

MultiNET Services GmbH

iptables

Fachhochschule München, 13.6.2009

Dr. Michael Schwartzkopff, MultiNET Services GmbH

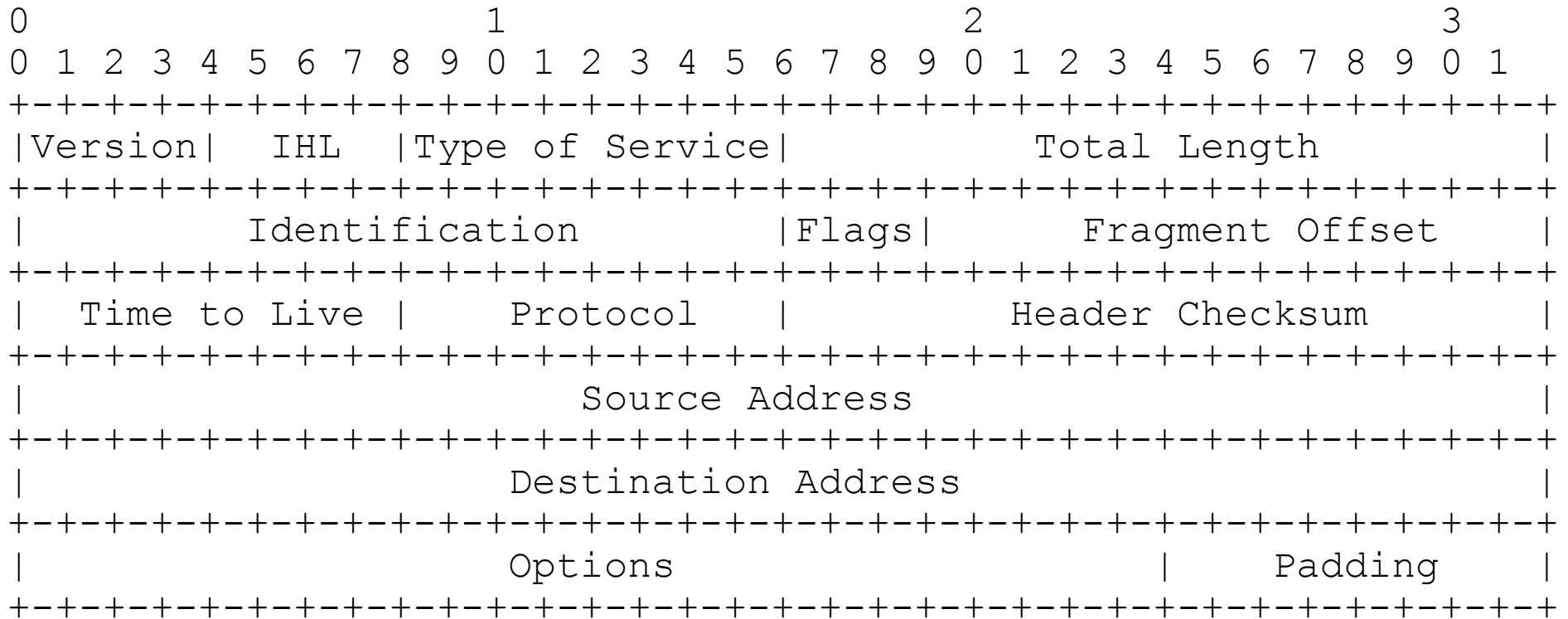
Entwicklung von Paketfiltern

- Seit es Internet gibt, sind Paketfilter notwendig.
- Grundproblem: Vertrauen!
- Erste Paketfilter in Routern, z.B. Cisco:

```
# access list 101 permit tcp 192.168.0.0 0.0.0.255 any eq 80
```
- Probleme:
 - Stateful Inspection
 - Protokolle mit verschiedenen Portinformationen: z.B. FTP!
- Erste Produkte:
 - IPFilter (*BSD, Solaris): 1993
 - Check Point mit Stateful Inspection: 1994

