Red Hat Firewalld

Einleitung

Der Red Hat Firewalld ist eine übersichtlich konfigurierbare Einrichtung, mit welcher sehr effizient Netzwerk Kommunikation kontrolliert werden kann.

Mit einer Firewall können Systeme vor unerwünschtem Datenverkehr von aussen geschützt werden. Benutzer können den eingehenden Netzwerkverkehr auf Linux Servern steuern, indem sie eine Reihe von Firewall-Regeln definieren. Diese Regeln werden verwendet, um den eingehenden Verkehr zu kanalisieren und entweder zu blockieren oder durchzulassen. RedHat firewalld ist ein Firewall-Service-Daemon, der eine dynamisch anpassbare hostbasierte Firewall mit einer D-Bus-Schnittstelle bietet. Da es dynamisch ist, können Regeln erstellt werden, geändert und gelöscht werden, ohne dass der Firewall-Dämon bei jeder Änderung der Regeln neu gestartet werden muss.

Der RedHat firewalld verwendet die Konzepte von Zonen und Services, welche das Netzwerk Traffic Management vereinfachen. Zonen sind vordefinierte Regelwerke. Netzwerkschnittstellen und Quellen können einer Zone zugewiesen werden. Der zulässige Datenverkehr hängt von dem Netzwerk ab, mit dem das Linux System verbunden ist, und von der Sicherheitsstufe, welcher dieses Netzwerk zugewiesen ist. Firewall-Dienste (ssh, http, etc) sind vordefinierte Regeln, die alle erforderlichen Einstellungen für den eingehenden Datenverkehr für einen bestimmten Dienst abdecken und für eine Zone gelten. Dienste verwenden einen oder mehrere Ports oder Adressen für die Netzwerkkommunikation. Firewalls filtern die Kommunikation nach Ports. Um Netzwerkverkehr für einen Dienst zuzulassen, müssen die Ports geöffnet sein. **Firewalld blockiert den gesamten Verkehr an Ports, die nicht explizit als offen definiert sind**. Das Blockieren von Datenverkehr wird nicht geloggt, es wird ein “silent drop” gemacht. In einigen Zonen (z.B. vertrauenswürdig) ist standardmässig sämtlicher Datenverkehr zugelassen.

