmten Zeitraum, der mit einem Anzahl- und Zeitbearbeiter in dem Format <Nummer>/<time> angegeben wird. Zum Beispiel wird mit --limit 5/hour nur 5-mal in einer Stunde mit einer Regel nach Übereinstimmungen gesucht. Wenn kein Anzahl- und Zeitbearbeiter angegeben ist, wird der Standardwert 3/hour angenommen.
--limit-burst	Setzt eine Grenze für die Anzahl von Paketen, deren Übereinstimmung mit einer Regel gleichzeitig geprüft werden kann. Diese Option sollte in Verbindung mit der --limit-Option verwendet werden. Man kann außerdem einen maximalen Grenzwert setzen. Wenn keine Zahl angegeben wird, können anfangs nur 5 Pakete in Übereinstimmung mit der Regel überprüft werden.

### 18.12.4 multiport

Mit diesem Modul ist es bis zu 15 Ports auf einmal anzugeben. Die Angabe erfolgt durch ein durch Kommata getrennte liste. Die ist nur bei tcp oder udp möglich.

-m multiport

<i>Option</i>	<i>Bedeutung</i>
--source-port	Die Regel passt auf die Quellports
--destination-port	Die Regel passt auf die Zielports
--port	Die Regel passt auf die Ports

### 18.12.5 Mac

Um die Übereinstimmung einer bestimmten Hardware-MAC-Adresse eines Ethernet-Geräts spezifisch zu überprüfen, verwenden Sie das mac-Modul, das -m mac --mac-source plus eine MAC-Adresse als Option akzeptiert. Um eine MAC-Adresse auszu-schließen, fügen Sie nach der --mac-source-Übereinstimmungsoption ein Ausrufezeichen (!) hinzu. n.

## 18.12.6 Die Mangle

-Tabelle ist zu laden mit `-t mangle`

<i>Ziel</i>	<i>Bedeutung</i>
MARK	Dieses Ziel wird dazu verwendet Pakete mit vorzeichenlosen Interger-werten zu markieren. Später kann man dan mit der Sucherweiterug mask diese verwenden. Dieses Ziel kann nur in der mangle Tabelle verwendet werden. Die Option ist <code>--set-mark Wert</code>
TOS	Setzt den Typ auf das Type of Service Feld im IP Header. Dieses Ziel kann nur in der mangle Tabelle verwendet.Die Option ist <code>--set-mark Wert</code> 0x10 = Minimale Verzögerung 0x08 = Maximaler Durchsatz 0x04 = Maximale Zuverlässigkeit 0x02 = Minimale Kosten
TTL	Setzt den TTL Wert im IP Header auf den angegebenen Wert Dieses Ziel kann nur in der mangle Tabelle verwendet. Mögliche Optionen sind: <code>--ttl-set ttl</code> Setzt TTL auf den Wert <code>--ttl-dec ttl</code> Erniedrigt die TTL um den Wert <code>--ttl-inc ttl</code> Erhöht die TTL um den Wert

Beispiele:

Setzen der Marken auf Pakete

```
iptables -t mangle -A PREROUTING -j MARK -p icmp --set-mark 456
```

Auswerten der Marken

```
iptables -A FORWARD -m mark --mark 456
```

Setzen des minimalen Delays

```
iptables -t mangle -A PREROUTING -j TOS -p tcp --dport 22 --set-tos 0x10
```

```
iptables -t mangle -A PREROUTING -j TOS -p tcp --sport 22 --set-tos 0x10
```

Setzen der TTL

```
iptables -t mangle -A PREROUTING -j TTL -p tcp --dport 80 --ttl-set 5
```

## 18.13 NAT Tabelle

Die Nat -Tabelle ist zu laden mit **-t nat**

<i>Ziel</i>	<i>Bedeutung</i>
SNAT	Ändert die Quelladresse dieses und aller zukünftigen Pakete dieser Verbindung. Dieses Ziel kann nur in der POSTROUTING Kette in der nat Tabelle verwendet werden. --to-source Adresse[-Adresse][Port-Port] gibt die neue Quelladresse oder den Quellbereich an. Wenn -p tcp oder -p udp können auch Quellports angegeben werden.
DNAT	Ändert die Zieladresse dieses und aller zukünftigen Pakete dieser Verbindung. Dieses Ziel kann nur in der PREROUTING Kette und in der OUTPUT in der nat Tabelle verwendet werden. --to-destination Adresse[-Adresse][Port-Port] gibt die neue Zieladresse oder den Quellbereich an. Wenn -p tcp oder -p udp können auch Quellports angegeben werden.
MASQUERADE	Maskiert Pakete so, dass es aussieht, als stamme es von dem lokalen System. Entgegengesetzte Pakete werden automatisch demaskiert. Dieses Ziel kann auch nur in der nat Tabelle benutzt werden. Optional kann man mit --to-ports Port[-Port] einen Bereich angeben der maskiert werden soll.
REDIRECT	Leitet das Paket auf einen lokalen Port um. Die Option --to-ports <port>[-<port>] Hiermit kann man einen Port oder Portbereich angeben der zu benutzen ist. Ohne diese Option wird der Port nicht verändert Auch hier muss -p tcp oder -p udp dann mit angegeben werden.

