# 4. *iptables*-Befehle

***Anm.:* weitere Details zu den nun folgenden Beschreibungen findet man in der *Manual-Page* von *iptables* `(man iptables)`.**

Mit *iptables* ist es nun möglich die *Regeln* in einer *Kette* zu beeinflussen. Jede Kette besitzt eine *Standardregel* die benutzt wird wenn keine der sonstigen Regeln zutrifft. Sie können Regeln neu hinzufügen, vorhandene löschen oder verändern. Dies sind die *Befehle* von *iptables.*

## 4.1 Beispiel

Bevor ich die einzelnen Befehle und Optionen erkläre hier mal ein Beispiel:

```
#!/bin/sh

# 1 Iptables
FW="/sbin/iptables"

# 2 vorhandene Regeln & Ketten löschen
$FW -F
$FW -X
$FW -t nat -F

# 3 Standardregeln
$FW -P INPUT   ACCEPT
$FW -P FORWARD ACCEPT
$FW -P OUTPUT  ACCEPT

# 4 Ping blockieren
$FW -A INPUT -p icmp -j DROP
```

**zu 1:** Zunächst benutze ich in Scripten nicht immer den Befehl *iptables*, sondern zur besseren Lesbarkeit die Variable *FW.* Diese ist nichts weiter als ein Verweis auf: */sbin/iptables.* Denn wenn man nur *iptables* als Befehl einsetzt, also ohne absolute Pfadangabe läuft man Gefahr, dass nicht der wirkliche, unter */sbin* liegende Befehl benutzt wird (Stichwort: PATH-Variable). Und immer den absoluten Pfad anzugeben macht das Ganze unleserlich.
***Anm.:*** der Ablageort von *iptables* kann je nach Distribution auch unter */usr/sbin* sein!
**zu 2:** bevor man komplett neue Regeln erstellt bzw. anfügt sollte man sicherstellen, dass man zunächst einmal alles 'zurücksetzt'.
**zu 3:** hier werden die *Standardregeln* erstellt. Sie werden mit dem Parameter *-P* gesetzt. Die in unserem Beispiel lassen **alles** passieren.
**zu 4:** die *Standardregeln* lassen alles passieren. Wenn jedoch jemand einen *Ping* an unseren Rechner sendet wird dieser geblockt.

Dieses Beispiel macht unseren Rechner also offen für alles, außer für *Ping*. Dies ist eine (leider) häufig benutzte Firewall-Methode: *der Rechner lässt alles passieren, außer das was verboten ist.* Und wenn man ein Verbot (Regel) vergisst, ...
Die zweite, bessere Methode ist: *der Rechner blockiert alles, außer das was erlaubt ist*. D.h. die *Standardregeln* blocken und Sie werden *Regeln* erstellen die bestimmte Pakete akzeptieren. Wenn Sie nun eine Regel vergessen, wird dies schnell bemerkt, da z.B. der *SSH-Zugang* nicht funktioniert. Wenn man jedoch bei der ersten Methode eine Regel vergisst, fällt dies meist erst auf wenn alles zu spät ist. Die erste Methode wird oft nur deshalb benutzt weil jemand bestimmte Regeln, um etwas zuzulassen, nicht setzen kann (und daher *eDonkey* nicht läuft ;).

## 4.2 Befehlsliste

In dem vorherigen Beispiel wurde *iptables* mit verschiedenen *Befehlen* aufgerufen um die filter zurückzusetzen und neue Regeln zu definieren. Hier werde ich nun die wichtigsten Befehle erklären. Vorher sollten Sie das Beispiel in ein Script verfrachten, ausführbar machen und einmal starten. Ich würde empfehlen es **einzutippen** und nicht die

Zwischenablage zu benutzen. Denn nur dadurch lernt man was man tut!