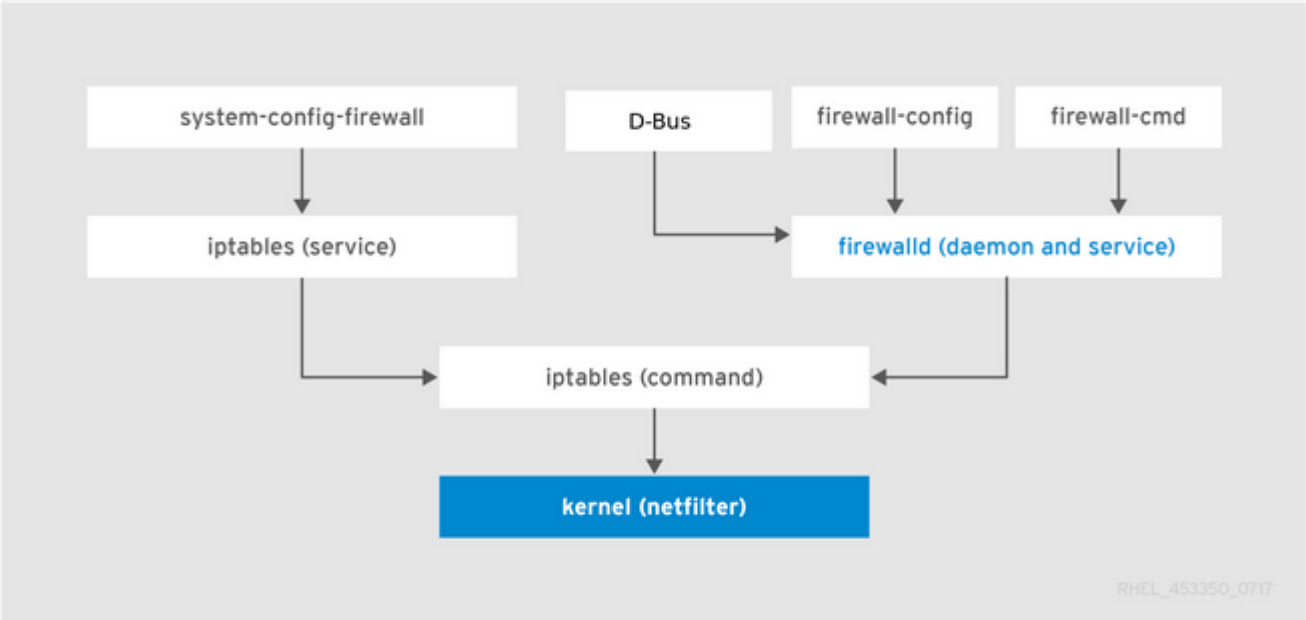

Figure 5.1. The Firewall Stack

Firewalld Zonen

Es gibt folgende Firewalld Zonen: Die folgende Auflistung ist von oben nach unten weniger strickt, also mehr vertraut. Oder anders ausgedrückt: Die oberste Zone ist diejenige, bei welcher alles suspekt und gesperrt ist und die unterste Zone ist diejenige, wo alles offen ist, weil vertrauenswürdig:

- `drop`: Any incoming network packets are dropped, there is no reply. Only outgoing network connections are possible.
- `block`: Any incoming network connections are rejected with an icmp-host-prohibited message for IPv4 and icmp6-adm-prohibited for IPv6. Only network connections initiated from within the system are possible.
- `public`: For use in public areas. You do not trust the other computers on the network to not harm your computer. Only selected incoming connections are accepted. This is the default zone. Every new and undefined network device will be placed here. This is the reason why this zone should not be used for active zone definitions.
- `external`: For use on external networks with masquerading enabled especially for routers. You do not trust the other computers on the network to not harm your computer. Only selected incoming connections are accepted.
- `dmz`: For computers in your demilitarized zone that are publicly-accessible with limited access to your internal network. Only selected incoming connections are accepted.
- `work`: For use in work areas. You mostly trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.
- `home`: For use in home areas. You mostly trust the other computers on networks to not harm your computer. Only selected incoming connections are accepted.
- `internal`: For use on internal networks. You mostly trust the other computers on the networks to not harm your computer. Only selected incoming connections are accepted.
- `trusted`: All network connections are accepted.

