Firewalld

Tant Fedora com Ubuntu ofereixen la possibilitat d'utilitzar un servei anomenat firewalld (*systemctl status firewalld*, etc), el qual funciona com una capa per sobre d'Nftables i que permet manipular les seves regles d'una forma més "amigable", a més d'oferir altres funcionalitats, com ara una interfície D-Bus (la qual permet modificar dinàmicament les regles del tallafocs sense haver de reiniciar cap servei), la implementació del concepte de zones, la gestió de la persistència de les regles (les quals es poden definir dins de fitxers específics del propi Firewalld sense haver de recòrrer a la sintaxis Nftables), etc.

NOTA: Ubuntu no porta instal.lat de sèrie Firewalld perquè la seva solució "equivalent" és Ufw. Si es vol fer servir Firewalld, doncs, primer caldrà desinstal.lar Ufw (*sudo apt remove ufw*)

To manage Firewalld from command line you can use *firewalld-cmd* command. For instance, to get the status of firewalld (that is: to confirm whether it's running), you can also execute: *sudo firewall-cmd --state*

NOTA: Other ways to configure Firewalld are via the graphical tool called "firewall-config" or via its D-BUS interface

Directory "**/usr/lib/firewalld**" contains the default configuration provided by firewalld package for several "concepts" we will study later, like **icmptypes, services** and **zones**. These files should not get modified as the changes are gone with an update of the firewalld package, so additional icmptypes, services and zones should be provided by creating custom files in "**/etc/firewalld**" (either by hand copying desired file from "/usr/lib/firewalld" to "/etc/firewalld" folder and changing it accordingly, or via firewalld admin tools). These custom files will overload the default configuration files.

NOTA: All of these concepts are explained in the firewalld.service(5) man page.

Services

The most straightforward method to control traffic is to add a (predefined) "service" to Firewalld. A Firewalld "service" is a combination of port and/or protocol entries; optionally with an IP destination address added, which can be "opened". Each "service" is defined in an individual XML configuration file inside "/usr/lib/firewalld/services" or "/etc/firewalld/services" folders, and it must be named in the following format: *service-name*.xml.

*To get a list of all recognized services: *firewall-cmd --get-services*
*To get a list of all recognized icmptypes: *firewall-cmd --get-icmptypes*
NOTA: You can add the *–permanent* argument in above commands to list the recognized permanent services or icmptypes, respectively

*To add permanent rules to default zone to allow "http","https" and "radius" services (that is, to "open" their associated ports):
sudo firewall-cmd --add-service={http|https|radius} --permanent and...
NOTA: If you want to add a service to another zone different from default one, add the *--zone=<zone>* argument

*...reload firewalld for this change (or any other) to take effect (without loosing state information):
sudo firewall-cmd –reload

Note any changes committed in runtime (that is, specified without the *–permanent* argument) only apply while firewalld is running: when firewalld is restarted, the settings revert to their permanent values. To make changes persistent across reboots, as you already know, you should apply them using the *--permanent* argument. Note, however, this argument doesn't affect inmediately on runtime but only after a reload or restart of firewalld (using the *--reload* argument, for instance), action which closes all open ports and stops the networking traffic. There's a more graceful alternative, though: you can make already applied (and tested) runtime changes be persistent if you execute *sudo firewall-cmd --runtime-to-permanent*

NOTA: It is possible, especially on remote systems, that an incorrect setting results in a user locking themselves out of a machine. To prevent such situations, use the *--timeout* <n> option. After a specified amount of time (30s, 15m, 1h, etc) , any change reverts to its previous state. Using this options excludes the --permanent option. For example, to add the SSH service for 15 minutes: *sudo firewall-cmd --add-service=ssh --timeout 15m*

*To confirm that services were successfully added to default zone: *firewall-cmd --list-services*
NOTA: If you want to know the list of services added to another zone, you can add the *--zone=<zone>* argument

*To remove a service from a zone: *sudo firewall-cmd --remove-service=<service>*
NOTA: If you want to remove a service from another zone different from default one, add the *--zone=<zone>* argument

*To query if a service is enabled in a zone: *firewall-cmd --query-service=<service>*
This returns "yes" if the service is enabled in the zone, otherwise "no"
NOTA: If you want to query a service from another zone different from default one, add the *--zone=<zone>* argument

*To get more information about a defined service: *firewall-cmd --info-service=<service>*

You can also manage ports/protocols directly (without having defined previously any "service") with these commands:

*To open ("enable") a port and protocol combination in a zone:
sudo firewall-cmd [--zone=<zone>] --add-port=<n>[-<m>]/<protocol>
The port can be a single port <n> or a port range <n>-<m>. The protocol can be "tcp" or "udp".
NOTA: This command has the *--timeout* argument, too.