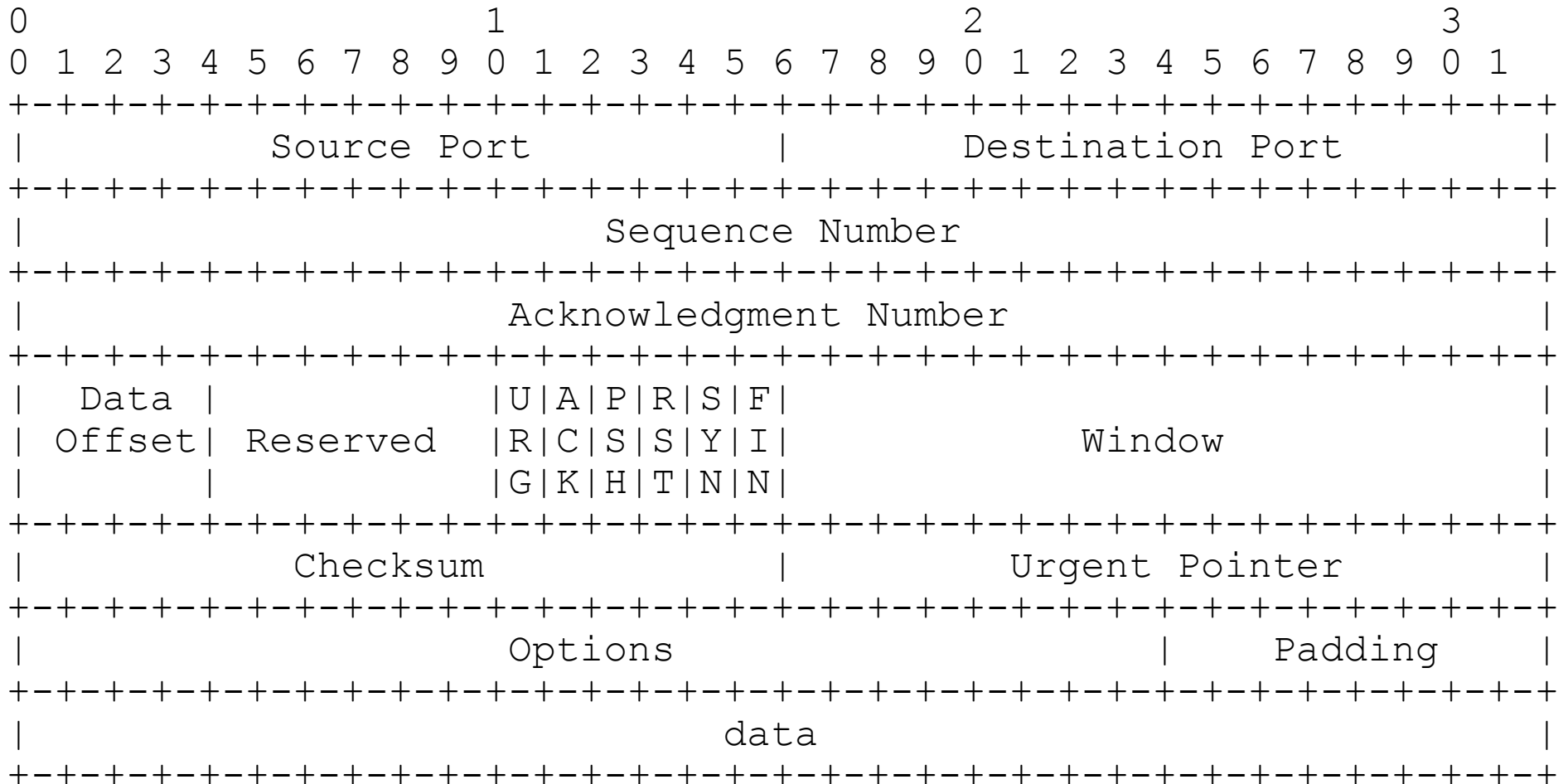
IP Paketheader

Aus RFC 791:



TCP Header

Aus RFC 793:



NAT

- Source NAT bei *ausgehenden* Paketen:
 - Jedem internen Client wird eine externe Adresse zugewiesen.
- Spezialfall Masquerade:
 - Viele interne Clients hinter einer externen Adresse verstecken.
- Destination NAT bei *eingehenden* Paketen:
 - Jedem internen Server wird eine externe Adresse zugewiesen.
- NA(P)T:
 - Nicht nur die IP-Adressen sondern auch die Ports werden umgesetzt.
 - In Kombination mit SRC- oder DST-NAT.

