Netzwerk-Zugriffskontrolle mit einer DMZ

Michael Dienert

22. März 2016

Inhaltsverzeichnis

1 De-Militarized Zone

Der Begriff *De-Militarisierte Zone* bezeichnet einen befriedeten Bereich, der zwei feindliche Lager voneinander trennt.

In der IT-Technik wird dieses Konzept auf zwei Arten realisiert. Es gibt eine zweistufige Anordnung, die zwei Router verwendet (Abb. 1) und eine etwas einfachere, einstufige Variante (Abb. 2), bei der ein Router mit drei Interfaces verwendet wird.

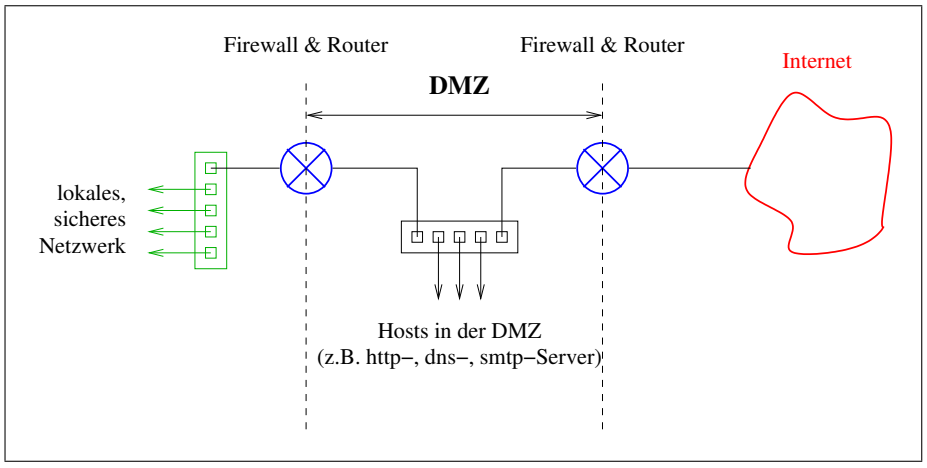

Abbildung 1: Zweistufige DMZ

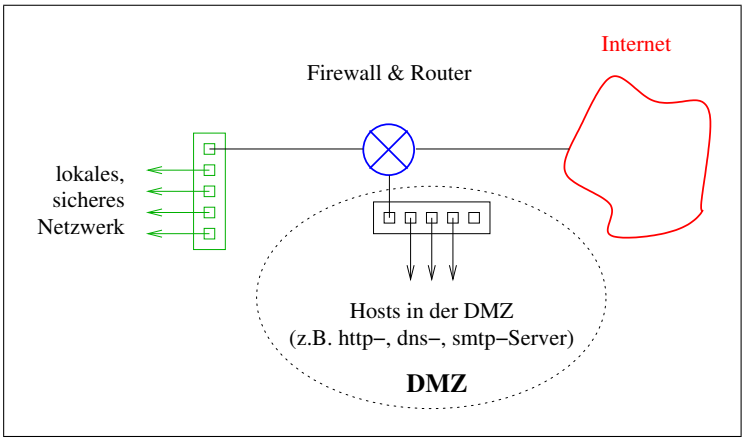

Abbildung 2: Einstufige DMZ

In den Abbildungen sind die Firewalls (Paketfilter) und die Router als ein Gerät dargestellt. Werden höchste Sicherheitsstandards verlangt, sollte man dafür zwei unabhängi-

ge Geräte einsetzen.

2 Paketfilterung mit dem Linux-Kernel

2.1 Kurzer Ausflug in die IP-Tables

- Die Firewall besteht aus **Tabellen**.
- Eine Tabelle enthält mehrere Filter-**Ketten**.
- Eine Kette besteht aus **Regeln**, die Regeln sind also die Kettenglieder. Die Regeln einer Kette werden nacheinander durchlaufen. Trifft eine Regel zu, wird die Kette verlassen.
- Eine Regel endet mit der Angabe eines Sprung-**Ziels**. Das Ziel bestimmt, was mit dem Paket gemacht wird: DROP, ACCEPT, DNAT, ... oder ob man zu einer anderen Kette springt.

2.1.1 Die Tabellen FILTER, NAT und MANGLE

Es gibt standardmässig die drei Tabellen:

filter ist die Standardtabelle. Ist keine Tabelle angegeben (Option `-t`), wird *filter* verwendet.

nat Die Tabelle für NAT wird mit `-t nat` aufgerufen.

mangle Die Tabelle mangle wird hier ausgespart.

