

Ralf Spenneberg

Linux-Firewalls mit iptables & Co.

Sicherheit mit Kernel 2.4 und 2.6
für Linux-Server und -Netzwerke



 ADDISON-WESLEY

An imprint of Pearson Education

München • Boston • San Francisco • Harlow, England
Don Mills, Ontario • Sydney • Mexico City
Madrid • Amsterdam



31 Etables

Der `etables`-Befehl ist ein zusätzlicher Befehl zur Filterung von Netzwerkpaketen auf einer Linux-Bridge. Mit diesem Befehl können Sie auch Nicht-IP-Pakete eingeschränkt filtern. Außerdem können Sie entscheiden, ob ein Paket geroutet oder gebridget werden soll. Der Aufbau eines kombinierten Routers mit Bridge-Funktionalität ist für Nicht-IP-Pakete möglich. Derartige Systeme werden als Router bezeichnet.

Die Anwendung des Befehls erfordert zusätzliche Patches für die 2.4er Linux-Kernel. Diese Patches sind in den 2.6er Kernen bereits enthalten und bei den meisten Distributionen auch aktiviert. Viele Distributionen liefern jedoch den Befehl selbst nicht mit. Daher wird zunächst die Installation des Befehls beschrieben.

31.1 Etables-Installation

Die letzte Etables-Version ist vom 1. November 2003. Falls Ihre Distribution (zum Beispiel Fedora Core 4) bereits Etables enthält, handelt es sich wahrscheinlich um diese Version, und ein Update ist nicht erforderlich. Für den Fall, dass Ihre Distribution nicht das Paket enthält und Sie möglicherweise auch noch den Kernel 2.4 einsetzen, schildere ich im Weiteren kurz die Installation des Pakets.

31.1.1 Konfiguration des Linux-Kernels

Während der Linux-Kernel 2.6 bereits den notwendigen Code enthält, müssen Sie den Linux-Kernel 2.4 noch patchen. Unter <http://etables.sf.net> finden Sie für den Linux-Kernel 2.4 die notwendigen Patches. Um den Patch anzuwenden, wechseln Sie in das Quelltextverzeichnis Ihres Kernels und rufen den Befehl `zcat patch.gz | patch -p1` auf. Anschließend müssen Sie Ihren Kernel noch konfigurieren. Zusätzlich zu den üblichen Iptables-Optionen wählen Sie die Option *802.1d Ethernet Bridging* an. Dann erscheint unter dieser Option die Option *Bridge: etables*. Wählen Sie diese und die anschließend erscheinenden Optionen ebenfalls aus.

Den Linux-Kernel 2.6 müssen Sie nicht mit einem Patch vorbereiten. Hier müssen Sie nur darauf achten, dass die entsprechenden Optionen ausgewählt wurden. Auch hier benötigen Sie die Option *802.1d Ethernet Bridging*, und unter *Bridge: Netfilter configuration* wählen Sie die gewünschten Etables-Funktionalitäten.

31.1.2 Installation des Userspace-Werkzeugs

Um den Befehl `etables` zu installieren, laden Sie zunächst das Paket `etables-2.0.6.tar.gz` von der Homepage und packen es aus. Anschließend übersetzen Sie das Paket mit dem Befehl `make`. Bei dem anschließenden `make install` können Sie mit den Variablen `KERNEL_INCLUDES`, `LIBDIR`, `MANDIR`, `BINDIR`, `ETCDIR`, `ETHERTYPEPATH` und `DESTDIR` die Zielverzeichnisse angeben.

Die Nutzung des CVS-Verzeichnisses, um den aktuellsten Code einzusetzen, ist nicht empfehlenswert, da sich hier seit November 2003 kaum Änderungen ergeben haben.

31.2 Die Etables-Tabellen

Etables verfügt über insgesamt 3 Tabellen. Die `broute`-Tabelle enthält die `BROUTING`-Kette. Die `filter`-Tabelle enthält die Ketten `FORWARD`, `INPUT` und `OUTPUT`. Die `nat`-Tabelle schließlich enthält die `PREROUTING`-, `OUTPUT`- und `POSTROUTING`-Ketten. Abbildung 31.1 zeigt die Ketten.



Achtung

Die Etables-Tabellen haben keinerlei Bezug zu den Iptables-Tabellen und Ketten. Es handelt sich um vollkommen eigenständige Tabellen und Ketten. Die `filter` und die `nat-OUTPUT`-Ketten sind ebenfalls eigenständig und werden nacheinander (zuerst `nat`, dann `filter`) durchlaufen.