Paketfilter in Linux

- Kernel 2.0: Übernahme von ipfw aus *BSD.
 - Benutzerschnittstelle: `ipfwadm`
 - Basierend auf drei CHAINS: INPUT, OUTPUT und FORWARD
- Kernel 2.2: Komplette Neuüberarbeitung
 - Weiterhin drei CHAINS
 - NAT im Code eingebaut: MASQ
 - Benutzerschnittstelle: `ipchains`
- Kernel 2.4: iptables
 - Aufräumen und Modularisierung
 - Benutzerschnittstelle: `iptables`
- Zukunft: nftables

iptables: Grundlagen

- Das Netfilter Modul des Kernels stellt Regeln auf:
 - Pakete werden gegen eine Reihe von Regeln getestet.
 - Die Tests werden auf Grund von Ausdrücken entschieden.
 - Bei Erfolg des Tests gibt die Regel das weitere Schicksal des Paketes vor.
- Ein Regelsatz wird als Policy bezeichnet.
- Strukturen innerhalb des `iptables` – Befehls erlauben effiziente und verständliche (lesbare) Regelsätze zu erstellen.

tables - Tabellen

- Erste Strukturebene innerhalb von `iptables`.
- Trennung der Aufgaben in
 - filter: Paketfilter
 - nat: NAT
 - mangle: Bearbeitung/Veränderung von Paketen
 - raw: Alles andere
- Option `-t` im `iptables` Befehl. Vorgabe: `-t filter`
- Spezialisierung innerhalb der Tabellen. Vorteil gegenüber `ipchains`!

TARGETS - Ziele

- Targets sind die Anweisung, was der Kernel mit dem Paket machen soll.
- Möglich ist die Angabe aller CHAINS, der zweiten Strukturebene in `iptables`.
- Eingebaut sind auch:
 - ACCEPT: Paket ist gut und soll weiter verarbeitet werden.
 - DROP: Paket verwerfen.
 - QUEUE: Weiterverarbeitung des Paketes im Userspace.
 - RETURN: Zurück an die aufrufende CHAIN.
- Option `-j` in `iptables`.
- Beliebig erweiterbar über Module.

CHAINS - Ketten

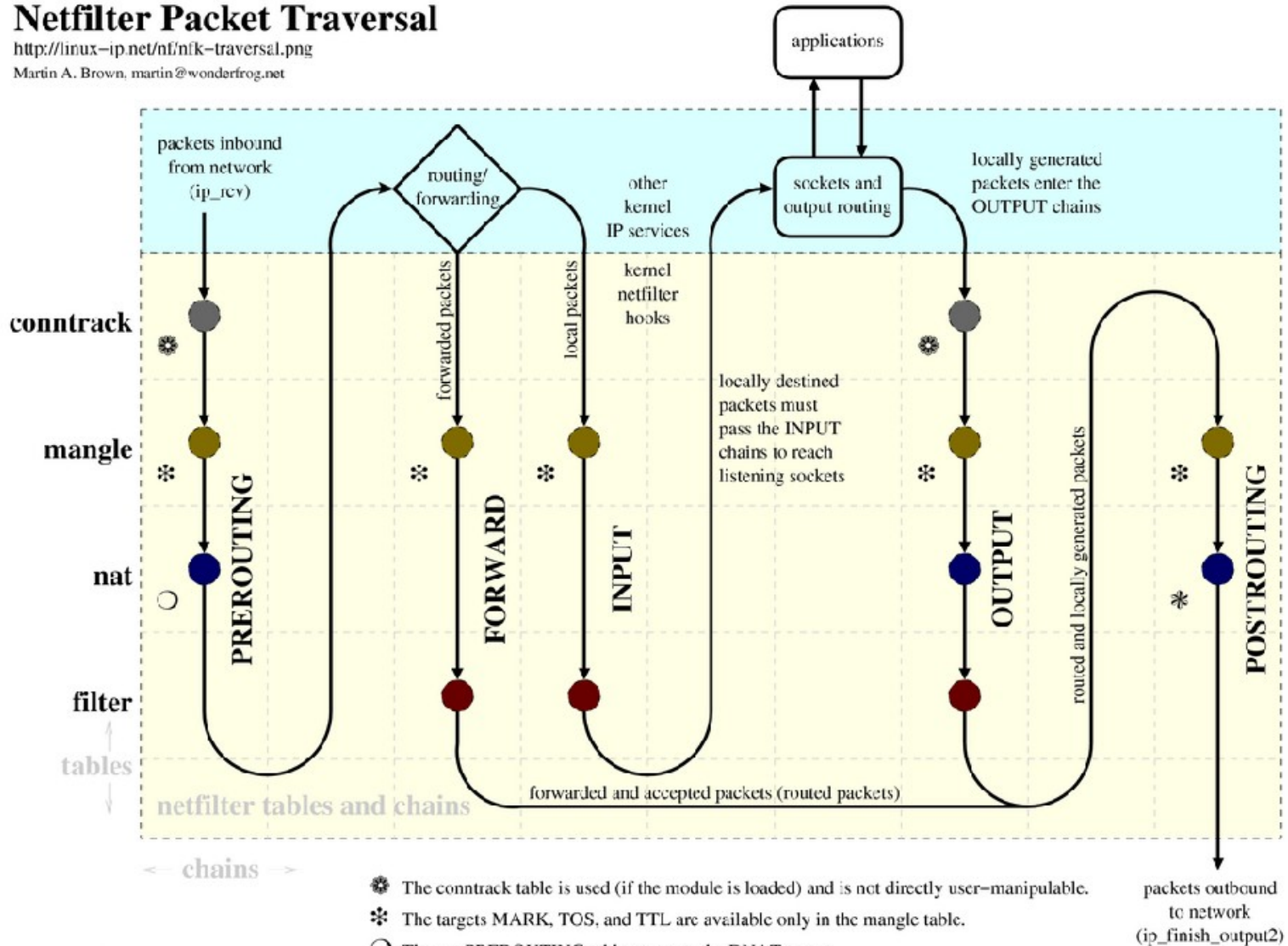
- Zweite Strukturebene: CHAINS.
- Syntax: CHAINS werden groß geschrieben!
- Unterschiedliche CHAINS für unterschiedliche Tabellen.
- Eingebaute CHAINS:
 - filter: INPUT, FORWARD und OUTPUT.
 - nat: PREROUTING und POSTROUTING.
 - mangle: INPUT, FORWARD und POSTROUTING

