Firewalld

Проверим, запущен ли firewalld:

```
# systemctl status firewalld
```

Тут будет расширенная информация. Чтобы коротко, да (работает) или нет можно так:

```
# firewall-cmd --state
running
```

Остановка firewalld:

```
# systemctl stop firewalld
```

Запрет автостарта:

```
# systemctl disable firewalld
```

Запуск firewalld:

```
# systemctl start firewalld
```

Включение автостарта:

```
# systemctl enable firewalld
```

Зоны firewalld

В firewalld широко используется понятие зоны. Список всех допустимых зон по-умолчанию:

```
# firewall-cmd --get-zones
block dmz drop external home internal public trusted work
```

Назначение зон (условно, конечно):

- drop – все входящие пакеты отбрасываются (drop) без ответа. Разрешены только исходящие соединения.
- block – входящие соединения отклоняются (rejected) с ответом icmp-host-prohibited (или icmp6-adm-prohibited). Разрешены только инициированные системой соединения.

- public – **зона по-умолчанию**. Из названия ясно, что эта зона нацелена на работу в общественных сетях. Мы не доверяем этой сети и разрешаем только определенные входящие соединения.
- external – зона для внешнего интерфейса роутера (т.н. маскарадинг). Разрешены только определенные нами входящие соединения.
- dmz – зона DMZ, разрешены только определенные входящие соединения.
- work – зона рабочей сети. Мы все еще не доверяем никому, но уже не так сильно, как раньше Разрешены только определенные входящие соединения.
- home – домашняя зона. Мы доверяем окружению, но разрешены только определенные входящие соединения
- internal – внутренняя зона. Мы доверяем окружению, но разрешены только определенные входящие соединения
- trusted – разрешено все.

Список всех активных зон:

```
# firewall-cmd --get-active-zones
public
  interfaces: enp1s0
```

Ага, зона *public*, к которой присоединен сетевой интерфейс *enp1so*. Дальше в зону *public* добавим новый порт, на котором будет висеть sshd.

Зная имя сетевого интерфейса (например, enp1s0), можно узнать, к какой зоне он принадлежит:

```
# firewall-cmd --get-zone-of-interface=enp1s0
public
```

А можно узнать, какие интерфейсы принадлежат конкретной зоне:

```
# firewall-cmd --zone=public --list-interfaces
enp1s0
```

Пример: разрешаем ssh на нестандартном порте

Давайте разрешим доступ к серверу по ssh на порте 2234/tcp, а не на 22/tcp, как по-умолчанию. Попутно чуть-чуть коснемся selinux.

Сначала посмотрим, что вообще разрешено постоянно на нашем сервере:

```
# firewall-cmd --permanent --list-all
public (default)
  interfaces:
  sources:
  services: ssh dhcpv6-client
  masquerade: no
  forward-ports:
  icmp-blocks:
  rich rules:
```

Я не использую пока ipv6, поэтому сразу уберу соотв. правило из firewalld:

```
#  firewall-cmd  --permanent  --zone=public  --remove-service=dhcpv6-client
```

Разрешим на постоянной основе (чтобы после перезагрузки не потерлось) соединение на порт 2234/tcp (на него повесим sshd):

```
# firewall-cmd --permanent --zone=public --add-port=2234/tcp
```

Перезагрузим правила:

```
# firewall-cmd --reload
```

Проверим:

```
# firewall-cmd --zone=public --list-ports
2234/tcp
```

Ок, порт открыт. Редактируем конфиг sshd:

```
# nano /etc/ssh/sshd_config
```

```
...
port 2234
...

# systemctl restart sshd.service
```

Но SELinux, которую вы, надеюсь, не отключали, не даст подключиться к ssh на нестандартном порте (порт 2234/tcp для sshd – нестандартный). Вы можете этот шаг пропустить и проверить, как сработает защита SELinux, а можете сразу все настроить:

```
# yum provides semanage
# yum install policycoreutils-python
# semanage port -a -t ssh_port_t -p tcp 2234
```

Вот теперь все ок. Проверяем подключение по ssh на новом порте. Если все ок, закрываем доступ к порту 22:

```
# firewall-cmd --permanent --zone=public --remove-service=ssh
# firewall-cmd --reload
```

Смотрим, что получилось:

```
# firewall-cmd --list-all
public (default, active)
  interfaces:
  sources:
  services:
  ports: 2234/tcp
  masquerade: no
  forward-ports:
  icmp-blocks:
  rich rules:
```

Вот и все.

