3. *iptables*-Grundlagen

Was *iptables* ist habe ich schon in der Einführung besprochen. In diesem Abschnitt werde ich ihnen nun die Grundlagen zu *iptables* erklären. Der Aufbau von *iptables* besteht aus *Tabellen*, *Ketten* und *Regeln*. Die *Tabellen* beinhalten mehrere *Ketten* und diese widerrum mehrere *Regeln*.

3.1 Regeln

Mit einer *Regel* wird entschieden was mit einem Paket passieren soll. Jede besitzt bestimmte Parameter nach denen sie überprüft ob die Informationen eines Paketes auf sie zutreffen. Wenn die Parameter zutreffend sind, wird das Paket meist an ein neues *Ziel* verwiesen oder es wird eine *Methode* angewandt. Für die Bearbeitung der Pakete gibt es mehrere *Ziele* und *Methoden (policy)*. Häufig benutzte sind:

- *ACCEPT:* das Paket kann passieren
- *REJECT:* das Paket wird zurückgewiesen und ein Fehlerpaket wird gesendet
- *LOG:* schreibt einen Eintrag in die *syslog*
- *DROP:* das Paket wird ignoriert und keine Antwort gesendet
- *REDIRECT:* die Ziel-Adresse des Paketes wird hiermit so verändert, dass es zum lokalen Rechner gesendet wird
- *MASQUERADE:* die *Quell-Adresse* des Paketes wird durch die *IP-Adresse* der Schnittstelle ersetzt, auf dem es den Rechner verlässt

Regeln sind die *Glieder* der *Kette*.

3.2 Ketten

Die *Ketten* sind eine Sammlung von *Regeln*. D.h. das jede *Kette* mehrere *Regeln* besitzen kann um ein Paket durchzulassen oder zu blockieren. Es sind **fünf** Typen von Standardketten vorhanden. Manche dieser *Ketten* werden von allen Paketen und einige nur, je nachdem welches Ziel sie haben, durchlaufen. Die *Regeln* einer *Kette* werden **nacheinander** abgearbeitet und wenn eine zutrifft, ist die Bearbeitung in dieser Kette beendet (es gibt Ausnahmen):

- *PREROUTING:* alle Pakete kommen hier durch **bevor** eine *Routing*-Entscheidung getroffen wird
- *FORWARD:* für alle Pakete, die von der **einen** zu einer **anderen** Netzwerkschnittstelle weitergeleitet werden - also **keine** Pakete die an einen lokalen Dienst gerichtet sind
- *INPUT:* für Pakete die über eine Schnittstelle **hereinkommen** und einen Dienst auf dem Rechner ansprechen
- *OUTPUT:* für die über eine Schnittstelle **herausgehenden** Pakete, die von einem lokalen Dienst kommen
- *POSTROUTING:* alle Pakete kommen am **Ende** der Verarbeitung hier durch

Die Ketten *INPUT, FORWARD* und *OUTPUT* besitzen immer eine *Standardregel*. Diese wird dann angewandt, wenn keine der in der jeweiligen Kette vorhandenen *Regel* zutrifft oder keine Regel vorhanden ist. In den Ketten *PREROUTING* und *POSTROUTING* können nur Pakete manipuliert (*mangle* und/oder *nat*), nicht jedoch gefiltert werden. Zu den vorhandenen *Ketten* kann man noch *benutzerdefinierte Ketten* erstellen. Dazu kommen wir jedoch erst später.

3.3 Tabellen

Die Art der Verarbeitung von Netzwerkpaketen ist in verschiedene *Tabellen* unterteilt. *iptables* besitzt standardmäßig **drei** *Tabellen*: *mangle*, *nat* und *filter*. Die Tabelle *mangle* (übersetzt: *zerhauen*) ermöglicht es dem Kernel, Daten im Paket-Header zu verändern. *NAT* wird benutzt um interne und externe IP-Adressen zu übersetzen (= *Network Adress Translation*). Regeln in dieser Tabelle ändern die IP und/oder den Port des Ziels. Die Tabelle *filter* prüft alle für die Firewall ankommenden Pakete und entscheidet ob sie durchgelassen oder geblockt werden.