Vordefinierte Firewalld Services

Die vordefinierten Firewalld Services können unter /usr/lib/firewalld/services gefunden werden, sofern das Package firewalld-filesystem-xx-nn.el7.noarch.rpm installiert ist.

```
[rebermi@vnixeh ~]$ ls -lsa /usr/lib/firewalld/services/
total 516
12 drwxr-xr-x. 2 root root 8192 Jan 30 17:14 .
 0 drwxr-xr-x. 8 root root   98 Aug 17  2018 ..
 4 -rw-r--r--. 1 root root  412 Aug 17  2018 amanda-client.xml
 4 -rw-r--r--. 1 root root  447 Aug 17  2018 amanda-k5-client.xml
 4 -rw-r--r--. 1 root root  320 Aug 17  2018 bacula-client.xml
 4 -rw-r--r--. 1 root root  346 Aug 17  2018 bacula.xml
 4 -rw-r--r--. 1 root root  339 Aug 17  2018 bgp.xml
 4 -rw-r--r--. 1 root root  275 Aug 17  2018 bitcoin-rpc.xml
 4 -rw-r--r--. 1 root root  307 Aug 17  2018 bitcoin-testnet-rpc.xml
 4 -rw-r--r--. 1 root root  281 Aug 17  2018 bitcoin-testnet.xml
 4 -rw-r--r--. 1 root root  244 Aug 17  2018 bitcoin.xml
 4 -rw-r--r--. 1 root root  294 Aug 17  2018 ceph-mon.xml
```

```
 4 -rw-r--r--. 1 root root  329 Aug 17  2018 ceph.xml
 4 -rw-r--r--. 1 root root  168 Aug 17  2018 cfengine.xml
 4 -rw-r--r--. 1 root root  260 Aug 17  2018 condor-collector.xml
 4 -rw-r--r--. 1 root root  296 Aug 17  2018 ctdb.xml
 4 -rw-r--r--. 1 root root  305 Aug 17  2018 dhcpv6-client.xml
 4 -rw-r--r--. 1 root root  234 Aug 17  2018 dhcpv6.xml
 4 -rw-r--r--. 1 root root  227 Aug 17  2018 dhcp.xml
 4 -rw-r--r--. 1 root root  346 Aug 17  2018 dns.xml
 4 -rw-r--r--. 1 root root  374 Aug 17  2018 docker-registry.xml
 4 -rw-r--r--. 1 root root  391 Aug 17  2018 docker-swarm.xml
 4 -rw-r--r--. 1 root root  228 Aug 17  2018 dropbox-lansync.xml
 4 -rw-r--r--. 1 root root  338 Aug 17  2018 elasticsearch.xml
 4 -rw-r--r--. 1 root root  374 Aug 17  2018 ftp.xml
 4 -rw-r--r--. 1 root root  184 Aug 17  2018 ganglia-client.xml
 4 -rw-r--r--. 1 root root  176 Aug 17  2018 ganglia-master.xml
 4 -rw-r--r--. 1 root root  212 Aug 17  2018 git.xml
 4 -rw-r--r--. 1 root root  132 Aug 17  2018 gre.xml
 4 -rw-r--r--. 1 root root  603 Aug 17  2018 high-availability.xml
 4 -rw-r--r--. 1 root root  448 Aug 17  2018 https.xml
 4 -rw-r--r--. 1 root root  353 Aug 17  2018 http.xml
 4 -rw-r--r--. 1 root root  372 Aug 17  2018 imaps.xml
 4 -rw-r--r--. 1 root root  327 Aug 17  2018 imap.xml
 4 -rw-r--r--. 1 root root  454 Aug 17  2018 ipp-client.xml
 4 -rw-r--r--. 1 root root  427 Aug 17  2018 ipp.xml
 4 -rw-r--r--. 1 root root  554 Aug 17  2018 ipsec.xml
 4 -rw-r--r--. 1 root root  255 Aug 17  2018 ircs.xml
 4 -rw-r--r--. 1 root root  247 Aug 17  2018 irc.xml
 4 -rw-r--r--. 1 root root  264 Aug 17  2018 iscsi-target.xml
 4 -rw-r--r--. 1 root root  213 Aug 17  2018 jenkins.xml
```

Ein Dienst kann eine Liste lokaler Ports und Destinationen sowie eine Liste von Firewall-Hilfsmodulen sein, die automatisch geladen werden, wenn ein Dienst aktiviert ist. Die Verwendung vordefinierter Dienste erleichtert es dem Benutzer, den Zugriff auf einen Dienst zu aktivieren und zu deaktivieren. Die Verwendung der vordefinierten Dienste oder der benutzerdefinierten Dienste im Gegensatz zum Öffnen von Ports oder Portbereichen kann die Verwaltung vereinfachen. Dienstkonfigurationsoptionen und Informationen zu generischen Dateien werden in der Manpage firewalld.service (5) beschrieben. Die Services werden durch individuelle XML-Konfigurationsdateien angegeben, die im folgenden Format benannt werden: service-name.xml. Ein vordefinierter Dienst erlaubt grundsätzlich einen definierten Port von Quelle 0.0.0.0 bis zum Ziel 0.0.0.0. Es ist nicht möglich, einem vordefinierten Dienst Quellziel hinzuzufügen. Vordefinierte Dienste sind XML-Dateien, die unter /usr/lib/firewalld/services gespeichert sind

```
[rebermi@vnixeh ~]$ cat /usr/lib/firewalld/services/http.xml
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>WWW (HTTP)</short>
  <description>HTTP is the protocol used to serve Web pages. If you plan to
make your Web server publicly \
   available, enable this option. This option is not required for viewing
```

```
pages locally or developing Web pages.</description>
  <port protocol="tcp" port="80"/>
</service>
```

Firewalld lokale Konfiguration

Die lokale aktive Konfiguration für den Firewalld ist unter /etc/firewalld abgelegt.