Weg eines Paketes durch den Kernel

Netfilter Packet Traversal

<http://linux-ip.net/nf/nfk-traversal.png>

Martin A. Brown, martin@wonderfrog.net



cf. <http://www.docum.org/qos/kp/d/>

cf. http://open-source.srkoo.net/kernel/kernel_net.png

cf. <http://iptables-tutorial.frozentux.net/>

⚙ The contrack table is used (if the module is loaded) and is not directly user-manipulable.

⚙ The targets MARK, TOS, and TTL are available only in the mangle table.

○ The nat PREROUTING table supports the DNAT target.

⚙ The nat POSTROUTING table supports SNAT and MASQUERADE targets.

Manipulation der Regelsätze

- Änderungen der Regelsätze mit dem iptables Befehl:
 - I Einbau einer neuen Regel am Anfang einer KETTE.
 - A Hinzufügen einer neuen Regel am Ende einer KETTE.
 - D Löschen der angegebenen Regel.
 - R Ersetzen der angegebenen Regel.
 - L Anzeigen eines Regelsatzes.
 - F Löschen aller Regeln (einer KETTE).
 - Z Löschen der Zählerstände.
 - N Neue KETTE anlegen.
 - P Vorgabe des Verhaltens der KETTE, wenn keine Regel passt.

Anzeigen des Regelsatzes

```
# iptables -L [CHAIN]
```

Mögliche Optionen

- n Keine Namensauflösung von IP-Adressen (schneller).
- v Ausführliche Ausgabe.