Разные полезные команды:

Включить режим блокировки всех исходящих и входящих пакетов:

```
# firewall-cmd --panic-on
```

Выключить режим блокировки всех исходящих и входящих пакетов:

```
# firewall-cmd --panic-off
```

Узнать, включен ли режим блокировки всех исходящих и входящих пакетов:

```
# firewall-cmd --query-panic
```

Перезагрузить правила firewalld без потери текущих соединений:

```
# firewall-cmd --reload
```

Перезагрузить правила firewalld и сбросить текущие соединения (рекомендуется только в случае проблем):

```
# firewall-cmd --complete-reload
```

Добавить к зоне сетевой интерфейс:

```
# firewall-cmd --zone=public --add-interface=em1
```

Добавить к зоне сетевой интерфейс (сохранится после перезагрузки firewall):

```
# firewall-cmd --zone=public --permanent --add-interface=em1
```

Можно в конфиге ifcfg-enp1s0 указать, какой зоне принадлежит этот интерфейс. Для этого добавим ZONE=work в файл /etc/sysconfig/network-scripts/ifcfg-enp1s0. Если параметр ZONE не указан, будет назначена зона по-умолчанию (параметр DefaultZone в файле /etc/firewalld/firewalld.conf.

Разрешить диапазон портов:

```
# firewall-cmd --zone=public --add-port=5059-5061/udp
```

Маскарад (masquerade, он же nat, он же…):

Проверить статус:

```
# firewall-cmd --zone=external --query-masquerade
```

Включить:

```
# firewall-cmd --zone=external --add-masquerade
```

Здесь надо отметить, что вы можете включить *masquerade* и для зоны *public*, например.

Перенаправить входящие на 22 порт на другой хост:

```
# firewall-cmd --zone=external --add-forward-port=port=22:proto=tcp:toaddr=192.168.1.23
```

Перенаправить входящие на 22 порт на другой хост с изменением порта назначения (с 22 на 192.168.1.23:2055):

```
# firewall-cmd --zone=external /
--add-forward-
port=port=22:proto=tcp:toport=2055:toaddr=192.168.1.23
```

На этом закончу, т.к. примеров может быть бесконечно много. Добавлю только, что лично я не составил окончательно свое мнение по поводу нововведения firewalld, т.к. к синтаксису привыкаешь долго и если в вашем зоопарке встречаются разные OS Linux, то по первости могут быть проблемы именно с привычкой. Но освоив firewalld, вы расширите кругозор – чаще всего, это стоит затраченных усилий.

Не хочу firewalld! Верните мне старый iptables!

Если все же вы хотите вернуть прошлое и заменить firewalld на iptables, то сделать это совсем не трудно:

```
# systemctl disable firewalld
# systemctl stop firewalld
```

Ставим старый добрый iptables:

```
# yum install iptables-services
```

Запускаем брандмауэр:

```
# systemctl start iptables
# systemctl start ip6tables
```

Автозапуск при включении:

```
# systemctl enable iptables
# systemctl enable ip6tables
```

Для сохранения правил iptables после перезагрузки:

```
# iptables-save > /etc/sysconfig/iptables
# ip6tables-save > /etc/sysconfig/ip6tables
```

Или по-старинке:

```
# service iptables save
```

Для загрузки правил iptables после перезагрузки:

```
# iptables-restore < /etc/sysconfig/iptables
# ip6tables-restore < /etc/sysconfig/ip6tables
```

Текущие правила находятся в файлах:

```
/etc/sysconfig/iptables
/etc/sysconfig/ip6tables
```

Перезапуск iptables (например, после совершения каких-либо изменений):

```
# systemctl restart iptables.service
```

http://bozza.ru/art-259.html