Mit der `BROUTING`-Kette der `broute`-Tabelle können Sie einen Brouter aufbauen. Ein Brouter nach Definition des Maintainers des Etables-Codes ist ein System, das zwei Netze verbindet und einige Pakete routet (zum Beispiel IP-Pakete), während es andere Frames (zum Beispiel NETBEUI-Frames) über die Bridge weiterleitet. Dabei können Sie in der `BROUTING`-Kette entscheiden, ob Sie die Frames/Pakete routen oder bridgen möchten.

Wie bereits weiter oben ausgeführt wurde, erfolgt das Bridging auf der Schicht 2 (Data-Link) des OSI-Modells. Dies ist die Schicht 1 des TCP-Modells. Das Routing erfolgt auf der Schicht 3 des OSI-Modells beziehungsweise auf der Schicht 2 des TCP-Modells. Pakete auf der Data-Link-Schicht des OSI-Modells werden üblicherweise als Rahmen (Frame) bezeichnet, während der Begriff »Paket« für die Netzwerkschicht (Schicht 3) reserviert ist. Ich werde im Folgenden versuchen, diese beiden Begriffe korrekt zu verwenden.

Wie durchlaufen nun die Frames/Pakete die einzelnen Ketten? Um dies nachzuvollziehen, ist es sinnvoll, einige Szenarien durchzuspielen.