### **1 <-> m nat (maskierung) (Source NAT)**

Wenn ein Paket eines Clients die Firewall erreicht wird in der Postrouting Kette die Source IP und der Source Port mit der ausgehenden IP der Firewall und einem freien Port ersetzt. Diese Zuordnung wird in einer Tabelle festgehalten. Kommt jetzt ein Antwortpaket des Servers zurück wird anhand der Tabelle das Paket entsprechend wieder umgeschrieben.

```
iptables -t nat -A POSTROUTING -j MASQUERADE -o ippp0
```

### **n <-> m (Source NAT)**

Wenn ein Paket eines Clients die Firewall erreicht wird in der Postrouting Kette die Source IP mit einer beliebigen IP ersetzt. Diese Zuordnung wird in einer Tabelle festgehalten. Kommt jetzt ein Antwortpaket des Servers zurück wird anhand der Tabelle das Paket entsprechend wieder umgeschrieben.

```
iptables -t nat -A POSTROUTING -j SNAT -o eth0 -s 195.145.95.1 \
```

```
--to-source 193.165.32.1
```

### **n <-> m (Destination NAT)**

Wenn ein Paket eines Clients die Firewall erreicht wird in der Prerouting Kette die Destination IP mit einer beliebigen IP ersetzt. Diese Zuordnung wird in einer Tabelle festgehalten. Kommt jetzt ein Antwortpaket des Servers zurück wird anhand der Tabelle das Paket entsprechend wieder umgeschrieben.

```
iptables -t nat -A PREROUTING -j DNAT -i eth0 -p tcp -d 217.7.27.3 --dport 22 \
--to-destination 192.168.254.3:22
```

Sonderform:

```
iptables -t nat -A PREROUTING -i eth0 -j REDIRECT --dport 80 --to-ports 3128
```

## **18.14 Auflistungsoptionen**

Der standardmäßige Auflistungsbefehl, `iptables -L`, liefert eine sehr allgemeine Übersicht der standardmäßigen aktuellen Regelketten der Filtertabelle. Es gibt zusätzliche Optionen, die mehr Informationen liefern und diese Informationen in einer besonderen Weise auflisten:

<i>Option</i>	<i>Bedeutung</i>
-v	Ausführliche Ausgabe anzeigen, wie zum Beispiel die Anzahl der Pakete und Bytes, die jede Kette gesehen hat, die Anzahl der Pakete und Bytes, die von jeder Regel auf Übereinstimmung geprüft wurden, und deren Schnittstellen einer bestimmten Regel entsprechen.
-x	Erweitert die Zahlen auf ihre exakten Werte. In einem arbeitenden System, können die Anzahl der Pakete und Bytes, die von einer bestimmten Kette oder Regel gesehen werden, unter Verwendung von K (Tausender), M (Millionen) und G (Milliarden) am Ende der Zahl gekürzt werden. Mit dieser Option werden die vollständigen Zahlen angezeigt.
-n	Zeigt IP-Adressen und Port-Nummern in numerischem Format, anstatt im standardmäßigen Host-Namen- und Netzwerkdienstformat, an.
--line-numbers	Listet Regeln in jeder Kette in Nähe von deren numerischer Reihenfolge in der Kette auf. Diese Option ist nützlich, wenn man versucht, eine bestimmte Regel aus einer Kette zu entfernen oder zu bestimmten, wo eine Regel in einer Kette eingefügt werden soll.

## **18.15 Zusätzliche Ressourcen**

Paketfilterung und iptables sind komplizierte Argumente. Zusätzliche Informationen können nützlich sein, um andere Gesichtspunkte und Methoden bzgl. der Kontrolle des Netzwerkverkehrs auf Ihrem System zu vergleichen. Installierte Dokumentation Die man-Seite iptables enthält eine vollständige Beschreibung verschiedener Befehle, Parameter und anderer Optionen, die Sie beim Hinzufügen von neuen Tabellen und beim Erstellen von Kettenregeln unterstützen.

## 18.16 Bedeutung einer Logfilezeile

```
Apr 22 08:59:32 jedzia kernel: fire-in IN=eth0 OUT=
MAC=00:01:02:42:26:59:00:00:c0:e5:b3:ad:08:00 SRC=172.20.4.12 DST=174.123.22.2
LEN=60 TOS=0x00 PREC=0x00 TTL=63 ID=35239 DF PROTO=TCP SPT=1060 DPT=22
WINDOW=5840 RES=0x00 CWR ECE SYN URGP=0
```

