

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



30 Iptables auf einer Bridge

Eine Bridge ist eine Software, die ähnliche Funktionen übernimmt wie ein Switch, der in Hardware implementiert wurde. Die Bridge verfügt häufig über zusätzliche Möglichkeiten, wie zum Beispiel die Änderung des Mediums. So gibt es Bridges, die Token-Ring- und Ethernet-Netze miteinander verbinden können. Linux kann derartige Bridges betreiben und auch mit Iptables die von der Bridge weitergeleiteten Pakete filtern.

30.1 Wie funktioniert die Bridge?

Wie funktioniert nun eine Bridge? Zunächst definieren Sie die Netzwerkkarten, die Teil der Bridge werden sollen. Nach Aktivierung der Bridge lernt die Bridge ähnlich wie ein Switch die MAC-Adressen der Rechner, die an der Bridge angeschlossen sind. Dazu betrachtet die Bridge bei jedem Paket die Absender-MAC-Adresse und speichert diese in einer Tabelle ab (siehe Abbildung 30.1).

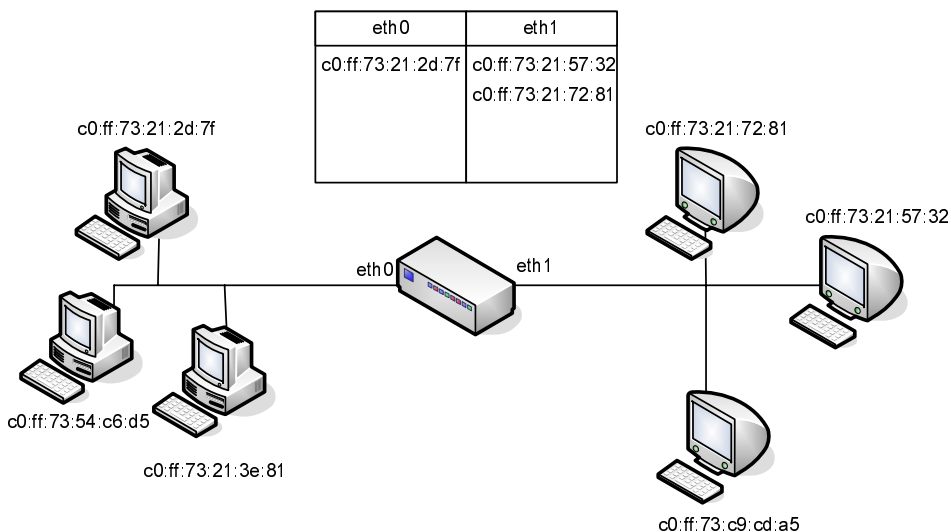


Abbildung 30.1: Eine Bridge lernt selbstständig die MAC-Adressen der angeschlossenen Geräte.

Solange die Bridge die Ziel-MAC-Adresse eines Pakets noch nicht kennt, kopiert die Bridge dieses Paket in alle angeschlossenen Netze, so dass auf jeden Fall eine Kommunikation möglich ist. Sobald aber die Bridge die Ziel-MAC-Adresse bereits kennt, ermittelt sie mit ihrer Tabelle die Netzwerkkarte, an der der entsprechende Rechner angeschlossen ist, und leitet das Paket auf dieser Netzwerkkarte weiter. So unterdrückt die Bridge Broadcasts und vermeidet Kollisionen in dem Netzwerk durch gleichzeitig gesendete Pakete. Damit erhöht die Bridge die Leistung des Ethernet-Netzwerks, das bei einer Kollision die Pakete erneut senden muss.

30.2 Bau einer Bridge mit Linux

Um mit einem Linux-System eine Bridge zu bauen, benötigen Sie lediglich einen Kernel 2.4 oder 2.6. Für die älteren Kernel benötigen Sie zusätzlich den Bridge-Patch von <http://bridge.sourceforge.net>. Für die Konfiguration der Bridge müssen Sie die Bridge-Utilities installieren. Diese sind in den meisten Distributionen enthalten. Das Paket heißt üblicherweise `bridge-utils`. Die aktuelle Version ist die Version 1.0.6 und kann ebenfalls von der oben angegebenen URL heruntergeladen werden. Sie benötigen die aktuellen Versionen nur für die neuesten Kernel. Meist genügen auch die älteren Versionen der Distributionen.