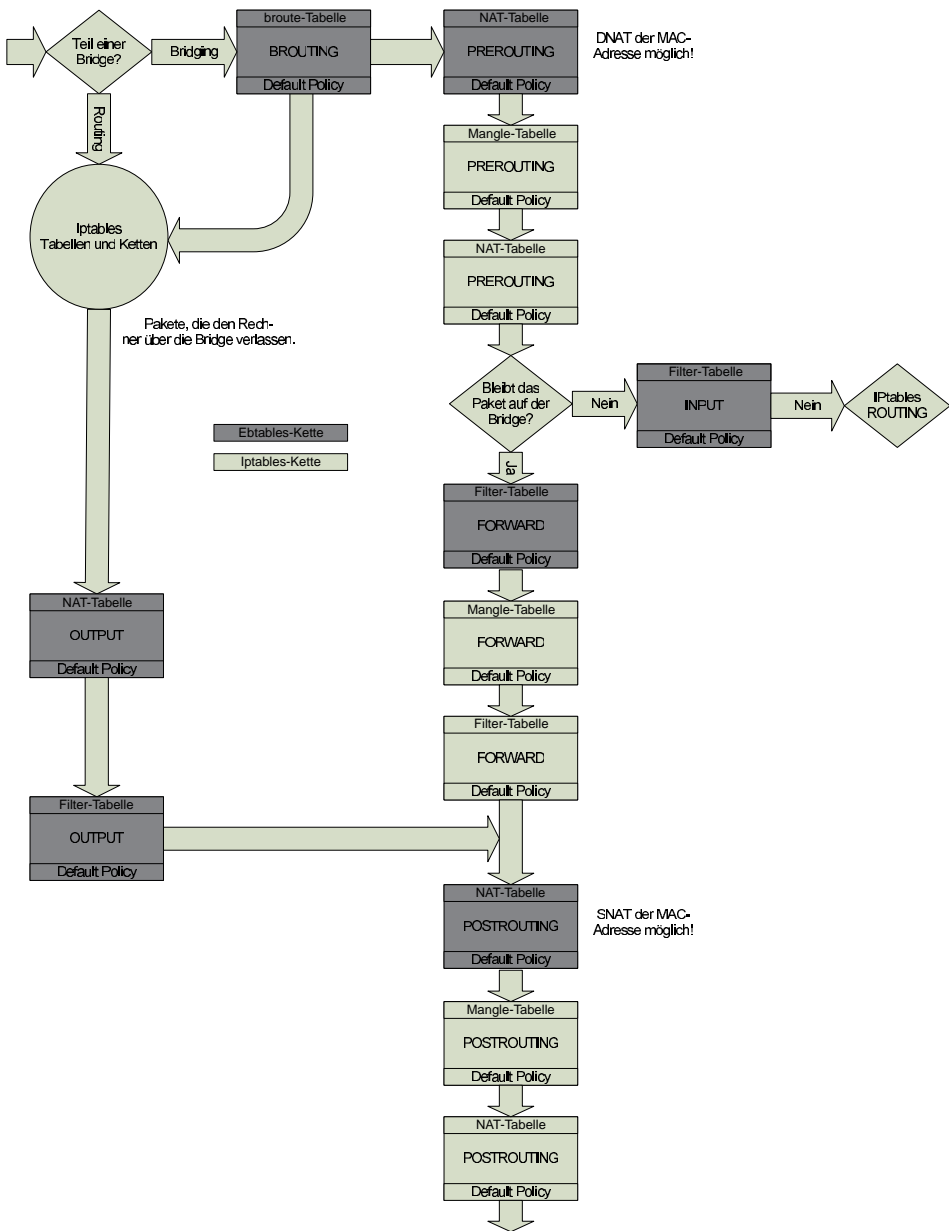


Abbildung 31.1: Etables verfügt über drei eigene Tabellen.

31.2.1 Der Rahmen erreicht über ein Bridge-Interface das System

Wenn das Paket über ein Bridge-Interface das System erreicht, wird in der Abbildung 31.2 das Paket an die `BROUTING`-Kette weitergegeben.

Hier können Sie mit einer Regel entscheiden, ob das Paket geroutet oder der Rahmen gebridgt werden soll. Default ist das Bridging. Bleibt der Rahmen in dem Bridging-Code, so wird er an die `PREROUTING`-Ketten weitergegeben. Hier werden nacheinander die Etables- und die Iptables-`PREROUTING`-Ketten durchlaufen. Anschließend erfolgt die Prüfung, ob das Paket gebridgt werden muss oder ob sein Ziel das lokale System ist. Ist Letzteres der Fall, so wird der Rahmen an die Etables-`INPUT`-Kette weitergereicht und anschließend weiter an die Netzwerkschicht geschickt. Dort wird dann die Routen-Entscheidung getroffen.

Muss der Rahmen gebridgt werden, so wird er durch die Etables- und Iptables-`FORWARD`-Ketten und die entsprechenden `POSTROUTING`-Ketten an das entsprechende Interface gesendet.

31.2.2 Der Rahmen erreicht über ein Nicht-Bridge-Interface das System

Falls das Paket über ein Bridge-Interface das System verlassen soll, durchläuft das Paket ganz normal die `mangle`- und `nat-PREROUTING`-Ketten. Wird das Paket geroutet, merkt der Kernel, dass das Paket über ein gebridgtes Interface den Rechner verlässt. Das Paket durchläuft die `mangle`- und `filter-FORWARD`-Ketten von Iptables und wird dann nach der Bridging-Entscheidung über die Etables-`nat`- und `-filter-OUTPUT`-Ketten an die Etables-`POSTROUTING`- und die Iptables-`POSTROUTING`-Ketten gesendet.

31.2.3 Ein lokal erzeugtes Paket verlässt das System über die Bridge

Dies ist der dritte mögliche Fall. Hierbei erzeugt ein lokaler Prozess ein Paket. Dies wird zunächst von den Iptables-`OUTPUT`-Ketten der `mangle`-, `nat`- und `filter`-Tabellen geprüft. Anschließend wird es über die Etables-`OUTPUT`-Ketten wie im letzten Fall transportiert.

31.3 Die broute-Tabelle

In manchen Umgebungen kann es sinnvoll sein, eine Bridge mit einem Router auf denselben Netzwerkkarten zu kombinieren (siehe Abbildung 31.3).

Dies kann zum Beispiel dann interessant sein, wenn Sie noch einige nicht routing-fähige Protokolle in Ihrem Netzwerk einsetzen. Dies können zum Beispiel `NET-BEUI`, `Appletalk` oder `DEC-LAT` sein. Während ein normaler IP-Router diese Protokolle nicht weiterleiten würde, kann dies eine Bridge für diese Rahmen.