<i>Eintrag</i>	<i>Bedeutung</i>
Apr 22 08:59:32	Der Zeitstempel
jedzia	Der Rechnername
kernel:	Der Prozess der die Meldung verursacht hat
fire-in	Die --log-prefix die angegeben wurde
IN=eth0 OUT=	Input und Outputinterface
MAC=00:01:02:42:26:59: 00:00:c0:e5:b3:ad: TYPE=08:00	Die Macadressen und das Typenfeld
SRC=172.20.4.12	Die Quelladresse
DST=174.123.22.2	Die Zieladresse
LEN=60	Die Länge des Paketes in Byte
TOS=0x00	Das Type of Service Feld Typenfeld
PREC=0x00	Das Type of Service Feld Precedencefeld
TTL=63	Time to Live Feld
ID=35239	Kennung des IP Headers
PROTO=TCP	Das Protokoll das IP Transportiert
SPT=1060 DPT=22	Source und Destination Port
WINDOW=5840	Grösse des TCP Fensters
RES=0x00 CWR ECE	Mit Explicit Congestion Notification (ECN) wurden 2 neue flags dem TCP header hinzugefügt: Congestion Window Reduced (CWR) und ECN-Echo (ECNE).
SYN URGP=0	Flags die gesetzt sind z.B SYN ACK FIN

## 18.17 Kernelparameter

Unterhalb von /proc/ befinden sich Parameter die zur Laufzeit verändert werden können  
 Infos zu den Parametern unter /usr/src/linux/Documentation/networking/ip-sysctl.txt.

<i>Eintrag</i>	<i>Bedeutung Wert =0 ist aus Wert = 1 ist an</i>
Im Verzeichnis /proc/sys/net/ipv4/ befinden sich die folgenden Parameter	
ip_forward	Aktivieren des IP Forwarding
tcp_syncookies	Syncookies aktivieren hiermit wird das sogenannte Synflooding verhindert. Empfehlung 1
icmp_echo_ignore_all	Um auch keinen ping zu zulassen kann man folgendes aktivieren für paranoide Menschen zu empfehlen. Empfehlung 1
icmp_echo_ignore_broadcasts	Um Smurf Attacken vorzubeugen sollte man dies einschalten Empfehlung 1
icmp_ratelimit	Begrenzt die maximale Rate von ICMP Paketen welche durch icmp_ratemask beschrieben werden 0 deaktiviert diese Option sonst ist die Einheit jiffies (Kernelzeiteinheit) Empfehlung 5-10
icmp_ratemask	Maske fuer ICMP Pakete die limitert sind. dest unreachable (3), source quench (4), time exceeded (11) and parameter problem (12) daraus errechnet $2^3 + 2^4 + 2^{11} + 2^{12} = 6168$
ip_contrack_max	Maximale Anzahl der Connections die ,die State Tabelle verwalten kann. Diese ist Abhängig vom Speicher 128 MByte ~ 8184
ip_dynaddr	Wenn dieser Wert ungleich 0 ist , wird der Dynamische Adress Unterstützung aktiviert . Wenn der Wert gösser als 1 ist wird beim überschreiben mit einer dynamischen Adresse eine Kernellognachricht generiert.
Die folgenden Parameter stehen im Unterverzeichnis /proc/sys/net/ipv4/conf und können für alle Interfaces oder auch seperat konfiguriert werden. Die Einstellungen werden in den jeweiligen Unterverzeichnissen vorgenommen. (all default eth0 lo)	
rp_filter	Hiermit soll Spoofing verhindert werden. Wenn eine Ip nummer von aussen kommt, die eigentlich aus einem inneren Netz kommen müsste wird sie abgelehnt. Empfehlung 1
accept_redirects	Akzeptiere ICMP redirect messages. Empfehlung 0
accept_source_route	Akzeptiere Sourcerouting Empfehlung 0
bootp_relay	Soll Rechner als bootp_relay dienen ? (Dhcprelay) Empfehlung 0
mc_forwarding	Aktivierung von Multicastrouting. Wenn CONFIG_MROUTE gesetzt sein um diese Option aktivieren zu können. Empfehlung 0
log_martians - BOOLEAN	Logt Pakete mit mit unmöglicher Adresse. Empfehlung 1
Unterhalb von /proc/net befinden sich Informationen die nicht verändert werden können	
ip_tables_names	Namen der aktuellen Tabellen (mangle ,nat und filter)
ip_contrack	Liste der verwalteten Verbindungen in der state Tabelle

## 19 ICMP TYPES AND CODES

Q = Query E = Error

<i>TYPE</i>	<i>CODE</i>	<i>Description</i>	<i>Q</i>	<i>E</i>
0	0	Echo Reply	x	
3	0	Network Unreachable		x
3	1	Host Unreachable		
3	2	Protocol Unreachable		
3	3	Port Unreachable		
3	4	Fragmentation needed but no frag. bit set		
3	5	Source routing failed		
3	6	Destination network unknown		
3	7	Destination host unknown		
3	8	Source host isolated (obsolete)		
3	9	Destination network administratively prohibited		
3	10	Destination host administratively prohibited		
3	11	Network unreachable for TOS		
3	12	Host unreachable for TOS		
3	13	Communication administratively prohibited by filtering		
3	14	Host precedence violation		
3	15	Precedence cutoff in effect		
4	0	Source quench		
5	0	Redirect for network		
5	1	Redirect for host		
5	2	Redirect for TOS and network		
5	3	Redirect for TOS and host		
8	0	Echo request	x	
9	0	Router advertisement		
10	0	Route solicitation		
11	0	TTL equals 0 during transit		x
11	1	TTL equals 0 during reassembly		
12	0	IP header bad (catchall error)		
12	1	Required options missing		
13	0	Timestamp request (obsolete)		
14	0	Timestamp reply (obsolete)		
15	0	Information request (obsolete)		
16	0	Information reply (obsolete)		
17	0	Address mask request		
18	0	Address mask reply		