*To disable a port and protocol combination in a zone:
sudo firewall-cmd [--zone=<zone>] --remove-port=<n>[-<m>]/<protocol>

*To list the opened ("enabled") ports in a zone: *firewall-cmd [--zone=<zone>] --list-ports*
NOTA: This command will only give you a list of ports that have been opened as ports: you will not be able to see any open ports that have been opened as a service. so you should consider using the *--list-all* option instead of *--list-ports.*

*To query if a port and protocol combination in enabled in a zone:
sudo firewall-cmd [--zone=<zone>] --query-port=<n>[-<m>]/<protocol>

*To quickly create a service configuration file skeleton in "/etc/firewalld" folder ready to be filled:
sudo firewall-cmd --permanent –new-service=haproxy

Zones

Una "zona" a Firewalld és un conjunt predefinit de regles que s'apliquen o bé al tràfic d'entrada provinent de determinades IPs remotes o bé a una determinada interfície. Aquestes regles bàsicament consisteixen en l'addició o supressió de determinats serveis segons la zona en qüestió. Les zones permeten així mostrar "cares" diferents del nostre servidor segons la xarxa a la què està exposat, definint el "nivell de confiança" del trànsit rebut des de l'origen i/o cap a la interfície en qüestió. O dit d'una altra manera: "you can use zones to manage incoming traffic based on its source; that enables you to sort incoming traffic and route it through different zones to allow or disallow services that can be reached by that traffic".

As each zone has its own configuration to accept or deny packets based on specified criteria, the general design pattern for multi-zoned firewalld configurations is to create a privileged source zone to allow specific IP's access to system services and a restrictive interface zone to limit the access of everyone else.

The predefined zones are stored in the "/usr/lib/firewalld/zones" directory and can be instantly applied. These files are copied to the "/etc/firewalld/zones" directory only after they are modified. The following table describes the default settings of the predefined zones (some of them are immutable, which means they are not customizable and there's no way to overload them), ordered from most restrictive to least:

***drop** (immutable) : Any incoming network packets are dropped, there is no reply. Only outgoing network connections are possible.
***block** (immutable): Any incoming network connections are rejected with an "icmp-host-prohibite" for IPv4 and "icmp6-adm-prohibited" for IPv6 message. Only network connections initiated within this system are possible.

*__public__ : For use in public areas where you do not trust other computers on the network. Only selected incoming connections are accepted.
*__external :__ For use on external networks with masquerading enabled (especially for routers). 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 where 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 where 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 where you mostly trust the other computers on the networks to not harm your computer. Only selected incoming connections are accepted.
*__trusted__ (immutable) : All network connections are accepted.

One of these zones is set as the *default* zone. On installation, the *default* zone is set to be the "public" zone. When interface connections are added to NetworkManager, they are assigned to the *default* zone. The *default* zone can be changed. Commands to get/set general information about zones are:

*To get a list of all supported (and permanent, if so) zones: *firewall-cmd [--permanent] --get-zones*
*To get the default zone: *firewall-cmd –get-default-zone*
*To get the active ("assigned to input traffic/interface") zones: *firewall-cmd –get-active-zones*
*To get the zone related to an interface: *firewall-cmd --get-zone-of-interface=<interface>*
*To get the list of all zones with its features (interfaces, services, protocols, sources, icmp-blocks...): *firewall-cmd --list-all-zones* To list these enabled features only from an specific zone, you can do *firewall-cmd [--zone=<zone>] --list-all* instead (if zone isn't specified, *default* one is understood) or *firewall-cmd --info-zone=<zone>* , too.

*To set the default zone: *sudo firewall-cmd --set-default-zone=<zone>* All interfaces that are located in the default zone will be pushed in the new default zone. Active connections are not affected. This command doesn't need the --*permanent* option to be permanent.
NOTA: This information is stored in the "/etc/firewalld/firewalld.conf" file.