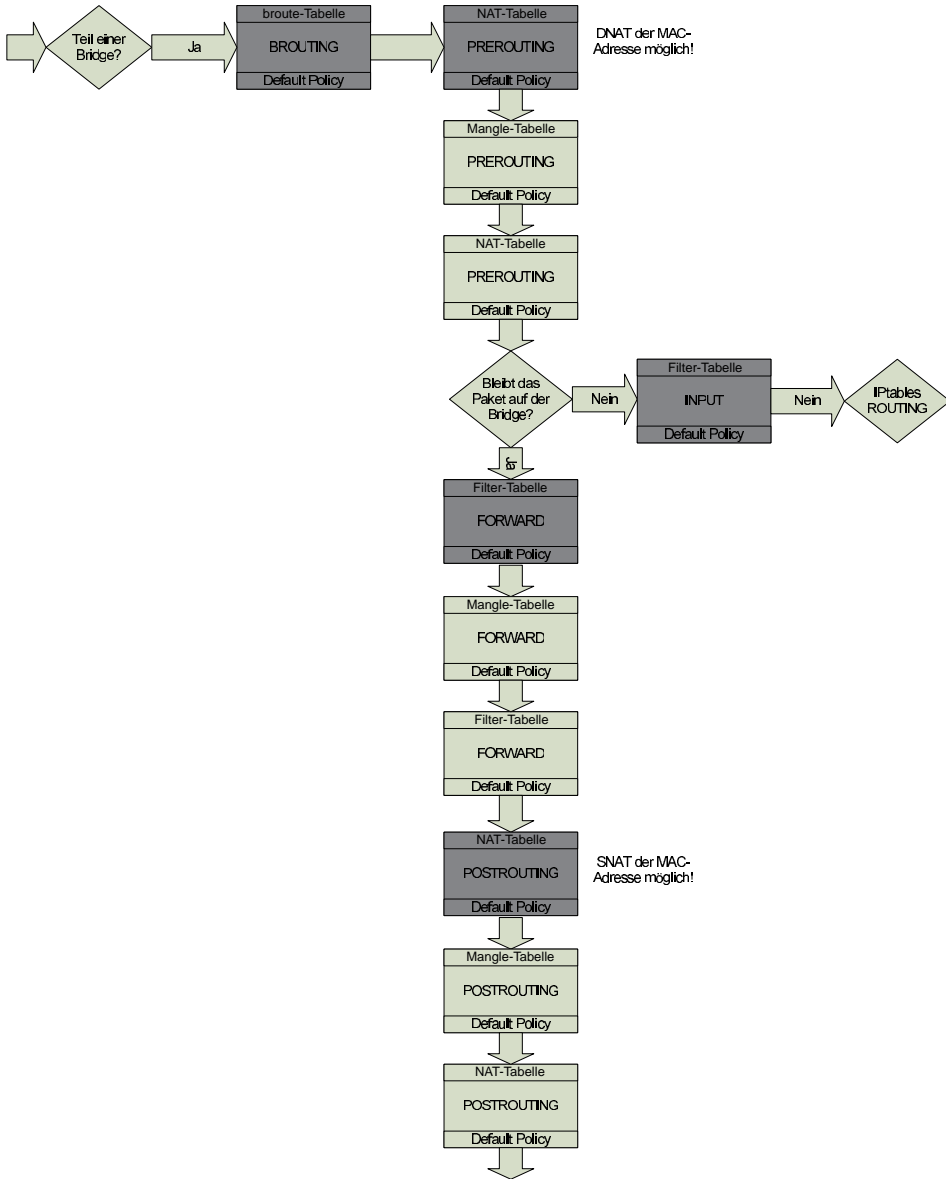


Abbildung 31.2: Die Ebtables- und Iptables-Ketten werden nacheinander durchlaufen.

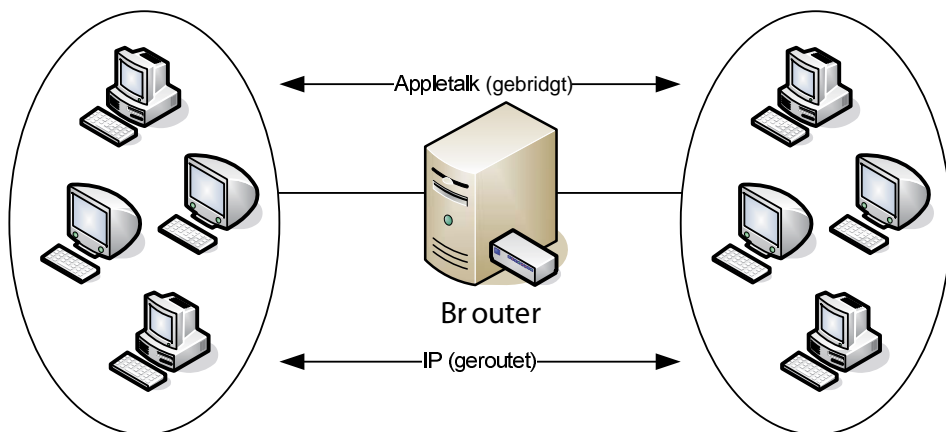


Abbildung 31.3: Ein Brouter routet einige Pakete, während er andere Rahmen bridgt.