## 20 IPTABLES MANUAL

### *iptables Befehl [Optionen]*

Befehl zur Systemverwaltung. Konfiguriert die Netfilter-Filterregeln. Im 2.4-Kernel ist die ipchains-Firewall-Funktionalität durch das Kernel-Modul *netfilter* ersetzt worden, *netfilter* kann so konfiguriert werden, daß es genau wie ipchains funktioniert, wird aber auch mit dem Modul *iptables* geliefert, das ähnlich wie ipchains funktioniert, aber erweiterbar ist. *iptables-Regeln* bestehen aus Suchkriterien und einem Ziel, einem Resultat, welches angewendet wird, wenn das Paket auf die Kriterien paßt. Die Regeln sind in Ketten organisiert. Sie können diese Regeln dazu verwenden, eine Firewall aufzubauen, Ihr lokales Netzwerk zu maskieren oder einfach nur bestimmte Arten von Netzwerkverbindungen zurückzuweisen.

Es gibt drei eingebaute Tabellen für *iptables*, eine für das Filtern von Paketen (*filter*), eine für Network Address Translation (*nat*) und die letzte für spezielle Paket-Manipulationen (*mangle*). Firewall-Regeln sind in Ketten organisiert, geordneten Listen von Regeln, die der Kernel auf der Suche nach Treffern durchläuft. Die Tabelle *filter* hat drei eingebaute Ketten: *INPUT*, *OUTPUT* und *FORWARD*. Die Ketten *INPUT* und *OUTPUT* bearbeiten Pakete, die vom Host-System ausgehen oder für dieses bestimmt sind. Die Kette *FORWARD* bearbeitet Pakete, die das System nur durchlaufen. Die Tabelle *nat* hat ebenfalls drei eingebaute Ketten: *PREROUTING*, *POSTROUTING* und *OUTPUT*, *mangle* hatte früher (vor Kernel 2.4.18) nur zwei Ketten: *PREROUTING* und *OUTPUT*. Mittlerweile sind *PREROUTING*, *INPUT* und *FORWARD* dazugekommen

*netfilter* überprüft Pakete, die im System eintreffen. Nachdem alle eventuellen *PREROUTING*-Regeln angewendet worden sind, werden die Pakete an die *INPUT*-Kette beziehungsweise die *FORWARD*-Kette, wenn die Pakete das System nur passieren, weitergereicht. Beim Verlassen des Systems werden die Pakete durch die *OUTPUT*-Kette und dann durch alle eventuellen *POSTROUTING*-Regeln durchgereicht.

Jede dieser Ketten hat ein Default-Ziel, das verwendet wird, wenn keine Regel paßt. Es können auch benutzerdefinierte Ketten erzeugt und als Ziele für Pakete verwendet werden, diese haben aber keine Default-Ziele. Wenn in einer benutzerdefinierten Kette kein Treffer gefunden wird, dann wird das Paket an die Kette zurückgegeben, von der die aktuelle Kette aufgerufen wurde, und mit der nächsten Regel in dieser Kette verglichen.

*iptables* ändert nur die Regeln im laufenden Kernel. Wenn das System abgeschaltet oder neu gestartet wird, gehen alle Änderungen verloren. Sie können den Befehl *iptables-save* verwenden, um ein Skript zu erzeugen, mit dem die Firewall-Einstellungen später mittels *iptables-restore* wiederhergestellt werden. Solche Skripten werden häufig beim Systemstart verwendet. Viele Distributionen haben ein Initialisierungsskript für *iptables*, das die Ausgabe von *iptables-save* verwendet.

**Befehle** iptables wird immer mit einem der folgenden Befehle aufgerufen:

**-A Kette Regeln, - -append Kette**

*Regeln* Die *Regeln* an die *Kette* anhängen.

**-I Kette Zahl Regeln, - -insert Kette Zahl Regeln**

Die *Regeln* an der durch *Zahl* angegebenen Positionen in die *Kette* einfügen.

**-D Kette Regeln, - -delete Kette Regeln**

*Regeln* aus der *Kette* löschen. Die *Regeln* können durch ihre Positionsnummer in der *Kette* und durch eine allgemeine Regelbeschreibung angegeben werden.

**-R Kette Zahl Regel, - -replace Kette Zahl Regel**

Eine *Regel* in der *Kette* ersetzen. Die zu ersetzende *Regel* wird durch ihre Position mit *Zahl* angegeben.

**-C Kette Regel, - -check Kette Regeln**

Ein Netzwerk-Paket erzeugen, das auf die angegebene *Regel* paßt, und überprüfen, was die *Kette* damit macht. Die *Regel* muß die *Quelle*, das *Ziel*, das *Protokoll* und die *Schnittstelle* des zu konstruierenden Pakets angeben.

**-L [Kette], - -list Kette**

Alle *Regeln* in *Kette* ausgeben. Wenn keine *Kette* angegeben wird, alle *Regeln* in allen *Ketten* ausgeben.