Der wesentliche Befehl der Bridge-Utilities ist `brctl`. Hiermit steuern Sie die gesamte Bridge:

```
[root@bibo ~]# brctl
commands:
    addbr          <bridge>          add bridge
    delbr          <bridge>          delete bridge
    addif          <bridge> <device>      add interface to bridge
    delif          <bridge> <device>      delete interface from bridge
    setageing      <bridge> <time>        set ageing time
    setbridgeprio  <bridge> <prio>        set bridge priority
    setfd          <bridge> <time>        set bridge forward delay
    sethello       <bridge> <time>        set hello time
    setmaxage      <bridge> <time>        set max message age
    setpathcost    <bridge> <port> <cost>    set path cost
    setportprio    <bridge> <port> <prio>    set port priority
    show           <bridge>          show a list of bridges
    showmacs       <bridge>          show a list of mac adrs
    showstp        <bridge>          show bridge stp info
    stp            <bridge> <state>    turn stp on/off
```

Um nun eine Bridge zu erzeugen, verwenden Sie den Befehl `brctl addbr br0`. Das Gerät `br0` ist anschließend sofort verfügbar:

```
[root@bibo ~]# ip link show br0
4: br0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
```

Nun können Sie der Bridge Netzwerkkarten hinzufügen. Hierzu verwenden Sie den Befehl `brctl addif`. Dabei ist es wichtig, dass diese Netzwerkkarten keine IP-Adressen tragen. Dann können Sie mit dem `ip`-Befehl die Netzwerkkarten aktivieren.

```
[root@bibo ~]# brctl addif br0 eth0
[root@bibo ~]# brctl addif br0 eth1
[root@bibo ~]# ip link set eth0 up
[root@bibo ~]# ip link set eth1 up
[root@bibo ~]# ip link set br0 up
[root@bibo ~]# ip link show
1: lo: <LOOPBACK,UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: eth0: <BROADCAST,MULTICAST,PROMISC,UP> mtu 1500 qdisc pfifo_fast
    qlen 100
    link/ether 00:20:e0:6c:72:1e brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST,PROMISC,UP> mtu 1500 qdisc pfifo_fast
    qlen 100
    link/ether 00:10:a4:c3:26:cb brd ff:ff:ff:ff:ff:ff
4: br0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue
    link/ether 00:10:a4:c3:26:cb brd ff:ff:ff:ff:ff:ff
```

Wenn bei Ihnen der `ip link show`-Befehl nicht das `PROMISC`-Flag bei den Netzwerkkarten anzeigt, arbeiten Sie wahrscheinlich auf einem Linux-Kernel 2.6. Hier wird dieses Flag nicht mehr angezeigt.

Mit dem Befehl `brctl` können Sie sich die Einzelheiten der Bridge anzeigen lassen:

```
[root@bibo ~]# brctl show
bridge name      bridge id          STP enabled      interfaces
br0              8000.0010a4c326cb yes                eth0
                                     eth1
```

Wenn Sie die Bridge für Firewall-Zwecke einsetzen möchten, sollten Sie das Spanning-Tree-Protokoll (STP) abschalten. Dies ermöglicht den Aufbau von redundanten Bridges. Da das Protokoll aber keine Sicherheit bietet, sollte es im Zusammenhang mit Firewalls nicht genutzt werden: `brctl stp br0 off`. Dann können Sie auch die Forward-Delay Zeit auf null setzen. Bei dem Einsatz des Spanning-Tree-Protokolls werden hiermit Schleifen zum Startzeitpunkt der Bridge ausgeschlossen:

```
brctl fd br0 0.
```

Nun sollte bereits eine Kommunikation über die Bridge möglich sein. Dazu können Sie bereits durch die Bridge pingen. Anschließend können Sie sich die MAC-Adressen, die von der Bridge bereits gelernt wurden, anzeigen lassen:

```
[root@bibo ~]# brctl showmacs br0
port no mac addr          is local?      ageing timer
1      00:20:e0:6c:72:1e      yes            0.00
2      00:20:e0:6c:72:10      yes            0.00
```

1	00:20:e0:73:71:20	yes	0.00
2	00:20:e0:73:71:21	yes	0.00

30.3 Filtern auf der Bridge mit iptables

Nun können Sie wie üblich auf der Bridge filtern. Die Pakete durchlaufen nacheinander die Mangle-PREROUTING-, die NAT-PREROUTING-, die Mangle-FORWARD-, die Filter-FORWARD-, die Mangle-POSTROUTING- und die NAT-POSTROUTING-Kette.