*To quickly create a zone configuration file skeleton in "/etc/firewalld" folder ready to be filled: *sudo firewall-cmd --permanent –new-zone=myzone*

To fill and play with content an specific zone, you can do this:

*To change the zone a traffic source's IP belongs to (if the zone is omitted, the default one will be used): *sudo firewall-cmd --zone=<zone> --change-source=<x.x.x.x> [--permanent]*
NOTA: The IP <x.x.x.x> can be a network one if you specify the mask (for instance, this one: 192.168.1.0/24). You can also specify a MAC address, instead.
NOTA: There's also the *--add-source-port* to specify remote source ports to add to a zone but we won't use it
NOTA: To get the defined sources, execute *firewall-cmd --zone=<zone> --list-sources*
NOTA: To allow incoming traffic to be accepted by a zone based on the protocol (where you will be able to apply further rules and filtering if necessary) you could do: *sudo firewall-cmd --zone=<zone> --add-protocol={tcp|udp}*

*To change the zone an interface belongs to (if the zone is omitted, the default one will be used): *sudo firewall-cmd --zone=<zone> --change-interface=<interface> [--permanent]*
NOTA: You could use NetworkManager for this too: *nmcli c mod <interface> connection.zone <zone>*

*To query if an interface is in a zone (if the zone is omitted, the default one will be used): *sudo firewall-cmd --zone=<zone> --query-interface=<interface>*

It is important to understand that sources and interfaces are just a mean to decide what zone will the packet be sorted into. Both sources and interfaces do not decide whether to filter or allow a package

For every zone, you can set a default behavior that handles incoming traffic that is not further specified. Such behaviour is defined by setting the target of the zone. There are three options: default, ACCEPT, REJECT, and DROP. By setting the target to ACCEPT, you accept all incoming packets except those disabled by a specific rule. If you set the target to REJECT or DROP, you disable all incoming packets except those that you have allowed in specific rules. When packets are rejected, the source machine is informed about the rejection, while there is no information sent when the packets are dropped. To set a target for a zone, execute *sudo firewall-cmd --zone=<zone> --set-target=<default|ACCEPT|REJECT|DROP>* . To know the current target for a zone, execute *sudo firewall-cmd --zone=<zone> --get-target --permanent*

*Example 1: To allow traffic from a specific network to use a service on a machine, do this:

1. Add the source to the trusted zone to route the traffic originating from the source through the zone:
   sudo firewall-cmd --zone=trusted --add-source=192.168.1.0/24
2. Add the *http* service in the trusted zone:
   sudo firewall-cmd --zone=trusted -add-service=http
3. Make the new settings persistent:
   sudo firewall-cmd --runtime-to-permanent
4. Check that the trusted zone is active and that the service is allowed in it:
   sudo firewall-cmd --zone=trusted --list-all

*Example 2: Let's suppose someone from 3.3.3.3 is trolling your website. To restrict access for that IP, simply add it to the preconfigured drop zone, like this: *sudo firewall-cmd –zone=drop –add-source=3.3.3.3*

NOTA: You can also enable "panic mode" to block all network traffic in case of emergency, such as a system attack, by executing this: *sudo firewall-cmd --panic-on* . To disable this panic mode, execute *sudo firewall-cmd --panic-off* To know if your system is in "panic mode" or not, execute *firewall-cmd --query-panic* This mode is only a runtime kind of change

*Example 3: If you want to deal with several zone, you can do this, for instance...:

```
firewall-cmd --zone=public --add-service=http --permanent
firewall-cmd --zone=public --remove-service=ssh --permanent
firewall-cmd --zone=internal –add-source=1.1.1.1 --permanent
firewall-cmd --reload
```

...if someone attemps to ssh from 1.1.1.1, succees because source zone is applied first. If someone attemps to ssh from somewhere else, there wolud'nt be a source zone therefore the request would pass directly to the interface zone (public) which does not explicitly handle ssh: since publics's target is default, the request passes to the firewalld default action, which is to reject it. If 1.1.1.1 attemps http access the source zone (internal) doesn't allow it but the target is default, so the request passes to the interface zone (public) which grants access.

Zones can also be created using a zone configuration file. This approach can be helpful when you need to create a new zone, but want to reuse the settings from a different zone and only alter them a little. This file has information about the zone description, services, ports, protocols, icmp-blocks, masquerade, forward-ports and rich language rules, all in an XML file format. The file name has to be *zone-name*.xml and can be located in the "/usr/lib/firewalld/zones" or "/etc/firewalld/zones" directories. The following example of a zone configuration file shows how to allow one service (SSH) and one port range, for both the TCP and UDP protocols (for more information, see the "firewalld.zones" manual page):