- Pfad: **/etc/firewalld/zones/**: Das ist der Pfad für die aktiv konfigurierten Zonen. Hier sind die XML Dateien abgelegt, in welchen die aktive firewalld Konfiguration abgespeichert ist
- Pfad: **/etc/firewalld/services/**: Das ist der Pfad für die definierten OEM Servces. Hier können Services definiert werden, welche selbst erstellt werden, oder Kopien von modifizierten Services von /usr/lib/firewalld/services abgelegt werden. Beispiel: Wenn für http Port 8080 gelten soll, anstelle Port 80, so kann eine Kopie von /usr/lib/firewalld/services/http.xml nach /etc/firewalld/services/http.xml kopiert werden und die Zeile port="80" modifiziert werden hzu port="8080". (Nachteil: Sofern Red Hat einen Bugfix, oder eine Erweiterung zum vordefinierten Service macht im Zusammenhang mit einem Feature Update am firewalld.service, könnte es vorkommen, dass der OEM Service nicht mehr funktioniert.)

Firewalld Zone Internal

Für die allermeisten Fälle inhouse kann die Firewalld Zone internal angewendet werden. Es sollte niemals die default Zone public angewendet werden, weil dort default Services aktiv sind.

Im Beispiel sind vordefinierte Services, welche auch Port Definitionen beinhalten, das Netzwerk Interface, für welches die Zonen Definition gelten soll und noch zusätzliche TCP Ports definiert. Die Services depo, ZDS und ovito sind selbsgt definierte Services unter /etc/firewalld/services. Es sind einzelne TCP Ports und auch Port Ranges definiert.

Die Firewall Konfiguration kann editiert werden. Damit die Konfiguration aktiv wird, genügt eine Aktualisierung `# firewall-cmd –reload`

Vorsicht: Ist die Firewall Konfiguration falsch, oder fehlt der Service ssh, so ist es möglich, sich selbst von System auszuschliessen. Es braucht danach Konsole Zugriff, um das Problem zu lösen.

```
[root@vtsttc ~]# cat /etc/firewalld/zones/internal.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Internal</short>
  <description>For use on internal networks. You mostly trust the other
computers on the networks to not harm your computer. \
Only selected incoming connections are accepted.</description>
  <interface name="ens160"/>
  <service name="depo"/>
  <service name="ssh"/>
  <service name="zds"/>
  <service name="ovito"/>
```

```
  <port protocol="tcp" port="7111"/>
  <port protocol="tcp" port="7000-7099"/>
</zone>
[root@vtsttc ~]#
```

Firewalld Zone Public

Die Firewalld Zone public beinhaltet default dhcp und ssh. Diese Services dürfen default nicht zugelassen werden in einem Umfeld, wo feste IP Adressierung eingerichtet ist, deshalb muss unter /etc/firewalld/zones die Public Zone ohne Definitionen angelegt werden. Es muss deshalb immer eine leere Konfiguration von public.xml gemacht werden.

```
[root@vtsttc ~]# cat /etc/firewalld/zones/public.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Public</short>
  <description>For use in public areas. You do not trust the other computers
\
on networks to not harm your computer. Only selected incoming connections
are accepted.</description>
</zone>
[root@vtsttc ~]#
```

Firewalld Service

Der Firewalld ist ein systemd Service

```
[root@vtsttc ~]# systemctl status firewalld.service
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled;
vendor preset: enabled)
   Active: active (running) since Tue 2018-11-06 08:33:48 CET; 3 months 15
days ago
     Docs: man:firewalld(1)
 Main PID: 801 (firewalld)
   CGroup: /system.slice/firewalld.service
           └─801 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid
```

Konfiguration im NIC Script

Die konfigurierte Firewalld Zone (ZONE=internal) **muss** im Netzwerk Interface Script eingetragen sein, damit die Regeln korrekt funktionieren. **Wird dieser Eintrag gelöscht, ist man vom System**

ausgesperrt.

```
[root@vtsttc ~]# cat /etc/sysconfig/network-scripts/ifcfg-ens160
# Generated by parse-kickstart
UUID="6690826a-807b-4ba1-b901-42d4f698c4fe"
IPADDR="172.18.10.20"
GATEWAY="172.18.10.254"
NETMASK="255.255.255.0"
BOOTPROTO="static"
DEVICE="ens160"
ZONE=internal
ONBOOT="yes"
IPV6INIT="yes"
NM_CONTROLLED=no
[root@vtsttc ~]#
```

Konfiguration auslisten

Für alle interaktiven Konfigurationen ist das Kommando frewall-cmd vorhanden, siehe auch # man firewall-cmd

```
[root@vtsttc ~]# firewall-cmd --zone=internal --list-all
internal (active)
  target: default
  icmp-block-inversion: no
  interfaces: ens160
  sources:
  services: depo ssh zds ovito
  ports: 7111/tcp 7000-7099/tcp
  protocols:
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
[root@vtsttc ~]#
```

Besondere Konfigurationen

Es lassen sich nicht alle Fälle einfach mit Einträgen: port protocol=“tcp” port=“7111” im Service oder Zone File abhandeln.