**-F [Kette], - -flush Kette**

Alle *Regeln* aus *Kette* oder aus allen *Ketten*, wenn *Kette* nicht angegeben wird, löschen.

**-Z [Kette], - -zero Kette**

Die *Zähler* für *Pakete* und *Bytes* in *Kette* zurücksetzen. Wenn keine *Kette* angegeben wird, werden die *Zähler* in allen *Ketten* zurückgesetzt. Wenn keine *Kette* angegeben wird, dafür aber auch der Befehl *-L*, dann werden die aktuellen *Zählerwerte* vor dem Zurücksetzen ausgegeben.

**-N Kette, - -new-chain Kette**

Eine neue *Kette* erzeugen. Der Name der *Kette* muß eindeutig sein. Auf diese Weise werden benutzerdefinierte *Ketten* erzeugt.

**-X [Kette], - -delete-chain Kette**

Die angegebene benutzerdefinierte *Kette* oder alle benutzerdefinierten *Ketten*, wenn keine *Kette* angegeben wird, löschen.

**-P Kette Ziel, - -policy Kette Ziel**

Das Default-Ziel für eine eingebaute *Kette* angeben; das *Ziel* darf nicht selbst wieder eine *Kette* sein.

**-E AlteKette NeueKette, - -rename-chain**

*AlteKette NeueKette AlteKette* in *NeueKette* umbenennen.

**-h [icmp]**

Hilfetext. Mit der Option *icmp* wird, werden die eine Liste *ICMP-Typen* ausgegeben.

## ZIELE

Ein Ziel kann der Name einer Kette oder einer der folgenden Werte sein:

### ACCEPT

Das Paket durchlassen.

### DROP

Das Paket verwerfen.

### QUEUE

Das Paket zur Verarbeitung an den User Space schicken.

### RETURN

In die Kette zurückkehren, aus der diese Kette aufgerufen wurde, und die nächste Regel überprüfen. Wenn RETURN das Ziel einer eingebauten Kette ist, wird das Default-Ziel der eingebauten Kette verwendet.

### *Parameter für Regelspezifikationen*

Diese Optionen werden dazu benutzt, Regeln zur Verwendung mit den vorstehenden Befehlen zu erzeugen. Regeln bestehen aus einem Suchkriterium und üblicherweise einem Ziel, zu dem gesprungen werden soll (-j), wenn das Suchkriterium erfüllt wurde. Viele der Parameter für diese Suchkriterien können durch Voranstellen eines Ausrufungszeichens (!) auch in ihrer logischen Bedeutung umgedreht werden. Diese Regeln passen dann auf alles außer den angegebenen Parameter.

#### **-p [!] *Name*, - -protocol [!] *Name***

Die Regel paßt auf Pakete des Protokolls *Name*. Der Wert von *Name* kann als Name oder als in der Datei */etc/protocols* vorhandene Zahl angegeben werden. Die gängigsten Werte sind tcp, udp, icmp und der spezielle Wert all. Die Zahl 0 entspricht dem Wert all; dies ist auch der Defaultwert, wenn diese Option nicht verwendet wird. Wenn es für das angegebene Protokoll erweiterte Regeln gibt, werden diese automatisch geladen. Sie müssen sie nicht explizit mit der Option -m laden.

#### **-s [!] *Adresse[Maske] [!] [Port]*, - -source [!] *Adresse[Maske] [!] [Port]***

Gibt die Quell-Adresse an, auf die diese Regel paßt. Die Adresse kann in Form eines Hostnamens, eines Netzwerknams oder einer IP-Adresse angegeben werden. Die optionale Maske ist die zu verwendende Netzmaske, die entweder in traditioneller Form (also z.B. (255.255.255.0) oder in der modernen Form (also z.B. 24) angegeben werden.)

#### **-d [!] *AdresseVMaske] [!] [Port]*, - -destination [!] *Adresse[Maske} [Port]***

Die Regel paßt auf Pakete mit der *Ziel-Adresse*. Die Syntax dieses Befehlsparameters ist die gleiche wie bei der Option -s.

#### **-j *Ziel*, - -jump *Ziel***

Zu einem bestimmten Ziel oder einer benutzerdefinierten Kette springen. Wenn diese Option für eine Regel nicht angegeben wird, dann führen Treffer bei dieser Regel nur zum Erhöhen der Zähler der Regel, und das Paket wird mit der nächsten Regel verglichen.

#### **-i [!] *Name[+]*, - -in-interface *Name[+]***

Die Regel paßt auf Pakete von der Schnittstelle *Name[+]*. *Name* ist die von Ihrem System verwendete Netzwerk-Schnittstelle (also beispielsweise eth0 oder ppp0). Ein + kann als Jokerzeichen verwendet werden, ppp+ würde also beispielsweise auf jede Schnittstelle passen, deren Name mit ppp beginnt.

**-o [!] *Name[+]*, - -out-interface *Name[+]***

Die Regel paßt auf Pakete, die von der Netzwerk-Schnittstelle *Name* geschickt werden. Die Syntax von *Name* wird bei der Beschreibung von -i erklärt.