In der `BRROUTING`-Kette der `broute`-Tabelle können Sie entscheiden, ob ein Rahmen, der normalerweise gebridgt werden würde, doch geroutet werden soll. Hierfür müssen Sie hier eine Regel hinzufügen, die das entsprechende Paket verwirft (`DROP`). Alle Pakete, die in der `BRROUTING`-Kette verworfen werden, verlassen die `Data-Link`-Schicht und werden an den Routing-Prozess und die `Netzwerk`-Schicht weitergegeben. Um zum Beispiel `IP`-Pakete zu routen, `ARP`-Anfragen zu verwerfen und alle weiteren Rahmen über die `Bridge` weiterzuleiten, können Sie folgendes Skript verwenden:

```
EBTABLES=/sbin/ebtables
BRCTL=/usr/sbin/brctl
IP=/sbin/ip

$EBTABLES -t filter -F
$EBTABLES -t nat -F
$EBTABLES -t broute -F

$BRCTL addbr br0
$BRCTL addif br0 eth0
$BRCTL addif br0 eth1

$IP addr add 172.16.1.1/24 dev eth0
$IP addr add 172.16.2.1/24 dev eth1
$IP link set eth0 up
$IP link set eth1 up
$IP link set br0 up

$EBTABLES -t broute -A BRROUTING -p IPv4 -j DROP
$EBTABLES -t broute -A BRROUTING -p ARP -j DROP
```

31.4 Die ebtables-Syntax

Der `ebtables`-Befehl verhält sich sehr ähnlich dem `iptables`-Befehl. Jedoch gibt es einige wesentliche Unterschiede, die hier kurz zusammengefasst werden sollen. Außerdem gibt es beim `ebtables`-Befehl einige zusätzliche Matches und Targets, die ich im Weiteren erläutern werde.

Zunächst weist der Befehl `ebtables` die identischen Kommandos wie der Befehl `iptables` auf. Es gibt im Einzelnen:

- `-A`, `--append`. Anhängen einer Regel.
- `-D`, `--delete`. Löschen einer Regel.
- `-P`, `--policy`. Ändern der Kettenrichtlinie.
- `-F`, `--flush`. Löschen einer Tabelle oder Kette.
- `-Z`, `--zero`. Löschen der Zähler einer Tabelle oder Kette.
- `-L`, `--list`. Anzeige der Regeln. Hier gibt es einen Unterschied. Zunächst zeigt dieser Befehl die Regeln in einem anderen Format an:

```
# ebtables -t broute -L
Bridge table: broute

Bridge chain: BROUTING, entries: 2, policy: ACCEPT
-p IPv4 -j DROP
-p ARP -j DROP
```

Um zusätzlich auch Zeilennummern angezeigt zu bekommen, müssen Sie auch die Option `--Ln` angeben. Die Rahmen- und Byte-Zähler erhalten Sie mit `--Lc`:

```
# ebtables -t broute -L --Ln --Lc
Bridge table: broute

Bridge chain: BROUTING, entries: 2, policy: ACCEPT
1. -p IPv4 -j DROP , pcnt = 1016 -- bcnt = 320521
2. -p ARP -j DROP , pcnt = 3 -- bcnt = 138
```

Wenn Sie die Regeln so anzeigen möchten, dass Sie diese direkt in ein Skript übernehmen können, müssen Sie die Option `--Lx` beim Befehl spezifizieren:

```
# ebtables -t broute -L --Lx
ebtables -t broute -A BROUTING -p IPv4 -j DROP
ebtables -t broute -A BROUTING -p ARP -j DROP
```

Schließlich zeigt die Option `-Lmac2` die MAC-Adressen an.

- `-N`, `--new-chain`. Erzeugt eine benutzerdefinierte Kette.
- `-X`, `--delete-chain`. Löscht eine benutzerdefinierte Kette.
- `-E`, `--rename-chain`. Benennt eine benutzerdefinierte Kette um.