Module

- Netfilter bietet nur das Grundgerüst. Erweiterungen sind mit Modulen möglich.
- Das macht die Stärke von netfilter aus!
- Beispiele für Module:
 - Connection Tracking / State Tables
 - NAT helper Module
 - Verschiedene Limitierungen
 - Policy
 - Time
 - U32
- Module werden mit `-m` geladen

Connection Table

- Netfilter kann sich den Status von Verbindungen merken.
- Realisiert über ein Modul von `iptables`.
- Realisiert über eine interne Kernel Tabelle. Verzeichnet sind alle wichtigen Parameter einer Verbindung, wie
 - SRC und DST IP und Port.
 - Status der Verbindung, auch erwartete Verbindungen.
 - Anzahl der Pakete und Bytes.
 - ...
- Beispiel: Ein TCP SYN Paket erzeugt einen Eintrag (Status: `SYN_SENT`) in der Tabelle `/proc/net/ip_conntrack`

Das state – Modul

- Das Module von `iptables` wird mit der Option `-m state` geladen.
- Das `state` – Modul bietet weitere Optionen, die mit `--state` getestet werden:
 - **NEW:** Ein gültiges Paket, das einen neuen Eintrag im connection table bedingt, z.B. ein neues TCP SYN.
 - **ESTABLISHED:** Ein Datenpaket, zu dem schon ein Eintrag im connection table existiert, z.B ein Datenpaket innerhalb einer HTTP Session.
 - **RELATED:** Ein Paket, das sich auf einen bestehenden Eintrag bezieht.
 - **INVALID:** Ungültiger Status.

Beispiele zum state Modul

```
# iptables -I FORWARD -m state --state  
ESTABLISHED,RELATED -j ACCEPT
```

```
# iptables -I FORWARD -p tcp --dport 80 -m state  
--state NEW -j ACCEPT
```

- Regelsatz für eine Firewall vor einem Webserver (HTTP).
- Frage: Wie müssen die Regeln aussehen, wenn sie auf dem Server selbst durchgesetzt werden sollen?

iptables Befehl

```
# iptables [-t table] Command Options
```

- Je nach Kommando müssen/können verschiedene Optionen angegeben werden.
- Optionen sind die KETTE, Ausdrücke, Ziele, ...
- Ausdrücke werden aus Paramtern zusammengesetzt; diese sind u.a.:
 - p Protokoll, z.B. tcp, udp, icmp, ...
 - s Source: Quelladresse (IP)
 - d Destination Zieladresse (IP)
- Parameter können mit „!“ negiert werden, z.B. `-p !tcp`

Mehr Parameter

- `-i` Schnittstelle, auf der das Paket eingeht
- `-o` Schnittstelle, auf der das Paket hinausgeht.
- `-f` Das Paket ist ein weiteres Fragment zu einem vorhergehenden Paket.
- Beliebige weitere Optionen durch Module
- Verschiedene Module werden implizit geladen, z.B. `tcp`
 - Optionen: `--source-port` oder `--destination-port` bei Verwendung von `-p (udp|tcp)`

iptables NAT

- Source NAT nur in nat Tabelle und POSTROUTING Kette
 - Ziel: SNAT
 - Optionen: `--to-source ipaddr[-ipaddr][:port[-port]]`

```
# iptables -t nat -A POSTROUTING -o eth0 \  
-s 192.168.0.0/24 -j SNAT --to-source 82.135.103.106
```

- Destination NAT nur in nat Tabelle und PREROUTING Kette:
 - Ziel: DNAT
 - Optionen: `--to-destination ipaddr[-ipaddr][:port[-port]]`

```
# iptables -t nat -A PREROUTING -p udp -m udp \  
-d 82.135.103.105 --dport 53 -j DNAT \  
--to-destination 192.168.189.4
```

NAT in netfilter

- Alternative zu SNAT: MASQUERADE
- Netfilter pflegt in der conntrack Tabelle auch die NAT Informationen:

```
# conntrack -L -d 82.135.103.102
```

```
tcp 6 431952 ESTABLISHED src=84.151.61.70 dst=82.135.103.102  
sport=63702 dport=80 packets=2 bytes=112 src=192.168.189.4  
dst=84.151.61.70 sport=80 dport=63702 packets=1 bytes=60  
[ASSURED] mark=0 use=1
```

- Nicht jedes Protokoll eignet sich für NAT:
 - IP Adresse im Protokoll (FTP, ...)