**[!]-f, [!]--fragment**

Die Regel paßt auf alles außer dem ersten Fragment eines fragmentierten Pakets.

## ***Optionen***

**-v, - -verbose**

Verbose-Modus

**-n, - -numeric**

Die IP-Adresse und Port-Nummern in numerischer Form ausgeben. Defaultmäßig werden, wo möglich, Namen angezeigt.

**-x, - -exact**

Alle Zahlen in einer Liste (-L) expandieren. Den exakten Wert der Paket- und Byte-Zähler anstelle von gerundeten Werten anzeigen.

**-m *Modul*, --match**

Die zu *Modul* gehörenden Regelerweiterungen explizit laden. Siehe dazu den folgenden Abschnitt »Suchen in Erweiterungen«.

**-h [*icmp*], - -help [*icmp*]**

Einen Hilfetext ausgeben. Wenn *icmp* angegeben wird, wird eine Liste gültiger ICMP-Typnamen ausgegeben, -h kann auch zusammen mit der Option -m verwendet werden, um Hilfe zu einem Erweiterungsmodul zu bekommen.

**--line-numbers**

Wird zusammen mit dem Befehl -L verwendet. Fügt jeder Regel im Listing die Zeilennummer hinzu, die die Position in der Kette angibt.

## Erweiterungen suchen

Netfilter wird mit mehreren Kernel-Modulen geliefert, die die Suchmöglichkeiten der Regeln erweitern. Diejenigen Erweiterungen, die zu bestimmten Protokollen gehören, werden automatisch geladen, wenn die Option `-p` verwendet wird, um das Protokoll anzugeben. Alle anderen müssen explizit mit der Option `-m` geladen werden.

**tcp** Wird geladen, wenn `-p tcp` das einzige angegebene Protokoll ist.

**--source-port [!] [port][:Port] - --sport [!] [Port][:Port]**

Die Regel paßt auf die angegebenen Quellports. Mit dem Doppelpunkt kann ein Bereich von Ports angegeben werden. Wenn der erste Port weggelassen wird, wird 0 angenommen, wenn der zweite Port weggelassen wird, ist der Default 65535. Anstelle des Doppelpunkts kann auch ein Bindestrich verwendet werden.

**--destination-port [!] [Port][:Port], - --dport [!] [Port][:Port]**

Die Regel paßt auf die angegebenen Zielports. Die Syntax ist die gleiche wie bei `--source-port`.

**--tcp-flags [!] *Maske comp***

Die Regel paßt auf Pakete, in denen die in *Maske* und *comp* genannten Flags gesetzt sind. *Maske* ist eine durch Kommata getrennte Liste von Flags, die geprüft werden sollen, *comp* ist eine Kommata-getrennte Liste von Flags, die gesetzt werden müssen, damit die Regel paßt. Mögliche Flags sind **SYN, ACK, FIN, RST, URG, PSH, ALL** und **NONE**.

**[!] - --syn**

Die Regel paßt auf Pakete, bei denen das SYN-Bit gesetzt und die ACK- und FIN-BITS gelöscht sind. Dies sind Pakete, mit denen TCP-Verbindungen angefordert werden; das Blockieren dieser Pakete verhindert eingehende Verbindungen. Diese Option ist eine Abkürzung für `--tcp-flags SYN, RST.ACK SYN`.

**udp** Wird geladen, wenn `-p udp` das einzige angegebene Protokoll ist.

**--source-port [!] [Port][:Port], - --Sport [!] [Port][:Port]**

Die Regel paßt auf die angegebenen Quellports. Die Syntax ist die gleiche wie bei der Option `--source-port` der TCP-Erweiterung.

**--destination-port [!] [Port][:Port], - --dport [!] [Port][:Port]**

Die Regel paßt auf die angegebenen Zielports. Die Syntax ist die gleiche wie bei der Option `--source-port` der TCP-Erweiterung.

**icmp**

Wird geladen, wenn `-p icmp` das einzige angegebene Protokoll ist.

**--icmp-type [!] *Typ***

Die Regel paßt auf den angegebenen *ICMP-Typ*. *Typ* kann ein numerischer ICMP-Typ oder einer der von iptables `-p icmp -h` angezeigten ICMP-Typnamen sein.

**mac** Wird explizit mit der Option -m geladen.

**- -mac-source [!] *Adresse***

Die Regel paßt auf die *Quell-Adresse*, die das Paket übertragen hat. *Adresse* muß in durch Doppelpunkte getrennter Hexbyte-Notation angegeben werden. (Beispiel: --mac-source 00:60:08:91:CC:B7)

**limit**

Wird explizit mit der Option -m geladen. Die limit-Erweiterung wird dazu verwendet, die Anzahl der passenden Pakete zu begrenzen. Das ist besonders in Kombination mit dem Ziel LOG nützlich. Regeln, die diese Erweiterung verwenden, passen solange, bis die angegebene Grenze erreicht ist.

**- -limit *Rate***

Die Regel paßt auf Adressen mit der angegebenen *Rate*. *Rate* wird als Zahl mit optional angehängtem /second, /minute, hour oder /day angegeben. Wenn diese Option nicht verwendet wird, ist der Default ' 3/hour'.