2.1.2 Sprungziele

Sprungziele (targets) bestimmen, wie mit dem Paket verfahren wird. Die Ziele werden mit **-j** oder **–jump** aufgerufen. Es gibt (vordefiniert, Liste nicht vollst.) :

- DROP
- ACCEPT
- MASQUERADE: gibt es nur in der nat-Tabelle
- DNAT: für Port-Forwarding; gibt es nur in der nat-Tabelle

2.1.3 Vordefinierte Filterketten

Es gibt 5 vordefinierte *Ketten* (in Blocksatz):

PREROUTING erste Kette, da muss der gesamte Verkehr durch (gut für z.B. *port forwarding* = *Destination NAT*)

INPUT Kette für Pakete, die *für* den Router selbst bestimmt sind

FORWARD Kette für Pakete, die geroutet werden

OUTPUT Kette für Pakete, die *vom* Router selbst stammen

POSTROUTING letzte Kette, da muss der gesamte Verkehr durch (für *Source NAT*)

2.2 Zusammenfassung

filter
FORWARD
INPUT
OUTPUT

Tabelle 1: Tabelle filter

nat
PREROUTING
OUTPUT
POSTROUTING

Tabelle 2: Tabelle nat

mangle
PREROUTING
POSTROUTING
OUTPUT
INPUT
FORWARD

Tabelle 3: Tabelle mangle