Source Destination Rules (Rich-Rules)

Müssen komplexe Rules definiert werden, wo eine Source IP auf einen bestimmten IP Port erlaubt werden soll, braucht es Rich Rules

Regel Interaktiv setzen

In diesem Beispiel sollen bestimmte IP Adressen auf einen IP Port erlaubt werden.

```
# firewall-cmd --zone=internal --permanent --add-rich-rule='rule
family="ipv4" source address="172.17.146.11" port "port=7001" protocol="tcp"
accept'
# firewall-cmd --zone=internal --permanent --add-rich-rule='rule
family="ipv4" source address="172.17.146.12" port "port=7001" protocol="tcp"
accept'
# firewall-cmd --zone=internal --permanent --add-rich-rule='rule
family="ipv4" source address="10.1.100.166" port "port=7001" protocol="tcp"
accept'

# firewall-cmd --reload
```

Regel in der Zone Konfiguration internal.xml

Es wäre auch möglich, eine Rich Rule direkt im Zone Konfigurations File zu editieren. Es ist auch möglich Rules für IPv4 und andere Rules für IPv6 zu setzen

```
[root@vtsttc ~]# cat /etc/firewalld/zones/internal.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Internal</short>
  <description>For use on internal networks. You mostly trust the other
computers on the networks to not harm your computer.
  Only selected incoming connections are accepted.</description>
  <interface name="ens160"/>
  <service name="ssh"/>
  <service name="zds"/>
  <service name="ovito"/>
  <rule family="ipv4">
    <source address="172.17.146.11"/>
    <port protocol="tcp" port="7001"/>
    <accept/>
  </rule>
  <rule family="ipv4">
    <source address="172.17.146.12"/>
    <port protocol="tcp" port="7001"/>
    <accept/>
  </rule>
  <rule family="ipv4">
```

```
    <source address="10.1.100.166"/>
    <port protocol="tcp" port="7001"/>
    <accept/>
  </rule>
</zone>
[root@vtsttc ~]#
```

Port Ranges mit Source Destination Rule

Regel interaktiv setzen

```
# firewall-cmd --zone=internal --permanent --add-rich-rule='rule
family="ipv4" source address="172.17.146.11" port "port=7001-7010"
protocol="tcp" accept'
# firewall-cmd --reload
```

Regel im Zone File

```
[root@vtsttc ~]# cat /etc/firewalld/zones/internal.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Internal</short>
  <description>For use on internal networks. You mostly trust the other
computers on the networks
 to not harm your computer. Only selected incoming connections are
accepted.</description>
  <interface name="ens160"/>
  <service name="ssh"/>
  <rule family="ipv4">
    <source address="172.17.146.11"/>
    <port protocol="tcp" port="7001-7010"/>
    <accept/>
  </rule>
</zone>
[root@vtsttc ~]#
```

Konfiguration mit Port Forwarding

Firewall Rule Interaktiv

Hier wird der Port 80 von Aussen auf den internen Port 7039 weiter geleitet. Dasselbe für den externen Port 443, welcher auf Port 7040 umgeleitet wird.

```
# firewall-cmd --permanent --zone=internal --add-forward-
```

```
port=port=80:proto=tcp:toport=7039
# firewall-cmd --permanent --zone=internal --add-forward-
port=port=443:proto=tcp:toport=7040

# firewall-cmd --reload
```

Firewall Rule im Zone File

Im Zone Konfigurationsfile sind die Einträge forward-port to-port gesetzt.

```
[root@vtsttc ~]# cat /etc/firewalld/zones/internal.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Internal</short>
  <description>For use on internal networks. You mostly trust the other
computers on the networks
to not harm your computer. Only selected incoming connections are
accepted.</description>
  <interface name="ens160"/>
  <service name="ssh"/>
  <forward-port to-port="7039" protocol="tcp" port="80"/>
  <forward-port to-port="7040" protocol="tcp" port="443"/>
</zone>
[root@vtsttc ~]#
```