Jede dieser Tabellen besitzt nun mehrere *Ketten*:

- *mangle* (*Paketmanipulationen):* enthält alle Ketten
- *nat (Network Adress Translation):* enthält die Ketten *PREROUTING, OUTPUT* und *POSTROUTING*

- *filter* (*Paketfilter):* enthält die Ketten *FORWARD, INPUT* und *OUTPUT*

3.4 Zusammenfassung *Tabellen, Ketten* und *Regeln*

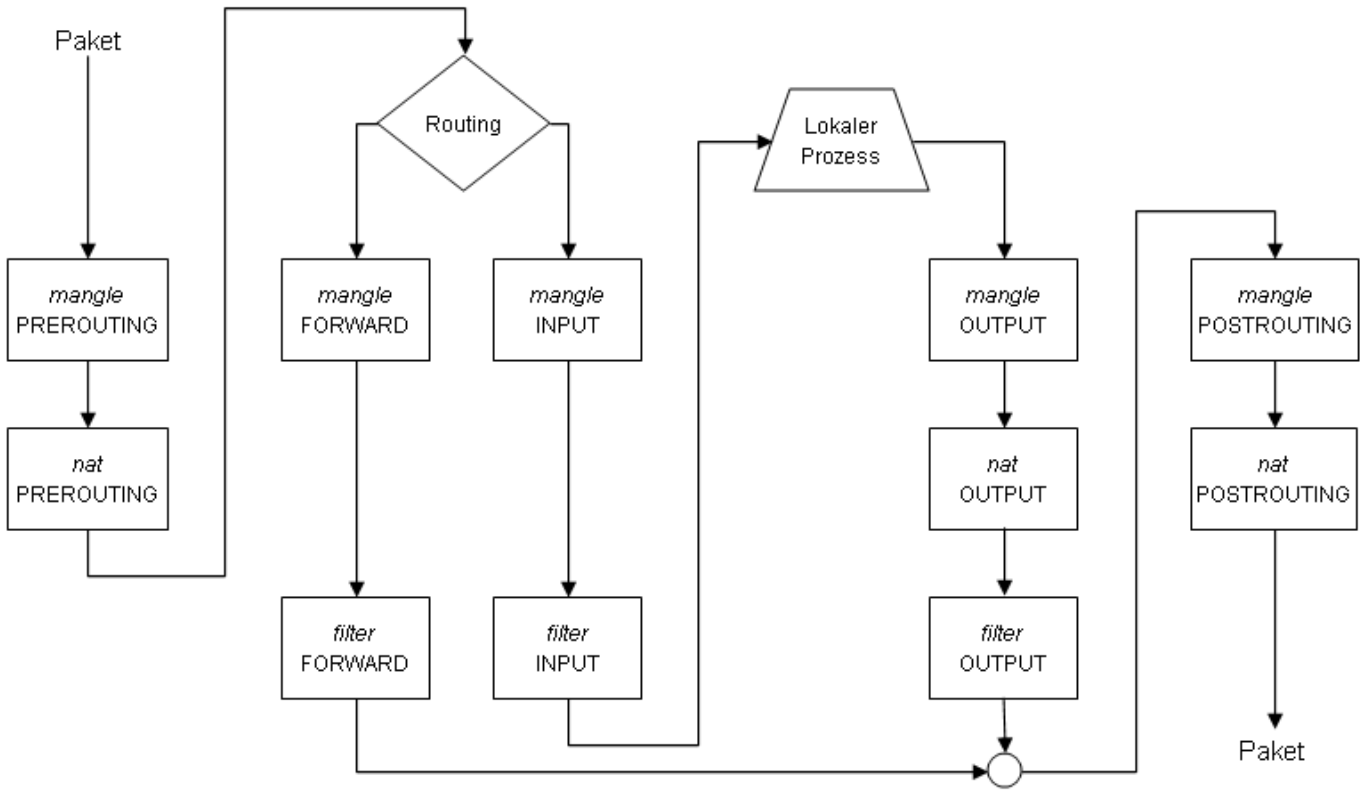
Sie wissen nun grob was *Tabellen, Ketten* und *Regeln* sind. Im folgenden sehen sie ein Listing einer simplen *filter-Tabelle*. In der linken Spalte (fettmarkiert) stehen die Beschreibungen der Zeilen. Das ganze Listing ist die *Tabelle filter*. Sie enthält die Ketten *INPUT, FORWARD* und *OUTPUT*. Jede der Ketten hat im Moment die *Standardregel ACCEPT*. D.h. sie akzeptieren alle Pakete - wenn nicht eine vorhandene Regel dies verbietet. Nur die Kette *INPUT* besitzt eine *Regel*. Diese Regel ignoriert Ping-Anfragen.

```
Tabelle FILTER

Kette INPUT
Standardregel:    Chain INPUT (policy ACCEPT)
Überschrift:      target     prot opt source               destination
Regel:            DROP       icmp --  anywhere             anywhere

Kette FORWARD
Standardregel:    Chain FORWARD (policy ACCEPT)
Überschrift:      target     prot opt source               destination

Kette OUTPUT
Standardregel:    Chain OUTPUT (policy ACCEPT)
Überschrift:      target     prot opt source               destination
```