- `--init-table`. Initialisiert die Tabelle. Dies löscht alle Regeln und setzt die Richtlinie (Policy) auf den Default zurück.
- `--atomic-init`. Das Kommando `etables` kann die Regeln einer Tabelle aus einer Datei lesen. Außerdem kann jeder Befehl auch auf eine Datei angewendet werden (siehe `--atomic-file`). Mit diesem Befehl initialisieren Sie die Datei.
- `--atomic-save`. Dieses Kommando speichert den aktuellen Zustand einer Etables-Tabelle in der angegebenen Datei.
- `--atomic-commit`. Dieses Kommando lädt die Regeln aus der angegebenen Datei. Dies ist ein atomarer Vorgang. Die Regeln werden alle gleichzeitig geladen! Die Datei kann mit dem Kommando `--atomic-save` oder `--atomic-init` erzeugt werden.

Sie können für die Verwendung mit dem Etables-Befehl zunächst alle Regeln in einer Datei aufbauen und dann atomar in einem einzigen Schritt aktivieren. So stellen Sie sicher, dass kein Paket ungefiltert Ihre Bridge passiert. Diese Gefahr besteht, wenn Sie zunächst die Tabelle löschen und dann inkrementell die Regeln aufbauen. Das folgende Listing zeigt beispielhaft, wie Sie die Regeln aufbauen können:

```
$EBTABLES --atomic-file filter_table -t filter --atomic-init
$EBTABLES --atomic-file filter_table -A FORWARD -s 00:11:22:33:44:55 -p IPV4 -j ↓
ACCEPT
# ... weitere Befehle

# Lade die Tabelle
$EBTABLES --atomic-file filter_table -t filter --atomic-commit
```

Wenn Sie nicht in jeder Zeile die Datei angeben möchten, können Sie auch die Umgebungsvariable `EBTABLES_ATOMIC_FILE` verwenden.

Um die Rahmen in den Regeln zu prüfen, verfügt Etables über eingebaute Tests (Matches) und über Erweiterungen (Match-Extensions und Watcher-Extensions). Diese werden im Folgenden kurz erläutert.

Zunächst können Sie das Protokoll des Rahmens mit der Option `-p`, `--protocol` prüfen. Hierbei handelt es sich nicht um ein Protokoll wie TCP oder UDP. Denken Sie daran, dass Sie hier einen Ethernet-Frame betrachten. Mögliche Protokolle sind IPv4, ARP und NetBEUI. Eine Liste aller möglichen Protokolle finden Sie in der Datei `/etc/ethertypes`. Das Protokoll kann sowohl mit seinem Namen als auch hexadezimal (IPv4 = 0x0800) angegeben werden. Einige Protokolle (802.2) verwenden dieses Feld als Längenangabe. Sobald der Wert in diesem Feld kleiner als 0x600 ist, handelt es sich um eine Länge. Dann verwenden Sie hier bitte `LENGTH`.

Natürlich kann Etables auch die Netzwerkkarten prüfen, über die das Paket die Bridge erreicht hat. Hier wird jedoch zwischen dem physikalischen Interface (z.B. `eth0`) und dem logischen Interface (z.B. `br0`) unterschieden. Die eingehenden Interfaces können Sie in den Ketten `INPUT`, `FORWARD`, `PREROUTING` und `BROUTING` testen, während Sie die ausgehenden Interfaces in den Ketten `OUTPUT`, `FORWARD` und `POSTROUTING` testen können. Die Optionen heißen:

3.1.4 Die ebtables-Syntax

- `-i, --in-if, --in-interface`
- `--logical-in`
- `-o, --out-if, --out-interface`
- `--logical-out`

Um die MAC-Adressen des Rahmens zu prüfen, gibt es die Optionen `-s, --src, --source` beziehungsweise `-d, --dst, --destination`. Die Adresse wird als hexadezimale Adresse mit Doppelpunkten geschrieben: `00:50:56:C0:00:03`. Beide Optionen erlauben auch die Angabe einer Netzmaske. Alternativ können Sie als Absenderadresse auch `Unicast, Broadcast, Multicast` oder `BGA (Bridge Group Address)` angeben. Um diese Typen als Zieladresse zu prüfen, existiert die Option `--pkt-type` (siehe unten).