Konfiguration Rich Rule Source Destination mit Port Forwarding

Firewall Rule Interaktiv

In diesem Beispiel werden nur Verbindungen von IP 172.17.146.11 auf Port 443 zugelassen und auf Port 7040 weiter geleitet, Dasselbe für Verbindungen auf 80, welche auf Port 7039 umgeleitet werden.

```
# firewall-cmd --permanent --zone=internal --add-rich-rule='rule family=ipv4
source address="172.17.146.11" forward-port port=443 protocol=tcp to-
port=7040'
# firewall-cmd --permanent --zone=internal --add-rich-rule='rule family=ipv4
source address="172.17.146.11" forward-port port=80 protocol=tcp to-
port=7039'

# firewall-cmd --reload
```

Firewall Rule im Zone File

Im Zone Konfiguration File sind die Rules pro Port-Forward eingetragen. Es können hier wiederum verschiedene Rules für IPv4 und auch für IPv6 eingetragen sein. Als Source Address braucht es dann eine IPv6 Adresse und rule family = “ipv6”

```
[root@vtsttc ~]# cat /etc/firewalld/zones/internal.xml
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>Internal</short>
  <description>For use on internal networks. You mostly trust the other
computers on
the networks to not harm your computer. Only selected incoming connections
are accepted.</description>
  <interface name="ens160"/>
  <service name="ssh"/>
  <rule family="ipv4">
    <source address="172.17.146.11"/>
    <forward-port to-port="7040" protocol="tcp" port="443"/>
  </rule>
  <rule family="ipv4">
    <source address="172.17.146.11"/>
    <forward-port to-port="7039" protocol="tcp" port="80"/>
  </rule>
</zone>
[root@vtsttc ~]#
```

Möglichkeit Outgoing Ports zu sperren

Soll von einem Service, welcher nicht konfiguriert werden kann, der outgoing Port gesperrt werden, so braucht es eine Rich Rule, welcher ein DROP macht. Auf diese Weise könnte eine UDP Kommunikation gesperrt werden.

Konfiguration interaktiv

Hier wird nicht mit der Firewall Zone gearbeitet, sondern mit der Konfiguration direct

```
# firewall-cmd --direct --permanent --add-rule ipv4 filter OUTPUT 1 -p tcp -
m tcp --dport=5200 -j DROP
# firewall-cmd --direct --permanent --add-rule ipv4 filter OUTPUT 1 -p udp -
m udp --dport=5222 -j DROP
```

Die Drop Targets werden nun mit `# iptables -L -n -v` in der Chain OUTPUT_direct sichtbar

```
[root@vtsttc ~]# iptables -L OUTPUT_direct -n -v
Chain OUTPUT_direct (1 references)
 pkts bytes target     prot opt in     out     source
destination
```

```
    0     0 DROP       tcp  --  *      *       0.0.0.0/0
0.0.0.0/0           tcp dpt:5200
    0     0 DROP       udp  --  *      *       0.0.0.0/0
0.0.0.0/0           udp dpt:5222
[root@vtsttc ~]#
```

Konfiguration /etc/firewalld/direct.xml

Der Eintrag in der “direct” Konfiguration ist

```
[root@vtsttc ~]# cat /etc/firewalld/direct.xml
<?xml version="1.0" encoding="utf-8"?>
<direct>
  <rule priority="1" table="filter" ipv="ipv4" chain="OUTPUT">-p tcp -m tcp
--dport=5200 -j DROP</rule>
  <rule priority="1" table="filter" ipv="ipv4" chain="OUTPUT">-p udp -m udp
--dport=5222 -j DROP</rule>
</direct>
[root@vtsttc ~]#
```

Möglichkeit interaktiv konfigurierte Ports zu sperren

Mit der Firewall können incoming Ports gesperrt werden, indem diese NICHT in der Konfiguration des Zone Files enthalten sind. Default sind bei aktivem firewalld.service alle Ports, welche nicht explizit im Zone File enthalten sind gesperrt; - es wird ein “silent drop” gemacht.