| **Befehl** | **Kürzel** | **Beschreibung** |
|---|---|---|
| --flush | -F | Löscht alle Regeln |
| --list | -L | Listet die Regeln auf |
| --zero | -Z | Für jede Regel wird mitgezählt wie oft sie zutreffend war und wieviel Bytes übermittelt wurden. Mit diesem Parameter werden diese Zähler zurückgesetzt. |
| --policy | -P | Setzt die *Methode* (auch *Standardmethode*) einer Kette |
| --append | -A | Hängt eine neue Regel an die Kette an |
| --insert | -I | Fügt eine Regel anhand einer Nummer ein |
| --replace | -R | Ersetzt eine Regel anhand ihrer Nummer |
| --delete | -D | Löscht eine identische Regel oder anhand ihrer Nummer aus der Kette |
| --new-chain | -N | Erstellt eine benutzerdefinierte Kette; Beschreibung folgt in 6.3 |
| --delete-chain | -X | Löscht eine benutzerdefinierte Kette; Beschreibung folgt in 6.3 |

Wenn sie die Kürzel benutzen ist **Groß-/Kleinschreibung** wichtig. Es gibt nämlich Parameter die den gleichen Buchstaben verwenden, sich aber eben dadurch unterscheiden ob sie Groß oder Klein geschrieben wurden. Ich werde beim Erklären der *Befehle* noch weitere *Optionen* miteinbringen und diese ebenfalls erläutern.

# 4.3 Befehle anwenden

***Anm.:* zu vielen *Befehlen* werden *Parameter* benötigt. Diese werde ich in diesem Kapitel stillschweigend benutzen. Erklärt werden sie später, im Kapitel 5.**

## 4.3.1 Regeln löschen: *--flush / -F*

Man kann mit dem Befehl *--flush* alle vorhandenen *Regeln* oder nur die einer bestimmten *Kette* löschen. Bevor man eine komplett neue Regelliste erstellt, muß man logischerweise zuerst eventuell vorhandene entfernen.

```
# Löscht alle Regeln
iptables -F (oder iptables --flush)

# Löscht nur die Regeln der Kette INPUT
iptables -F INPUT
```

***Anm.:* Ich benutze nur die Befehle in ihrerer abgekürzten Form. Ansonsten wird eine Befehlszeile schnell sehr lang.**

## 4.3.2 Regeln auflisten: *--list / -L*

Zunächst führen wir unser Script aus und testen ob es funktioniert indem wir einen Ping an den Rechner senden (z.B. `ping localhost`). Dieser Ping müsste abgelehnt werden. Dann sehen uns die mal die Liste der Regeln an:

```
iptables -L
```

Es werden uns nun die *Standardregeln* der drei wichtigsten Ketten und unsere 'Ping'-Regel angezeigt. Hier ein Auszug:

```
Chain INPUT (policy ACCEPT)                          <- Standardregel der Kette INPUT
target     prot opt source               destination   <- Überschriften
DROP       icmp --  anywhere             anywhere      <- Ping-Regel
```

In der ersten Zeile steht die *Standardregel* der *Kette (Chain) INPUT*. Die *Methode (policy)* dieser Regel lautet *ACCEPT*. Sie akzeptiert also, wenn keine Regel zutrifft, alles. Die nächste Zeile enthält die Überschriften der einzelnen *Regeln*. Unter *target* steht das *Ziel* bzw. die *Methode* die angewandt wird, wenn die Regel zutrifft (also

nicht etwa das Ziel des Netzpaketes!). *prot* ist die Abkürzung für das *Protokoll* - hier ICMP (wird von *Ping* benutzt). Unter *Source* und *Destination* stehen die Quell- oder Zieladressen. Bei unserer Regel sind diese egal. D.h. egal wer einen Ping sendet - er wird abgelehnt. Das ganze was sie sehen ist die *filter-Tabelle.*

Unter 4.2 gab es einige Befehle die *Nummern* verlangen. Um diese Nummern anzuzeigen benutzt man zusätzlich die Option *--line-numbers*:

```
iptables -L --line-numbers
```

Hierdurch habe wir in der Auflistung eine weitere Spalte mit dem Namen *Num*. In dieser steht einfach die Nummer der Regel:

```
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    DROP       icmp --  anywhere             anywhere
```

Standardmäßig wurden bisher nur die Daten der Tabelle *filter* aufgelistet. Um auch die Informationen von *nat* und *mangle* aufzurufen muss man diese mit übergeben:

```
# für nat
iptables -t nat -L

# oder mangle
iptables -t mangle -L
```

Weitere zusätzliche Parameter sind:

*--verbose, -v:* Zeigt eine ausführlichere Liste (u.a. mit der Anzahl der Pakete und Bytes die gesendet/empfangen wurden)
*--numeric, -n:* IP- und Port-Nummern werden nicht in Namen aufgelöst (wie bei *route -n*)

### 4.3.3 Zähler zurücksetzen: *--zero / -Z*

*iptables* benutzt Zähler um festzustellen wieviel Pakete und welche Datenmengen über bestimmte Regeln gingen. Wenn man den Zähler der übertragenen Pakete zurücksetzen will benutzt man den Befehl *--zero*. Dies betrifft jedoch nur die Zähler der *Regeln* und nicht die der *Ketten.*

```
# Zähler anzeigen
iptables -L -v

iptables -Z
# oder nur für eine Kette
iptables -Z OUTPUT

# Zähler wieder anzeigen
iptables -L -v
```

### 4.3.4 Standardverhalten festlegen: *--policy / -P*

Wie bereits im Beispiel angesprochen besitzt jede *Kette* eine *Standardmethode*. Diese werden mit dem Befehl *--policy* festgelegt. Sie werden angewandt wenn keine der *Regeln* zutreffend war.

```
# die Kette FORWARD lässt alles durch
iptables -P FORWARD ACCEPT
```

### 4.3.5 Regeln anhängen: *--append / -A*

Dies ist der wohl am meisten benutzte *iptables-Befehl*. Mit ihm werden Regeln an die vorhandenen **angehängt**. In dem obigen Beispiel haben wir die *Ping-Regel* angehängt:

```
iptables -A INPUT -p icmp -j DROP
```

Nach dem *Befehl* (-A) **muss** die *Kette* (INPUT) zu der die Regel gehört angegeben werden. Ab nun folgen zunächst

ein oder mehrere Parameter mit denen man die Regel bis ins kleinste Detail spezifizieren kann. Ich habe hier nur den Parameter *protocol* (-p) benutzt und diesem als Argument das Protokoll *icmp* übergeben. Am Ende wird das Ziel übergeben, das angewandt wird wenn die Regel zutrifft: *DROP* - das Paket wird verworfen.

### 4.3.6 Regeln einfügen: *--insert / -I*

Wie bereits erwähnt werden die Regeln ja **nacheinander** abgerufen (siehe auch 3.6) und wenn eine Zutrifft wird die Verarbeitung in der *Tabelle* abgebrochen. Unsere *Ping-Regel* lässt ja nun überhaupt keinen Ping mehr zu. Nun möchten wir aber dass ein Ping vom *lokalen Netzwerk* ankommt und beantwortet wird. Eine Regel mit *--append* anzuhängen würde nichts bringen. Daher müssen wir mit *--insert* diese Regel **vor** der Vorhandenen plazieren. Hierzu benötigen wir zuerst deren Nummer:

```
iptables -L --line-numbers
```

Nun können wir die Regel an deren *Nummer* plazieren. Hierdurch rücken alle folgenden Regeln um einen Platz weiter:

```
iptables -I INPUT 1 -p icmp -s 192.168.0.0/24 -j ACCEPT
```

Ab jetzt werden *Pings* aus dem lokalen Netzwerk (*-s 192.168.0.0/24*) zugelassen, da die *ACCEPT-Regel* **vor** der *DROP-Regel* steht. Dieser Befehl wird allerdings seltener benötigt. Beim Erstellen eines Scripts für alle Regeln eines Rechners kümmert man sich ja schon um die Reihenfolge und benutzt daher nur *--append*. Die Übergabe der *Nummer* ist *optional*. Wenn man keine angibt wird die Nummer 1 benutzt.

### 4.3.7 Regeln ersetzen: *--replace/ -R*

Dieser Befehl wird wie --insert angewendet, nur mit dem Unterschied, dass die vorhandene **überschrieben** wird.

### 4.3.8 Regeln löschen: *--delete / -D*

Mit *--delete* können Regeln aus einer Kette entfernt werden. Als Argumente wird entweder die *Nummer* oder die *Regel* benutzt.

```
# Löscht die Regel Nr. 2
iptables -D INPUT 2

# Löscht die Regel auf die die Syntax zutrifft
iptables -D INPUT -p icmp -s 192.168.0.0/24 -j ACCEPT
```

---