

Netzwerk Teil 2

Linux-Kurs der Unix-AG

Zinching Dang

7. Dezember 2015



UNIX
AG
TU Kaiserslautern

RH Regionales
Hochschul-
Rechenzentrum
Kaiserslautern **RK**

Unterschied Host – Router

- ▶ Standardverhalten eines Linux-Rechners: Host
- ▶ nur IP-Pakete mit Zieladressen, die dem Rechner zugeordnet sind, werden angenommen
- ▶ IP-Adresse muss auf einem Interface konfiguriert sein
- ▶ Pakete für andere IP-Adressen werden verworfen
- ▶ über Sysctl kann Forwarding aktiviert werden
- ▶ Rechner verhält sich dann wie ein Router
- ▶ leitet Pakete für fremde IP-Adressen anhand der Routing-Tabelle weiter

Forwarding per Sysctl aktivieren

- ▶ IPv4:
echo 1 > /proc/sys/net/ipv4/conf/all/forwarding
- ▶ IPv6:
echo 1 > /proc/sys/net/ipv6/conf/all/forwarding
- ▶ oder dauerhaft in /etc/sysctl.conf:

```
1 net.ipv4.conf.all.forwarding=1  
2 net.ipv6.conf.all.forwarding=1
```

- ▶ Nebenwirkungen: deaktiviert u. a. stateless autoconfig für IPv6

iproute2

- ▶ Ersatz für u. a. `ifconfig`, `route`, `arp`
- ▶ kann den Status von Netzwerk-Interfaces anzeigen und ändern
- ▶ IP-Adressen hinzufügen und löschen
- ▶ Routen anzeigen und ändern
- ▶ ARP-Tabelle anzeigen und ändern
- ▶ erweiterte Funktionalität im Vergleich zu den Vorgänger-Werkzeugen
- ▶ z. B. mehrere IP-Adressen auf einem Interface

ip link

- ▶ Link-Status anzeigen: `ip link show dev eth0`
- ▶ von allen Interfaces: `ip link show`
- ▶ Interface einschalten: `ip link set eth0 up`
- ▶ ausschalten: `ip link set eth0 down`

ip address

- ▶ IP-Adresse hinzufügen:
`ip address add 192.0.2.4/28 dev eth0`
- ▶ löschen: `ip address del 192.0.2.4/29 dev eth0`
- ▶ anzeigen: `ip address show [dev eth0]`
- ▶ auch mit IPv6:
`ip addr add 2001:db8:f00d::3/64 dev eth0`

ip route

- ▶ Route hinzufügen:
`ip route add 198.51.100.0/24 via 192.0.2.1 dev eth0`
- ▶ löschen analog mit `del`
- ▶ Routing-Tabelle anzeigen: `ip route show`
- ▶ für IPv6: `ip -6 route show`

ip neighbour

- ▶ Neighbour-Tabelle anzeigen: `ip neighbour show`
- ▶ für IPv4 auch als ARP-Tabelle bekannt

Lab 4.1: Routing

- ▶ Forwarding aktivieren
- ▶ dem Rechner eine weitere IP-Adresse hinzufügen
- ▶ Routen zu den zusätzlichen IP-Adressen auf den anderen Rechnern hinzufügen
- ▶ VLAN 7 deaktivieren und über Partner-Rechner erreichen

iptables

- ▶ Paketfilter für IP-Pakete unter Linux
- ▶ ip6tables für IPv6
- ▶ kann eingehende, ausgehende und weitergeleitete Pakete filtern
- ▶ einsetzbar als Firewall auf Rechnern und auf Routern
- ▶ kann auch als NAT-Gateway verwendet werden
- ▶ in Linux 3.13 durch nftables ersetzt
- ▶ nftables in vielen aktuellen Distributionen noch nicht enthalten
- ▶ iptables über Kompatibilitätslayer weiter unterstützt

Tabellen und Chains

- ▶ jedes Paket durchläuft verschiedene Tabellen, die wiederum Chains enthalten
- ▶ Chains enthalten Filter-Regeln, die nacheinander ausgewertet werden
- ▶ die erste zutreffende Regel wird angewandt
- ▶ wichtigste Tabellen:
 - ▶ filter: für „normales“ Filtern
 - ▶ nat: für NAT

Tabellen

- ▶ Tabelle filter aufgeteilt in die Chains
 - ▶ INPUT: Pakete für den lokalen Rechner
 - ▶ FORWARD: Pakete, die vom Rechner weitergeleitet werden (Router)
 - ▶ OUTPUT: Pakete, die vom lokalen Rechner erzeugt wurden
- ▶ Tabelle nat aufgeteilt in die Chains
 - ▶ PREROUTING: Pakete manipulieren, bevor entschieden wird, wie sie weitergeleitet werden
 - ▶ OUTPUT: Pakete, die vom lokalen Rechner erzeugt wurden, manipulieren
 - ▶ POSTROUTING: Pakete manipulieren, nachdem entschieden wurde, wie sie weitergeleitet werden