Achtung



Ich habe bei mir auf einigen Kernen feststellen müssen, dass die Pakete die Mangle-PREROUTING-Kette teilweise dreimal durchlaufen. Dies scheint ein Artefakt zu sein.

Wenn Sie nun die Pakete filtern möchten, können Sie das ganz normal in der FORWARD-Kette tun. Allerdings sollten Sie darauf achten, dass die Bridge, während sie die MAC-Adressen lernt, auch noch Pakete weiterleitet, die nicht weitergeleitet werden müssen. Das bedeutet, dass Sie mit REJECT-Regeln sehr vorsichtig sein sollten. In der Abbildung 30.2 ist ein Szenario gezeigt, in dem eine REJECT-Regel zum Abbruch gültiger Verbindungen führt. Hier baut der Rechner A zum Rechner B in demselben Netzwerk eine Verbindung auf. Die Bridge sieht ebenfalls das Paket und muss es weiterleiten, da sie die Ziel-MAC-Adresse noch nicht kennt. Es durchläuft die FORWARD-Kette und wird abgelehnt. Da die Ablehnung durch eine REJECT-Regel erfolgt, sendet die Bridge einen Fehler an Rechner A, obwohl die Verbindung sehr wohl aufgebaut werden darf.

Bei der Definition der Regeln ist es häufig sinnvoll zu prüfen, über welche Netzwerkkarte der Bridge das Paket die Bridge erreicht. Hierfür verwenden Sie beim Linux-Kernel 2.4 wie gewohnt die Optionen `-i` bzw. `--in-interface` und `-o` bzw. `--out-interface`. Bei dem Linux-Kernel 2.6 wurden hierfür neue Optionen geschaffen, da insgesamt die Firewall-Möglichkeiten auf der Bridge erweitert wurden (siehe das nächste Kapitel 31). Diese Optionen heißen `--physdev-in` und `--physdev-out`. Um diese Optionen zu nutzen, müssen Sie die Erweiterung `physdev` in Ihren Regeln laden. Ein Beispiel zeigt die Anwendung:

```
$IPTABLES -A FORWARD -m physdev --physdev-in eth0 --physdev-out eth1 -d 192.168.0.1 -j ACCEPT
```

Die weiteren Optionen (`--physdev-is-in`, `--physdev-is-out` und `--physdev-is-bridged`) der `physdev`-Erweiterung werden nur selten benötigt und erklären sich von selbst.

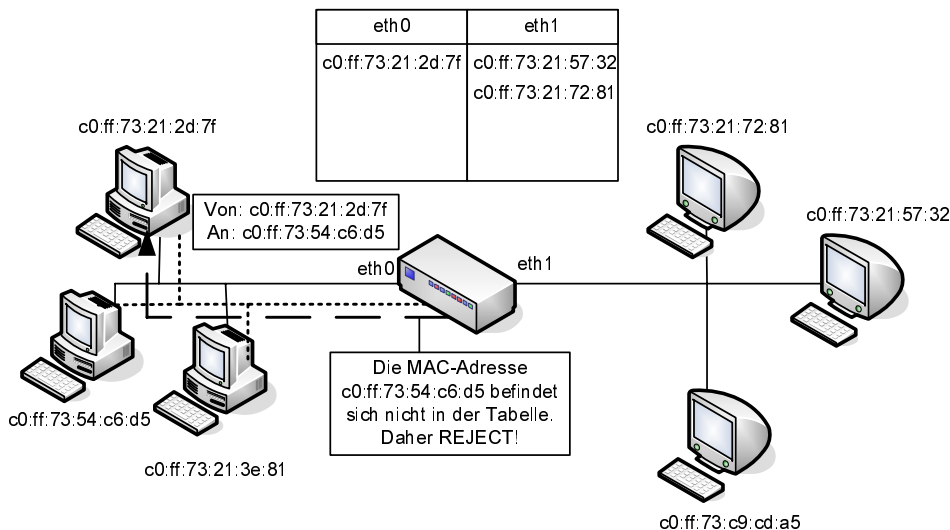


Abbildung 30.2: Ein REJECT kann auf einer Bridge Probleme erzeugen.