Viele schöne Module und Ziele

- man `iptables` bringt so manche Erleuchtung.
- Viele Tricks möglich, die es bei anderen Paketfiltern auch für viel Geld (!) nicht gibt.
 - Meine Lieblingsmodule: `recent`, `limit`
 - Mein Geheimtipp als `TARGET`: `CLUSTERIP`

Logging

- Ziel: LOG
- Ausgabe an syslog()
- Viele Optionen des Ziels zur Ausführlichkeit des Logs.
- Das Ziel LOG beendet die Bearbeitung des Paketes nicht!

```
# iptables -N RULE_26
# iptables -A FORWARD -p tcp -m tcp -d 192.168.189.4 \
  -m multiport --dports 143,993,110,995,25,465 \
  -m state --state NEW -j RULE_26
# iptables -A RULE_26 -j LOG \
  --log-prefix "RULE_26 -- ACCEPT "
# iptables -A RULE_26 -j ACCEPT
```

- **Module limit hilfreich!**

Ein guter Regelsatz

- 1) Anti-Spoofing Regeln.
- 2) Regeln, die die Firewall selbst betreffen.
- 3) Aller sonstiger Verkehr zur Firewall wird blockiert.
- 4) Restliche Regeln.
- 5) DROP ALL Regel zum Schluss.

Logging nach Bedarf.

GUI: fwbuilder

- Geniale GUI zur Erstellung von Regelsätzen.
- Vor- und Nachteile von GUIs: Fast schon eine Glaubensfrage.
- Nicht nur für `iptables`. Auch für viele andere Paketfilter.
- GUI sehr flexibel.
- Alle zusätzlichen Module nutzbar.

MARK Target

- In der Tabelle mangle kann man das Paket markieren.

```
# iptables -t mangle -A PREROUTING (...) \  
    -j MARK --set-mark 1
```
- Diese Markierung kann man in vielen anderen Systemen nutzen:
 - Policy Based Routing von iproute2
 - Quality of Service (QoS) / Traffic Shaping (tc)
 - Load-Balancer (LVS)
 - ...
- Geht auch mit fwbuilder

Auswertung

- Ich habe nichts (freies) brauchbares gefunden.
- Ansprüche:
 - Online
 - Aggregation / Statistiken
 - Berichte
 - Suchfunktionen
- Möglichkeiten
 - FirewallEyes
 - ~~IPtables log analyser~~
 - Analog
 - 123LogAnalyser
 - ...

Logging und Auswertung


 displayed lines:
 log file:
 read from the end

auto-refresh
 resolv IP
 resolv services
 exact search

Jun 5	15:59:25	tun0	→	10.9.0.6	→	192.168.188.116	TCP	43307	→	www	0	ACCEPT
Jun 5	15:59:25	tun0	→	10.9.0.6	→	192.168.188.116	TCP	43306	→	www	0	ACCEPT
Jun 5	15:59:25	tun0	→	10.9.0.6	→	192.168.188.116	TCP	43305	→	www	0	ACCEPT
Jun 5	15:59:25	tun0	→	10.9.0.6	→	192.168.188.116	TCP	43304	→	www	0	ACCEPT
Jun 5	15:59:25	tun0	→	10.9.0.6	→	192.168.188.116	TCP	43303	→	www	0	ACCEPT
Jun 5	15:59:24	tun0	→	10.9.0.6	→	192.168.188.116	TCP	43302	→	www	0	ACCEPT
Jun 5	15:59:23	vlan100	→	141.68.238.11	→	82.135.103.106	UDP	500	→	isakmp	1	ACCEPT
Jun 5	15:59:21	vlan102	→	192.168.190.161	→	192.168.197.50	TCP	80	→	4037	51	DENY
Jun 5	15:59:20	vlan100	→	192.168.0.1	→	192.168.188.168	TCP	80	→	39289	51	DENY
Jun 5	15:59:20	eth0	→	192.168.188.169	→	192.168.189.4	TCP	45679	→	smtp	23	ACCEPT

Firewall Eyes - GPL - Creabilis © 2004 - Web site : <http://firewalleyes.creabilis.com>

Testwerkzeuge

- Hirnschmalz!
- Grundsätze:
 - Wenn es „langsam“ ist, ist es NIE die Firewall.
 - Wahrscheinliche Ursachen: Regeln, Routing, NAT
 - Wenn es langsam ist: DNS
- `tcpdump`: Liest auf der „Leitung“ mit.
 - Unentbehrlich zur Fehlersuche!
 - Auf welcher Schnittstelle kommt das Paket herein, auf welcher Schnittstelle verlässt es die Firewall
- `Nmap` ist ein Portscanner.
- `hping` erzeugt beliebige IP Pakete .

Praxis

- Viel Spass beim probieren.
- Wünsche / Anregungen / Fragen?