Filterregeln einfügen

- ▶ `iptables -t <tabelle> -A <chain> ... <Kriterien>`
`... -j <aktion>`
- ▶ fügt eine Regel ans Ende des angegebenen Chains an (-A: append)
- ▶ löschen mit -D statt -A
- ▶ auch möglich: -I (insert), -R (replace)
- ▶ -t filter ist Standard
- ▶ wichtigste Aktionen:
 - ▶ ACCEPT: das Paket akzeptieren
 - ▶ DROP: das Paket verwerfen
 - ▶ REJECT: das Paket verwerfen und eine Fehlermeldung an den Absender senden

Filterkriterien

- ▶ `-s <adresse>`: nach Quell-IP filtern
- ▶ `-d <adresse>`: nach Ziel-IP filtern
- ▶ `-p tcp --sport <port>`: nach TCP-Quell-Port filtern
- ▶ `-p tcp --dport <port>`: nach TCP-Ziel-Port filtern
- ▶ `-p udp --sport <port>`: nach UDP-Quell-Port filtern
- ▶ `-p udp --dport <port>`: nach UDP-Ziel-Port filtern
- ▶ `-m state --state <state>`: nach Zustand der Verbindung filtern
- ▶ wichtigste Zustände: NEW, INVALID, ESTABLISHED, RELATED
- ▶ Kriterien können kombiniert werden
- ▶ ! vor dem Kriterium kehrt die Bedeutung um

Beispiel

```
1 iptables -A INPUT -s 192.0.2.0/27 -p tcp --dport 443
   -j ACCEPT
2 iptables -A INPUT -p tcp --dport 22 -j ACCEPT
3 iptables -A INPUT ! -s 203.0.113.0/24 -p tcp --dport
   25 -j ACCEPT
4 iptables -A INPUT -p tcp -m state ! --state
   ESTABLISHED,RELATED -j REJECT
```

- ▶ Zugriff von 192.0.2.0/27 auf Port 443 erlauben
- ▶ Zugriff von überall auf Port 22 erlauben
- ▶ Zugriff von überall außer 203.0.113.0/24 auf Port 25 erlauben
- ▶ Zugriff auf alle anderen TCP-Ports verbieten (außer bestehende Verbindungen)

NAT

- ▶ Network Address Translation
- ▶ allgemein: Verändern von Adressen in IP-Paketen
- ▶ häufigste Anwendung: mehrere private IPv4-Adressen auf eine öffentliche umschreiben
- ▶ auch Masquerading genannt
- ▶ bei vielen Internetzugängen nötig, da für viele Rechner nur eine öffentliche IP-Adresse bereit steht
- ▶ andere häufige Anwendung: Port-Forwarding eines Ports auf dem Gateway zu einer internen IP-Adresse

NAT mit iptables

- ▶ NAT-Regeln in der Tabelle nat, Chain POSTROUTING
- ▶ MASQUERADE-Aktion sorgt dafür, dass die Quell-IP durch die Adresse des externen Interfaces des Gateways ersetzt wird

```
1 iptables -t nat -A POSTROUTING -o eth0 -s  
192.168.254.0/24 -j MASQUERADE
```

- ▶ bei fester externer Adresse SNAT statt MASQUERADE
- ▶ Verbindungen werden nicht unterbrochen, wenn das Interface aus- und wieder eingeschaltet wird

```
1 iptables -t nat -A POSTROUTING -o bond0 -s  
192.168.2.0/24 -j SNAT --to-source 203.0.113.7
```

Port-Weiterleitungen

- ▶ werden verwendet um Dienste auf internen IP-Adressen von außen zugänglich zu machen
- ▶ Port auf der externen Adresse wird auf eine interne Adresse umgeleitet
- ▶ mit DNAT im Chain PREROUTING

```
1 iptables -t nat -A PREROUTING -p tcp --dport 22200  
-j DNAT --to-destination 192.168.2.100:22
```

- ▶ TCP-Port 22200 von außen wird auf Port 22 (SSH) der internen IP 192.168.2.100 umgeleitet

- ▶ Anzeigen der Regeln mit `iptables -t <tabelle> -L`
- ▶ mehr Informationen mit `-v`
- ▶ Löschen aller Regeln mit `iptables -t <tabelle> -F`
- ▶ Zurücksetzen der Paketzähler: `iptables -t <tabelle> -Z`
- ▶ Regeln (und Zähler) werden beim Herunterfahren automatisch gelöscht
- ▶ müssen beim Hochfahren neu definiert werden

iptables-persistent

- ▶ iptables-persistent stellt ein Init-Skript bereit, das gespeicherte Regeln beim Hochfahren automatisch lädt
- ▶ Speichern der Regeln mit
`/etc/init.d/iptables-persistent save`
- ▶ Regeln werden im Verzeichnis `/etc/iptables/` gespeichert
- ▶ einfaches Löschen aller geladenen Regeln:
`/etc/init.d/iptables-persistent flush`
- ▶ Laden mit `start`

Lab 4.2: iptables

- ▶ Zugriff auf SSH einschränken
- ▶ Zugriff auf VLAN 7 von anderen Rechnern verbieten
- ▶ iptables-Regeln beim Hochfahren wiederherstellen