In diesem Beispiel wird die Ping-Anfrage bereits beim *INPUT* geblockt. Hierdurch werden **keine** weiteren Regeln verarbeitet - das Paket ist ja weg. Wenn man nun die Methode von *DROP* auf *ACCEPT* ändert wird es in der Kette *INPUT* akzeptiert und noch die Kette *OUTPUT* abgearbeitet.

3.5 Ablaufreihenfolge

Nun stellt sich die Frage in welcher Reihenfolge ein Paket die *Tabellen* und *Ketten* durchläuft? Dies soll die folgende Abbildung erläutern (in der ersten Zeile steht immer der Name der *Tabelle* und in der zweiten der Name der *Kette*):



Ein Paket passiert noch **vor** einer Routing-Entscheidung die *PREROUTING-Ketten*. Und zwar zuerst die der Tabelle

mangle (Daten im Paket-Header können verändert werden) und dann die Kette der Tabelle *nat* (IP-Adressen können verändert werden). Dann wird vom *Routing* entschieden ob es für einen lokalen Dienst (z.B. Mailabruf, Samba-Freigabe usw.) auf dem Rechner bestimmt ist oder ob es 'geforwarded' wird. Es werden also entweder die Ketten *FORWARD* oder, nacheinander, die Ketten *INPUT* und *OUTPUT* passiert. Danach kommt noch, egal wie gerouted wurde, die Kette *POSTROUTING*.
Nun erkennt man auch, dass sich eine *Regel* in der Kette *INPUT* **nicht** auf ein Paket auswirkt das 'geforwardet' wird und umgekehrt. *Regeln* in den *PRE*- und *POSTROUTING-Ketten* werden jedoch auf **alle** Pakete angewandt.
Wenn Sie nun noch Verständnisprobleme haben, sollten sie sich das Bild ausdrucken und Kapitel 3 noch einmal durchlesen.

***Anm.:* für diejenigen, denen diese Abbildung nicht ausreicht, hier noch eine Detailreichere von *Mark A. Brown* (Homepage: http://linux-ip.net). Ich habe mich an diesem orientiert und es ein wenig 'abgespeckt'.**

3.6 Regelverarbeitung

In welcher Reihenfolge die *Tabellen* und *Ketten* abgearbeitet werden wissen sie nun. Wie aber werden *Regeln* abgearbeitet? Beim Erstellen von Regeln bekommen diese eine fortlaufende Nummer und werden anhand derer **nacheinander** abgearbeitet. Wenn nun eine *Regel* zutrifft wird die Verarbeitung in der entsprechenden *Kette* beendet.

Ich erkläre das ganze noch einmal an einem Beispiel:

- Ich erstelle eine *Regel* die einen Ping **nicht** zulässt - egal von wo er kommt.
- Danach füge ich eine Regel an, die einen Ping aus dem **lokalen** Netz zulassen soll
- Was passiert? Der Ping wird **nie** ankommen - egal ob vom externen oder lokalen Netz, da die erste Regel ihn ablehnt und die Zweite garnicht mehr abgearbeitet wird.
- Würde ich die Regeln umgekehrt einfügen, wäre alles in Ordnung: ein Ping vom lokalen Netz wird passieren gelassen. Ein Ping vom externen Netz wird von der Regel, die das lokale Netz betrifft nicht verarbeitet (er hat ja eine andere Quelle und damit ist die Regel nicht zutreffend) und er wird von der zweiten Regel geblockt.

Daran sehen sie wie wichtig die korrekte **Reihenfolge** der Regeln ist, da sie sich auch gegenseitig aufheben können oder ein *blockieren* nichts bringt wenn vorher *akzeptiert* wurde.