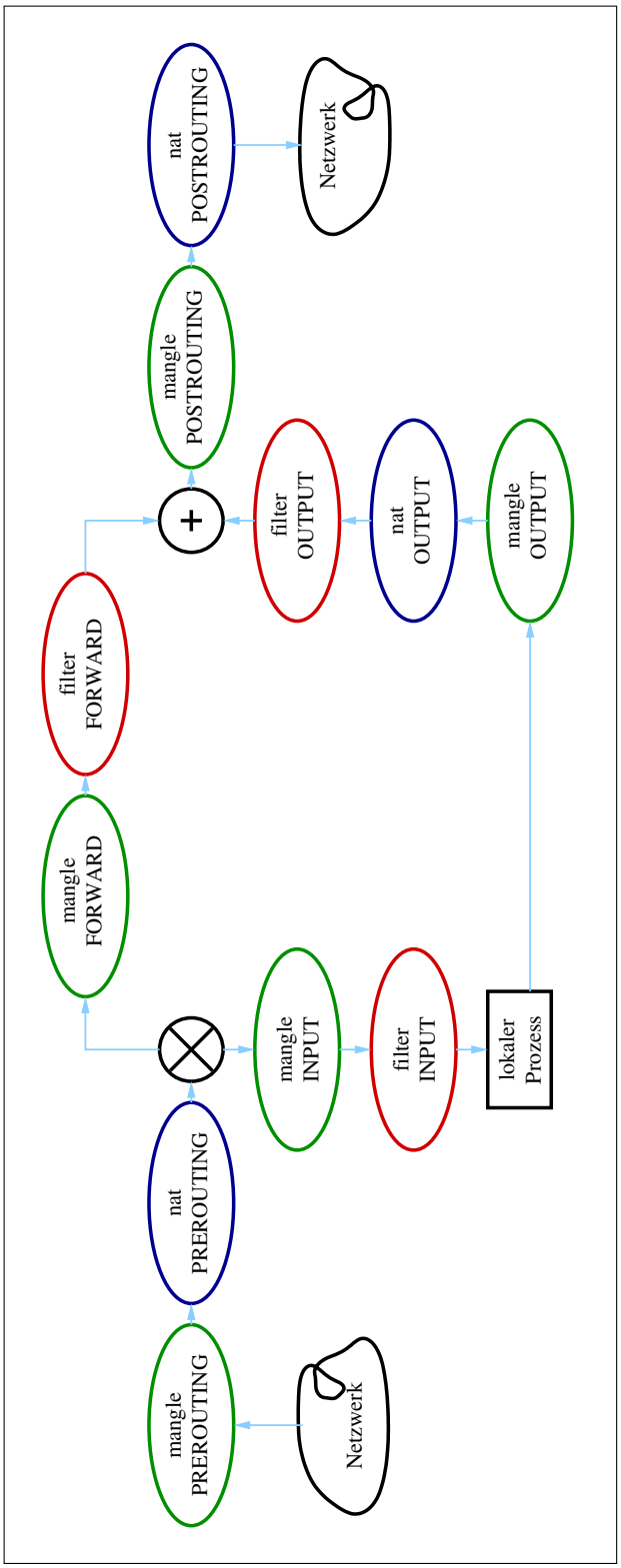

Abbildung 3: Weg der Pakete durch iptables

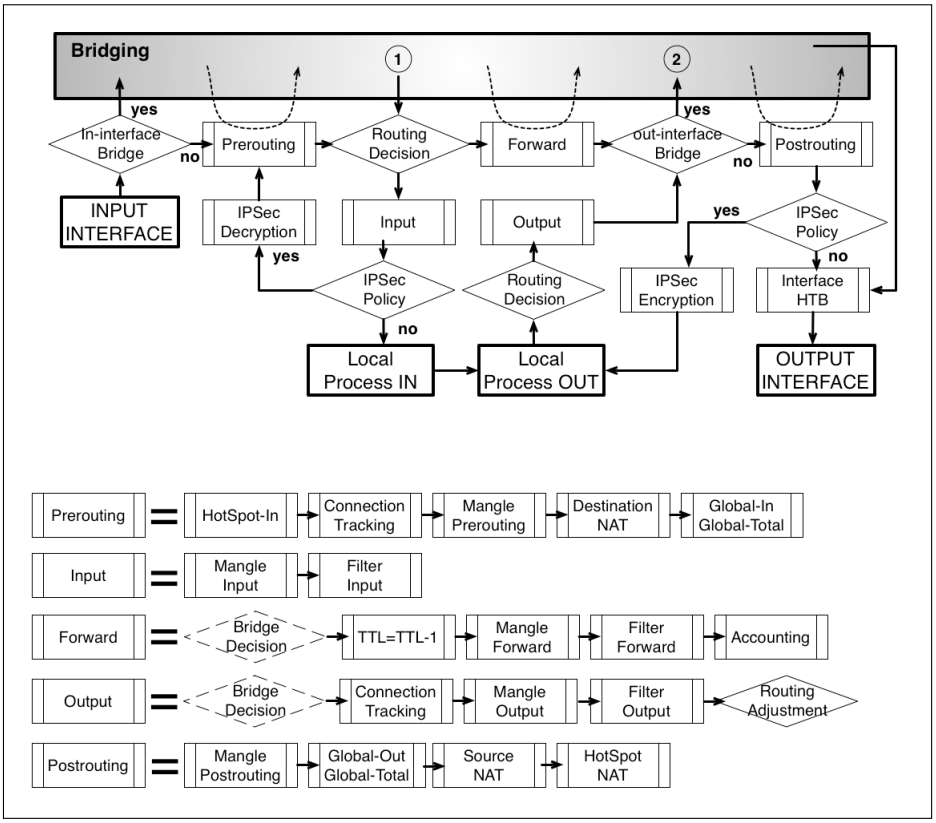

Abbildung 4: Schema der Paketweiterleitung auf Layer 3 auf MikroTik Routerboards

3 Paketfilterung auf MikroTik-Routerboards

Die Abb. 4 zeigt das Prinzip der Paketweiterleitung auf Layer3 (IP) von MikroTik-Routern. (`Quelle: http://wiki.mikrotik.com/wiki/Manual:Packet_Flow`) Da das MikroTik RouterOS auf dem Linux-Kernel basiert, sieht das Schema ähnlich dem von *iptables* aus.

3.1 Konfiguration der Firewall mit dem Router-OS

Untenstehende Tabelle listet auf, über welche Router-OS-Menues man die Filterregeln konfigurieren kann.

Regeln	Router-OS Menue
Filter Input, Filter Forward, Filter Output	`/ip firewall filter`
Source NAT, Destination NAT	`/ip firewall nat`

Tabelle 4: Konfiguration der Filter über das Router-OS-Menue

4 Eine DMZ mit RouterBoards

Die Abb. 5 zeigt, wie eine zweistufige DMZ mit RouterBoards aufgebaut werden soll.