Der Ebtables-Befehl verfügt über eine Reihe von Erweiterungen, die für die Prüfung von Rahmen genutzt werden können. Diese Erweiterungen müssen nicht wie bei `iptables` mit der Option `-m` geladen werden. Sie werden jedoch in eigenen Kernel-Modulen implementiert.

Für die Prüfung von 802.2- oder 802.3-Rahmen stehen die Optionen `--802_3-sap` und `--802_3-type` zur Verfügung.

Für das ARP- und das RARP-Protokoll können Sie die folgenden Optionen verwenden. Diese Optionen sind nur gültig, wenn Sie auch als Protokoll (`-p`) ARP oder RARP spezifiziert haben.

- `--arp-opcode`. Diese Option gibt den ARP-Operation-Code an. Insgesamt existieren neun verschiedene Opcodes:
 - Request (1)
 - Reply (2)
 - Request_Reverse (3)
 - Reply_Reverse (4)
 - DRARP_Request (5)
 - DRARP_Reply (6)
 - DRARP_Error (7)
 - InARP_Request (8)
 - ARP_NAK (9)

Diese Opcodes können als Zeichenkette oder numerisch angegeben werden.

- `--arp-htype`. Der Hardware-Typ. Eigentlich immer Ethernet (1).
- `--arp-ptype`. Der Protokoll-Typ. Eigentlich immer IPv4 (0x0800).
- `--arp-ip-src, --arp-ip-dst`. Die IP-Absender- beziehungsweise IP-Zieladresse des ARP-Pakets.
- `--arp-mac-src, --arp-mac-dst`. Die MAC-Absender- beziehungsweise MAC-Zieladresse des ARP-Pakets.

Wenn Sie als Protokoll (-p) IPv4 in der Regel spezifizieren, dürfen Sie die folgenden Optionen nutzen, um rudimentär das IP-Paket zu prüfen. Diese Optionen erreichen bei weitem nicht die Mächtigkeit des `iptables`-Kommandos.

So können Sie mit den Optionen `--ip-source`, `--ip-src` und `--ip-destination`, `--ip-dst` die Absender- und Ziel-IP-Adresse prüfen. Mit `--ip-tos` kann durch Angabe der hexadezimalen Nummer der Type-of-Service geprüft werden. Die Option `--ip-protocol`, `--ip-proto` erlaubt die Prüfung des IP-Protokolls (z.B. TCP oder UDP). Handelt es sich um ein TCP- oder UDP-Paket, können Sie mit `--ip-source-port`, `--ip-sport` und `--ip-destination-port`, `--ip-dport` die Ports testen.

Ebttables unterstützt genauso wie Iptables die Markierung von Paketen. Mit der Option `--mark` können Sie diese Markierung testen.

Die Option `--pkt-type` gibt Ihnen die Möglichkeit, den Pakettyp zu testen. Mögliche Typen sind `broadcast`, `multicast`, `host` und `otherhost`. Ist das Paket an den lokalen Rechner gerichtet, so ist der Typ `host`. Alle anderen Unicast-Pakete sind `otherhost`.

Für die Prüfung von Spanning-Tree-Protokoll-Rahmen (STP) gibt es eine Vielzahl von Optionen, die hier nur verwirren würden. Auf der Manpage finden Sie weitere Informationen. Auch die VLAN-Testoptionen sollen hier nicht besprochen werden.

Um Pakete zu protokollieren, existiert die Option `--log`. Die Protokollierung ist bei Ebttables im Gegensatz zu Iptables eine Option und kein Ziel (Target). Die Option `--log` protokolliert das Paket mit den Defaultwerten für den Level, das Präfix und ohne IP- und ARP-Informationen. Wenn Sie diese Defaultwerte ändern möchten, nutzen Sie anstelle der Option `--log` die Optionen `--log-prefix`, `--log-level`, `--log-ip` und `--log-arp`.

Sobald eine Regel auf einen Rahmen zutrifft, kann Ebttables verschiedene Aktionen ausführen. Zusätzlich zu den von Iptables bekannten Aktionen `ACCEPT`, `DROP` und `RETURN` besitzt Ebttables auch noch `CONTINUE` und Erweiterungen (Target Extensions). Wurde `CONTINUE` als Ziel (Target) einer Regel angegeben, so wird bei Zutreffen dieser Regel die Kette nicht verlassen, sondern die weiteren Regeln der Kette werden abgearbeitet.