**- -limit-burst [*Zahl*]**

Legt die maximale Anzahl von Paketen fest, die in einem Burst passen. Wenn diese Zahl erreicht ist, passen keine weiteren Pakete mehr auf diese Regel, bis die Zahl wieder aufgeladen worden ist. Das geschieht mit der durch die Option - -limit angegebenen Rate. Wenn diese Option nicht angegeben wird, ist der Default 5.

**multiport**

Wird explizit mit der Option -m geladen. Die multiport-Erweiterungen passen auf Mengen von Quell- oder Ziel-Ports. Diese Regeln können nur zusammen mit -p tcp und -p udp verwendet werden. Bis zu 15 Ports können in einer durch Kommata getrennten Liste angegeben werden.

**- -source-port [*Ports*]**

Die Regel paßt auf die angegebenen Quellports.

**- -destination-port [*Ports*]**

Die Regel paßt auf die angegebenen Zielports.

**- -port [*Ports*]**

Die Regel paßt, wenn das Paket den gleichen Quell- und Zielport hat und der Port in *Ports* enthalten ist.

**mark**

Wird explizit mit der Option - m geladen. Dieses Modul arbeitet mit dem Erweiterungsziel MARK zusammen:

**- -mark *Wert[Maske]***

Paßt auf den angegebenen, vorzeichenlosen Markierungswert-Wert. Wenn eine Maske angegeben wird, wird diese vor dem Vergleich logisch-UND mit der Markierung verknüpft.

## **owner**

Wird explizit mit der Option -m geladen. Die owner-Erweiterungen prüfen die Benutzer-, Gruppen-, Prozeß- und Sitzungs-IDs eines lokalen Paket-Erzeugers. Das ist nur in einer OUTPUT-Kette sinnvoll.

### **- -uid-owner *Benutzerin***

Die Regel paßt auf Pakete, die von einem Prozeß erzeugt wurden, der *Benutzerin* gehört.

### **- -gid-owner *GruppenID***

Die Regel paßt auf Pakete, die von einem Prozeß erzeugt wurden, der *GruppenID* gehört.

### **- -pid-owner *ProzeßID***

Die Regel paßt auf Pakete, die vom Prozeß mit der ID *Prozeß ID* erzeugt wurden.

### **- -sid-owner *SitzungsID***

Die Regel paßt auf Pakete, die von einem Paket in der Sitzung *SitzungsID* erzeugt wurden.

## **state**

Wird explizit mit der Option -m geladen. Dieses Modul prüft den Verbindungszustand eines Pakets.

### **- -state *Zustände***

Die Regel paßt, wenn das Paket einen der in der durch Kommata getrennten Liste angegebenen Zustände hat. Zulässige Zustände sind **INVALID, ESTABLISHED, NEW** und **RELATED**.

## **tos**

Wird explizit mit der Option -m geladen. Dieses Modul prüft das Type of Service-Field im Header eines Pakets.

### **- -tos *Wert***

Die Regel paßt, wenn das Paket den TOS *Wert* hat. *Wert* kann ein numerischer Wert oder ein Type of Service-Name sein. iptables -m tos -h gibt eine Liste zulässiger TOS-Werte aus.

## ***Ziel-Erweiterungen***

Erweiterungs-Ziele sind optionale zusätzliche Ziele, die von separaten Kernel-Modulen unterstützt werden. Sie haben eigene Option.

**LOG** Informationen über das Paket im Systemprotokoll speichern.

### **- -log-level *Ebene***

Stellt die Ebene als Name oder Nummer (wie in *syslog.conf* definiert) ein.

- **-log-prefix *Präfix***

Jedem Protokolleintrag den String *Präfix(max 30 Zeichen)* voranstellen.

- **-log-tcp-sequence**

Die TCP-Sequenz-Nummern protokollieren. Dies ist ein Sicherheitsrisiko, wenn Anwender das Systemprotokoll lesen können.

- **-log-tcp-options**

Optionen aus dem TCP-Paket-Header protokollieren.

- **-log-ip-options**

Optionen aus dem IP-Paket-Header protokollieren.

**MARK**

Wird dazu verwendet, Pakete mit einem vorzeichenlosen Integer-Wert zu markieren, den Sie später in der Sucherweiterung mark verwenden können. Kann nur in der mangle-Tabelle verwendet werden.

- **-set-mark *Wert***

Markiert das Paket mit *Wert*.

**REJECT**

Verwirft das Paket und sendet gegebenenfalls eine ICMP-Nachricht an den Absender, in der das Verwerfen des Pakets gemeldet wird. Wenn das Paket eine ICMP Fehlermeldung, ein unbekannter ICMP-Typ oder ein Nicht-Head-Fragment war oder wenn bereits zu viele ICMP-Nachrichten an diese Adresse geschickt worden sind, wird keine Nachricht geschickt.

- **-reject-with *Typ***

Schickt den angegebenen ICMP-Nachrichtentyp. Zulässige Werte sind icmp-net-unreachable, icmp-host-unreachable, icmp-port-unreachable und icmpprotounreachable. Wenn das Paket ein ICMP-Ping-Paket war, kann *Typ* auch echo-reply sein.