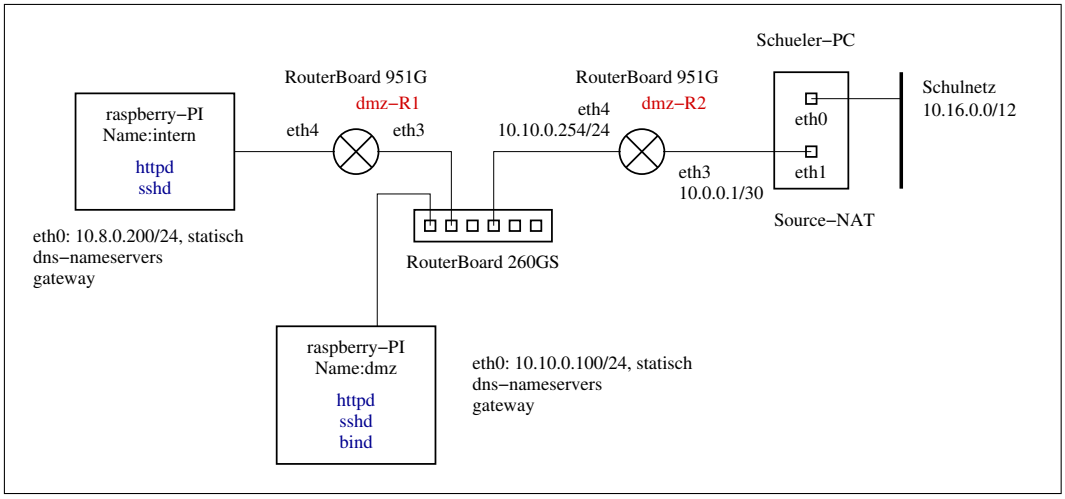

Abbildung 5: Zweistufige DMZ mit RouterBoards

Folgende Arbeiten sind notwendig, um diese Topologie aufzubauen:

1. Konfigurieren der Raspberry-Pis *dmz* und *intern*
   - Die Raspberries mit einem Notebook/PC oder einem RouterBoard (hier läuft ohnehin schon ein dhcp-Server) verbinden, das wiederum mit einem PC verbunden ist und sich per ssh anmelden
   - Feste Adressen vergeben. Siehe dazu Anhang A.1
   - Raspi neu starten, Netzwerk am PC anpassen und sich erneut verbinden
   - apache2 installieren (Anhang A.2)
   - Eine Begrüssungsseite schreiben
2. Konfigurieren des Routers dmz-R2 (Anhang B)
   - Alle Interfaces mit *master-port=none* konfigurieren.
   - ether4 und ether5 mit IP-Adressen versehen.
   - Statische Routen konfigurieren
3. Konfigurieren des Schüler-PCs (Anhang C)
   - Interface eth1 konfigurieren
   - Routen Richtung dmz setzen
   - Source-NAT und Routing einschalten

5 Absicherung des internen Netzwerks

Das interne Netzwerk soll durch sog. *Masquerading* weiter abgesichert werden. Dabei wird das in Abb. 6 gezeigte Netzwerkschema verwendet.

Das *interne* Netz ist nun **192.168.40.0/24**.

Im Netzwerkschema des Beispiels liegen die Adressen des **dmz**-Netzes (10.10.0.0/24) im privaten Klasse-A-Bereich 10.0.0.0/8. Möchte man die Konfiguration produktiv im Internet einsetzen, müsste man das DMZ-Netz jedoch mit *öffentlichen* IP-Adressen versehen.

Die Filter- und NAT-Regeln des Routers dmz-R1 sind im Anhang E enthalten.