Wenn Sie mit einer Bridge ein Netzwerk in zwei Kollisionsdomänen aufgeteilt haben und nun auf der Bridge filtern möchten, ist häufig die `ipset`-Erweiterung des Linux-Kernels hilfreich. Dies ist eine Weiterentwicklung des Befehls `ipool`. Damit ist es sehr leicht möglich, mehrere nicht zusammenhängende IP-Adressen in einer Regel zusammenzufassen. Dies reduziert den Regelaufwand und erlaubt das Erzeugen übersichtlicher Regelwerke. Sie finden die Informationen über diese Erweiterungen in Kapitel 25.

30.4 Filtern auf der Bridge mit arptables

Der Befehl `arptables` ist in den meisten modernen Distributionen enthalten. Er erlaubt die Filterung der Address-Resolution-Protocol-Pakete (ARP), die das IPv4-Protokoll benötigt, um ausgehend von einer IP-Adresse die MAC-Adresse eines Rechners zu ermitteln.

Eine Filterung der ARP-Pakete auf einem Router oder einem einzelnen Rechner ist ungewöhnlich, da dies nur dazu führt, dass ein Rechner scheinbar unsichtbar in dem IPv4-Netzwerk wird.

Wesentlich interessanter ist die Anwendung auf einer Bridge, die im Gegensatz zu einem Router ARP-Pakete weiterleitet. Hier kann durch eine sinnvolle Filterung der ARP-Pakete erreicht werden, dass die von der Firewall-Bridge geschützten Systeme auch auf der Ebene des ARP-Protokolls nur für bestimmte Clients erreichbar sind.

Der `arptables`-Befehl kann auf einem Linux-Kernel 2.4 zwei Ketten (INPUT und OUTPUT) und auf einem Linux-Kernel 2.6 drei Ketten (zusätzlich FORWARD) in der Filter-Tabelle verwalten. Die FORWARD-Kette ist auf einem Linux-Kernel 2.4 nur nach Anwendung

des Bridge-nf-Patches verfügbar. Diese Ketten verwalten lediglich die ARP-Pakete. Die INPUT-Filter-Kette betrachtet alle ARP-Pakete, die an den Rechner gerichtet sind. Die OUTPUT-Filter-Kette betrachtet alle lokal erzeugten ARP-Pakete, während die FORWARD-Filter-Kette auf einer Bridge alle von der Bridge durchgeleiteten ARP-Pakete betrachtet.

So ist es möglich, dass nur bestimmte Clients eine ARP-Anfrage für eine bestimmte IP-Adresse senden dürfen. Mit `arptables` können Sie das mit dem folgenden Befehl erreichen:

```
ARPTABLES=/usr/sbin/arptables
$ARPTABLES -A FORWARD -s 192.168.0.7 -d 192.168.0.25 -j ACCEPT
$ARPTABLES -A FORWARD -d 192.168.0.25 -j DROP
```

30.5 Fazit

Dieser Abschnitt hat Ihnen gezeigt, wie Sie sehr einfach mit `iptables` auf einer Bridge die Pakete filtern können. So können Sie sehr einfach auf der IP-Ebene transparente Paketfilter aufbauen. Jedoch möchten Sie vielleicht eine noch speziellere Funktion implementieren. Es besteht die Möglichkeit, einige Pakete (zum Beispiel alle IP-Pakete) zu routen und andere Pakete (zum Beispiel NETBEUI-Pakete) über die Bridge zu senden und dort zu filtern. Hierzu benötigen Sie den Befehl `ebtables`, und die entsprechenden Funktionen müssen in Ihrem Kernel aktiviert worden sein. Das nächste Kapitel erklärt die Funktion.