**TOS**

Setzt das Type of Service-Feld im IP-Header. TOS ist nur bei Regeln in der Tabelle mangle ein zulässiges Ziel.

- **-set-tos *Wert***

Setzt das TOS-Feld auf *Wert*. Der Wert kann als 8-Bit breiter Wert oder als TOS-Name angegeben werden. Eine Liste der zulässigen Namen bekommen Sie mit **iptables -j TOS -h**.

## SNAT

Ändert die Quelladresse dieses und aller zukünftigen Pakete in der aktuellen Verbindung. SNAT kann nur als Teil der Kette POST-ROUTING in der Tabelle *nat* verwendet werden.

### - **-to-source** *Adresse*[-*Adresse*][*Port*-*Port*]

Gibt die neue Quelladresse oder einen Bereich von Adressen an. Wenn eines der Protokolle *tcp* oder *udp* mit der Option *-p* angegeben worden sind, können auch Quellports angegeben werden. Wenn keiner angegeben wird, wird die neue Quelle nach Möglichkeit auf den gleichen Port abgebildet. Wenn das nicht möglich ist, werden Ports unter 512 auf andere Ports unter 512, Ports zwischen 512 und 1024 auf andere Ports unter 1024 und Ports über 1024 auf andere Ports über 1024 abgebildet.

## DNAT

Ändert die Zieladresse des aktuellen und aller weiteren Pakete in der aktuellen Verbindung. DNAT kann nur als Bestandteil der POSTROUTING-Tabelle in der Tabelle *nat* verwendet werden.

### - **-to-destination** *Adresse*[-*Adresse*][*Port*-*Port*]

Gibt die neue Zieladresse oder einen Bereich von Adressen an. Die Argumente für diese Option sind die gleichen wie bei der Option *-to-source* des Erweiterungs-Ziels SNAT.

## MASQUERADE

Maskiert das Paket so, daß es aussieht, als stamme es vom aktuellen System. Entgegengesetzte Pakete aus maskierten Verbindungen werden automatisch demaskiert. Dieses Ziel kann nur in Ketten in der Tabelle *nat* verwendet werden, die eingehende Pakete bearbeiten, und sollte nur bei dynamischen IP-Adressen (wie bei Dial-Up-Verbindungen) benutzt werden. Für statische Adressen sollte DNAT verwendet werden.

### - **-to-ports** *Port*[-*Port*]

Gibt den Port oder einen Bereich von Ports an, der zum Maskieren verwendet werden soll. Diese Option ist nur zulässig, wenn eines der Protokolle *tcp* und *udp* mit der Option *-p* angegeben worden sind. Wenn diese Option nicht benutzt wird, wird der Port des maskierten Pakets nicht verändert.

## REDIRECT [- **-to-port** *Port*]

Leitet das Paket auf den lokalen *Port* um. Dies ist nützlich für transparente Proxies.

### - **-to-ports** *Port*[-*Port*]

Gibt den Port oder den Bereich von Ports im lokalen System an, auf den das Paket umgeleitet werden soll. Diese Option ist nur zulässig, wenn eines der Protokolle *tcp* oder *udp* mit der Option *-p* angegeben worden ist. Wenn diese Option nicht angegeben wird, wird der Port des umgeleiteten Pakets nicht geändert.

## 21 Hilfreiche Websites

<http://netfilter.samba.org>

Enthält verschiedene Informationen über iptables, sowie FAQ (häufig gestellte Fragen) zu spezifischen Problemen, denen Sie begegnen, und verschiedene hilfreiche Handbücher von Rusty Russell, dem Linux-IP-Firewall-Warter. In diesen Anleitungen werden Themen, wie Netzwerkgrundlagen, 2.4-Kernel-Paketfilterung und NAT-Konfigurationen und Netfilter besprochen.

[http://www.linuxnewbie.org/nhf/intel/security/iptables\\_basics.html](http://www.linuxnewbie.org/nhf/intel/security/iptables_basics.html) -

Ein allgemeiner Überblick darüber, wie sich Pakete durch den Linux-Kernel bewegen, plus eine Einleitung zur Erstellung von einfachen iptables-Befehlen.

<http://securityportal.com/cover/coverstory20010122.html> -

Eine Einführung zu den Paketbearbeitungseigenschaften des 2.4-Kernels, einschließlich Firewall, Schutz vor Denial of Service-Angriffe und heimliche Überprüfungen, MAC-Adressenfilterung und fortgeschrittene Paketprotokollierung.

Quelle : RedHat Benutzerhandbuch , Linux in a Nutshell , iptables Manual ,  
Firewallanleitung von [www.pro-linux.de](http://www.pro-linux.de)  
[www.xinux.de](http://www.xinux.de)

## 22 Exploits

Beispiel eines Exploits

### Dämonisches: Exploits

- Realisierung der Internetservices über Programme im Hintergrund
- Schlampige Programmierung liefert Angriffspunkte:  
Code wird auf den Stack plziert und ausgeführt

C-Code:

```
input ( int n )
{
    char zeichen[100] ;
    gets ( zeichen ) ;
    ...
}
```