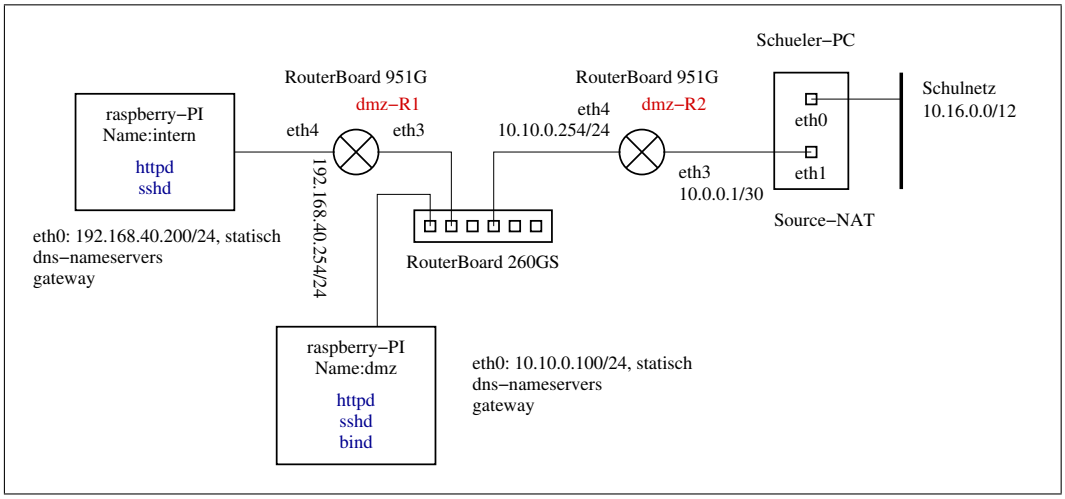

Abbildung 6: Zweistufige DMZ mit Masquerading

5.1 Source-NAT

Damit nun die Rechner im internen Netz Zugriff auf das Internet haben, muss der Router dmz-R1 die privten, internen Quelladressen in seine öffentliche Adresse übersetzen. Man nennt diesen Vorgang Source-NAT.

5.2 Destination-NAT

Damit man von aussen z.B. per ssh auf interne Rechner zugreifen kann, muss zusätzlich auf dmz-R1 eine Portweiterleitung eingerichtet werden.

Im Beispiel soll diese so eingestellt werden, dass Zugriffe auf IP=10.10.0.253 (ether3 von dmz-R1), Port=2222 auf IP=192.168.40.200, Port=22 übersetzt werden. Da hierbei die Zieladresse der ankommenden Pakete geändert wird, arbeitet dieses Verfahren mit Destination-NAT.

Wir müssen für den Zugriff auf die internen ssh-Server von aussen eine andere Portadresse als 22 nehmen, sonst kann man sich nicht mehr auf den Router dmz-R1 mit ssh verbinden.

A Raspberry konfigurieren

A.1 Netzwerk einrichten

Die Datei /etc/network/interfaces anpassen:

```
# interfaces(5) file used by ifup(8) and ifdown(8)

# Please note that this file is written to be used with dhcpcd
# For static IP, consult /etc/dhcpcd.conf and 'man dhcpcd.conf'

# Include files from /etc/network/interfaces.d:
source-directory /etc/network/interfaces.d

auto lo
iface lo inet loopback

#iface eth0 inet manual
auto eth0
iface eth0 inet static
  address 10.10.0.100
    netmask 255.255.255.0
    dns-nameservers 10.16.1.1
    gateway 10.10.0.254
```

A.2 apache2 installieren und einrichten

```
aptitude update
aptitude install apache2
```

Im Verzeichnis `/var/www/html` eine Begrüssungsdatei erzeugen:

```
<html>
<h1>das ist eine wenig informative seite auf dem raspi "dmz"</h1>
</html>
```

B Router dmz-R2

```
/interface ethernet
set ether3,ether4,ether5 master-port=none

/ip address
add address=10.10.0.254/24 interface=ether4-slave-local network=10.10.0.0
add address=10.0.0.1/30 interface=ether3-slave-local network=10.0.0.0

/ip route
add distance=1 gateway=10.0.0.2

/system identity
set name=dmz-R2
```

C Schüler-PC

C.1 Interface eth1 konfigurieren

Auszug aus der Datei /etc/network/interfaces:

```
auto eth1
#iface eth1 inet dhcp
iface eth1 inet static
      address 10.0.0.2
      netmask 255.255.255.252
```

C.2 Route setzen

```
ip route add 10.10.0.0/24 via 10.0.0.1
```

C.3 Source-NAT konfigurieren

Folgende Datei als Shell-Skript erzeugen und ausführen:

```
#Routing einschalten
echo 1 > /proc/sys/net/ipv4/ip_forward
#Nat loeschen
iptables -t nat -F
#Nat neu konfigurieren
iptables -A FORWARD -o eth0 -i eth1 -s 10.8.0.0/14 -m conntrack \
--ctstate NEW -j ACCEPT
iptables -A FORWARD -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
#Nat Tabelle auflisten
iptables -t nat -n -L
```