Als Erweiterungen stehen `arpreply`, `dnat`, `mark`, `redirect` und `snat` zur Verfügung. Achtung, diese Ziele werden entgegen der Konvention kleingeschrieben! Bei allen diesen Zielen müssen Sie zusätzlich noch angeben, was anschließend mit dem Paket passieren soll. Hierfür gibt es bei jedem Ziel noch eine zusätzliche Option, die weiter unten erläutert wird.

- `-j arpreply`. Das `arpreply`-Ziel erlaubt es, ARP-Anfragen zu beantworten. Dieses Ziel dürfen Sie nur in der `PREROUTING`-Kette der `nat`-Tabelle verwenden. Die Regel erkennt selbstständig, ob es sich um ARP-Anfragen handelt. Mit der Option `--arpreply-mac` können Sie die MAC-Adresse angeben, die in der Antwort verwendet werden soll. Mit der Option `--arpreply-target` geben Sie an, wie Ebttables anschließend den originalen Rahmen behandeln soll. Die Default-Einstellung ist `DROP`. Dies hat den Nachteil, dass der lokale ARP-Cache nicht mit den Absenderinformationen aktualisiert wird. Wenn Sie das Target auf `ACCEPT` oder `CONTINUE` setzen, ist dies dennoch der Fall. Sie müssen dann nur aufpassen, dass nicht ein weiterer ARP-Reply ausgesendet wird!

- `-j dnat`. Dieses Ziel darf in der `BROUTING`-Kette und in den `PREROUTING`- und `OUTPUT`-Ketten der `nat`-Tabelle genutzt werden. Die Ziel-MAC-Adresse geben Sie mit `--to-destination`, `--to-dst` an. Anschließend wird das Paket akzeptiert. Wenn Sie dies ändern möchten, können Sie mit `--dnat-target` ein anderes Ziel angeben.
- `-j mark`. Ebttables unterstützt wie Iptables eine Markierung der Rahmen/Pakete. Die Markierung geben Sie mit `--set-mark` als hexadezimale unsignierte Zahl an. Anschließend wird das Paket akzeptiert. Wenn Sie weitere Modifikationen des Pakets in der gleichen Kette vornehmen wollen, können Sie das mit `--mark-target CONTINUE` erreichen.
- `-j redirect`. Dieses Ziel ist analog dem `REDIRECT`-Ziel von Iptables und erlaubt es, Pakete in der `BROUTING`- und `PREROUTING`-Kette, die eigentlich an eine andere MAC-Adresse gerichtet sind und gebridgt werden müssten, auf die lokale MAC-Adresse umzuleiten. Dabei wird als neue Ziel-MAC-Adresse die Adresse der Netzwerkkarte genutzt, über die der Rahmen das System erreicht hat. Anschließend wird das Paket akzeptiert. Das Verhalten können Sie mit `--redirect-target` ändern.

Mit dieser Option können Sie auf einer Bridge Verbindungen über einen transparenten Proxy lenken!

- `-j snat`. Hiermit können Sie in der `POSTROUTING`-Kette die Absender-MAC-Adresse ändern. Die neue MAC-Adresse geben Sie mit `--to-source`, `--to-src` an. Mit `--snat-target` können Sie ein anderes Ziel als `ACCEPT` (Default) angeben.

31.5 Start einer Bridge auf Fedora Core

Die Linux-Distribution Fedora Core enthält bereits in ihren Startskripten die Möglichkeit, eine Bridge zu initialisieren. Hierzu erzeugen Sie die folgenden Dateien in dem Verzeichnis `/etc/sysconfig/network-scripts`:

- `ifcfg-br0`

```

DEVICE=br0
ONBOOT=yes
BOOTPROTO=static
IPADDR=10.1.2.3
NETMASK=255.255.255.0
TYPE=Bridge
STP=off
DELAY=0
GCINT=30

```
- `ifcfg-eth0`

```

DEVICE=eth0
ONBOOT=yes
BOOTPROTO=static
BRIDGE=br0

```

- ifcfg-eth1
DEVICE=eth1
ONBOOT=yes
BOOTPROTO=static
BRIDGE=br0

- Für jede weitere Netzwerkkarte der Bridge gibt es eine entsprechende Datei.

Fedora Core wird nun beim Booten die entsprechende Bridge initialisieren und die physikalischen Netzwerkkarten hinzufügen. Mit dem Parameter `STP` können Sie das STP-Protokoll an- beziehungsweise abschalten.