Man kann nun interaktiv, definierte Ports aus dem Zone Konfigurationsfile entfernen mit dem Kommando firewall-cmd –zone=internal –remove-port='portnummer'/tcp oder port1-port2/tcp. Ports können NICHT von definierten Services entfernt werden!

```
[root@vtsttc firewalld]# grep port zones/internal.xml
  <port protocol="tcp" port="7111"/>
  <port protocol="tcp" port="7000-7099"/>
[root@vtsttc firewalld]# firewall-cmd --zone=internal --permanent --remove-
port=7111/tcp
success
[root@vtsttc firewalld]# grep port zones/internal.xml
  <port protocol="tcp" port="7000-7099"/>
[root@vtsttc firewalld]# firewall-cmd --zone=internal --permanent --remove-
port=7000-7099/tcp
success
[root@vtsttc firewalld]# grep port zones/internal.xml
[root@vtsttc firewalld]#
```

Besonderheit Service FTP

Ein FTP Transfer kann im PASV Mode abgehandelt werden. Das ist dann von Vorteil, wenn auf der Server Seite kein Verbindungsaufbau zum Client möglich ist, z.B. hinter einer Firewall. Der Passive (PASV) FTP Mode vom Client erlaubt es, nach dem ersten Verbindungsaufbau zum Server einen Port > 1023 zur Datenübertragung auszuhandeln. Dann erfolgt der Transfer auf diesem ausgehandelten Port. Für die FTP-Übertragung im PASV-Modus müssen bestimmte Datentransport Ports geöffnet werden. Daher muss ein FTP Conntrack Servicemodul geladen werden. Der standardmässige Red Hat Firewall Service ftp.xml enthält bereits das Contrack-Modul. Wenn FTP über einen anderen Port als 21 erfolgen soll, muss ein OEM-Firewalld-Dienst unter /etc/firewalld/services erstellt werden.

```
[root@vtsttc ~]# cat /usr/lib/firewalld/services/ftp.xml
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>FTP</short>
  <description>FTP is a protocol used for remote file transfer.
  If you plan to make your FTP server publicly available, enable this
option.
  You need the vsftpd package installed for this option to be
useful.</description>
  <port protocol="tcp" port="21"/>
  <module name="nf_conntrack_ftp"/>
</service>
[root@vtsttc ~]#
```

Troubleshooting

Wird mit RedHat Firewall gearbeitet, so steht man oft vor dem Problem, dass eine Kommunikation nicht funktioniert. Folgende Schritte könnten hilfreich sein bei der Problemlösung:

Firewalld abschalten: Funktioniert die Verbindung danach? Wenn JA: Ruleset überprüfen.

Port Verbindung überprüfen (nicht telnet, sondern netcat)

Netcat ist default unter RedHat installiert. Netcat kann viel mehr und hat auch eine Telnet Abfrage für Ports integriert. Weiter verfügt Netcat über eine sehr gute man page!

Zum Port debuggung deshalb mit netcat (nc) anstelle telnet arbeiten: Link Digitalocean.

Beispiel 1: Telnet /TCP Port Abfrage

Beispiele: TCP Port 10250 Verbindung überprüfen -i 1 bedeutet, dass bei einem Connect nach 1s idle Time die Verbindung geschlossen werden soll. Damit ist es möglich, in einem Loop viele Ports, oder

viele Server abzufragen. (-t Telnet Abfrage) (-v Verbose)

```
[rebermi@vostm1 ~]$ nc -vt vostn1.pnet.ch 10250 -i 1
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connection refused.
[rebermi@vostm1 ~]$

[rebermi@vosme1 ~]$ nc -vt vosne1.pnet.ch 10250 -i 1
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connected to 172.18.184.14:10250.
Ncat: Idle timeout expired (1000 ms).
[rebermi@vosme1 ~]$
```

Beispiel 2: UDP Port Abfrage

Beispiel UDP Port Verbindung überprüfen (-u UDP Abfrage)

```
[rebermi@vostm1 ~]$  nc -vu vostn1.pnet.ch 4789 -i 1
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connected to 172.18.184.19:4789.
Ncat: Idle timeout expired (1000 ms).
[rebermi@vostm1 ~]$
```

Last update: **2019/03/07 10:47**