C.4 Routing-Tabellen

```
[admin@dmz-R1] > ip route print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        GATEWAY            DISTANCE
 0 A S  0.0.0.0/0                          10.10.0.254               1
 1 ADC  10.8.0.0/24        10.8.0.254      ether4-slave-local        0
 2 ADC  10.10.0.0/24       10.10.0.253     ether3-slave-local        0
 3 ADC  192.168.88.0/24    192.168.88.1    bridge-local              0

[admin@dmz-R2] /ip route> print
Flags: X - disabled, A - active, D - dynamic,
C - connect, S - static, r - rip, b - bgp, o - ospf, m - mme,
B - blackhole, U - unreachable, P - prohibit
 #      DST-ADDRESS        PREF-SRC        GATEWAY            DISTANCE
 0 A S  0.0.0.0/0                          10.0.0.2                  1
 1 ADC  10.0.0.0/30        10.0.0.1        ether3-slave-local        0
 2 A S  10.8.0.0/24                        10.10.0.253               1
 3 ADC  10.10.0.0/24       10.10.0.254     ether4-slave-local        0
 4 ADC  192.168.88.0/24    192.168.88.1    bridge-local              0

root@intern:/home/pi# ip route show
default via 10.8.0.254 dev eth0
10.8.0.0/24 dev eth0  proto kernel  scope link  src 10.8.0.200

pi@dmz:/home/pi# ip route show
default via 10.10.0.254 dev eth0
10.10.0.0/24 dev eth0  proto kernel  scope link  src 10.10.0.100

root@franck:/home/micha# ip route show
default via 192.168.178.1 dev eth0  proto static  metric 100
10.0.0.0/30 dev eth1  proto kernel  scope link  src 10.0.0.2
10.8.0.0/24 via 10.0.0.1 dev eth1
10.10.0.0/24 via 10.0.0.1 dev eth1
169.254.0.0/16 dev eth1  scope link  metric 1000
192.168.178.0/24 dev eth0  proto kernel  scope link  src 192.168.178.34  metric 100
```

D Firewall-Regeln dmz-R2

```
[admin@dmz-R2] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0   chain=input action=accept protocol=icmp

 1   ;;; default configuration
     chain=input action=accept connection-state=related

 2   ;;; default configuration
     chain=input action=accept connection-state=established

 3   chain=forward action=accept protocol=tcp
     dst-address=10.10.0.100 src-port=" dst-port=80

 4   chain=forward action=accept protocol=tcp
     dst-address=10.10.0.100 dst-port=22

 5   chain=forward action=log dst-address=10.10.0.100 log-prefix="

 6   chain=forward action=drop dst-address=10.10.0.100

 7   chain=forward action=accept protocol=tcp
     dst-address=10.8.0.0/24 dst-port=22

 8   chain=forward action=drop dst-address=10.8.0.0/24

 9   ;;; default configuration
     chain=forward action=accept connection-state=related

10   ;;; default configuration
     chain=forward action=accept connection-state=established

11   ;;; default configuration
     chain=forward action=drop connection-state=invalid
```

E Regeln für das Masquerading auf dmz-R1

E.1 Tabelle *filter*

```
[admin@dmz-R1] /ip firewall filter> print
Flags: X - disabled, I - invalid, D - dynamic
 0  D ;;; special dummy rule to show fasttrack counters
      chain=forward

 1    chain=forward action=accept connection-state=new
      src-address=192.168.40.0/24 in-interface=ether4-slave-local
      out-interface=ether3-slave-local log=no log-prefix="

 2    ;;; default configuration
      chain=forward action=accept connection-state=established,related log=no
      log-prefix="

 3    ;;; default configuration
      chain=input action=accept protocol=icmp log=no log-prefix="

 4    ;;; default configuration
      chain=input action=accept connection-state=established,related log=no
      log-prefix="

 5    ;;; default configuration
      chain=forward action=drop connection-state=invalid log=no log-prefix="
```

E.2 Tabelle *nat*

```
[admin@dmz-R1] /ip firewall nat> print
Flags: X - disabled, I - invalid, D - dynamic
 0    chain=srcnat action=masquerade out-interface=ether3-slave-local log=no
      log-prefix="

 1    chain=dstnat action=dst-nat to-addresses=192.168.40.200 to-ports=22
      protocol=tcp dst-port=2222 log=no log-prefix="
```

E.3 Zusätzliche Filterregeln auf dmz-R2

```
 9   chain=forward action=accept protocol=tcp
     dst-address=10.10.0.253 dst-port=22

10   chain=forward action=accept protocol=tcp
     dst-address=10.10.0.253 dst-port=2222

11   chain=forward action=drop dst-address=10.10.0.0/24
```