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
  <short>My zone</short>
  <description>Here you can describe the characteristic features of the zone.</description>
  <service name="ssh"/>
  <port port="1025-65535" protocol="tcp"/>
  <port port="1025-65535" protocol="udp"/>
</zone>
```

"Masquerading" i "port forwarding"

*To enable masquerading (we suppose IP-Forwarding is already on) you only have to put an interface into "internal" zone and the other into "external" zone, that's is: Firewalld will establish masquerading automatically, mapping the addresses of a private network to a public IP address. But i f you wanted to enable it manually, you can do:
sudo firewall-cmd --zone=external --add-masquerade [--permanent]

*To disable masquerading:
sudo firewall-cmd --zone=external --remove-masquerade [--permanent]

*To check if masquerading in enabled (it shows "yes" or "no" on standard output):
firewall-cmd --zone=external --query-masquerade

Using firewalld masquerading, you can also set up ports redirection so that any incoming traffic that reaches a certain port on your system is delivered to another internal port of your choice or to an external port on another machine:

*To enable port forwarding or port mapping in a zone (once masquerading already is on zone):
sudo firewall-cmd [--zone=<zone>] --add-forward-port=
port=<port>[-<port>]:proto=<protocol>
{:toport=<port>[-<port>]
|:toport=<port>[-<port>]:toaddr=<address>
|:toaddr=<address> }

The port is either mapped to the same port on another host or to another port on the same host or to another port on another host. For example, to forward ssh in the home zone to host 127.0.0.2, do:
sudo firewall-cmd --zone=home –add-forward-port=port=22:proto=tcp:toaddr=127.0.0.2
To forward ssh in the home zone to host 127.0.0.2 listening on 2222 port, do this instead:
sudo firewall-cmd --zone=home –add-forward-port=port=22:proto=tcp:toport=2222:toaddr=127.0.0.2
To simply redirect ssh traffic to local 2222 port do this instead:
sudo firewall-cmd --zone=home –add-forward-port=port=22:proto=tcp:toport=2222

*To disable port forwarding or port mapping in a zone:
sudo firewall-cmd [--zone=<zone>] --remove-forward-port=
port=<port>[-<port>]:proto=<protocol>
{:toport=<port>[-<port>]
|:toport=<port>[-<port>]:toaddr=<address>
|:toaddr=<address> }

*To query port forwarding or port mapping in a zone:
firewall-cmd [--zone=<zone>] --query-forward-port=
port=<port>[-<port>]:proto=<protocol>
{:toport=<port>[-<port>]
|:toport=<port>[-<port>]:toaddr=<address>
|:toaddr=<address> }

ICMP blocks

*To enable ICMP blocks in a zone:
sudo firewall-cmd [--zone=<zone>] --add-icmp-block=<icmptype> [--permanent]
This enabled the block of a selected ICMP message, like "echo-reply" o "echo-request".
NOTA:More information in https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/7/html/security_guide/sec-managing_icmp_requests

*To disable ICMP blocks in a zone:
sudo firewall-cmd [--zone=<zone>] --remove-icmp-block=<icmptype>[--permanent]

*To query ICMP blocks in a zone (it shows "enabled" or "disabled" on standard output):
firewall-cmd [--zone=<zone>] --query-icmp-block=<icmptype>

Opcions "direct"

The "direct" options permit you to add custom rules. The --*direct* option needs to be the first option for all direct options and then you should write the --*passthrough ipv4* option, whose arguments are the same as the corresponding *iptables* arguments.

NOTA: Note "direct" rules are not saved and have to get resubmitted after reload or restart.
NOTA: There's to the –*passthrough ipv6* option and the –*passthrough eb* option, corresponding to ip6tables and ebtables options, respectively.

But you can manage basic iptables chains, tables and rules with specific arguments of firewall-cmd:

*To add a new chain <chain> to a table <table>:
sudo firewall-cmd --direct --add-chain {ipv4 |ipv6|eb} <table> <chain>
*To remove a chain with name <chain> from table <table>:
sudo firewall-cmd --direct --remove-chain { ipv4 | ipv6 | eb } <table> <chain>
*To query if a chain with name <chain> exists in table <table>. Returns "yes" if true, "no" otherwise:
firewall-cmd --direct --query-chain { ipv4 | ipv6 | eb } <table> <chain>
*To get all chains added to table <table> as a space separated list:
firewall-cmd --direct --get-chains { ipv4 | ipv6 | eb } <table>

*To add a rule with the arguments <args> to chain <chain> in table <table> with priority <priority>:
sudo firewall-cmd --direct --add-rule { ipv4 | ipv6 | eb } <table> <chain> <priority> <args>
*To remove a rule with the arguments <args> from chain <chain> in table <table>:
sudo firewall-cmd --direct --remove-rule { ipv4 | ipv6 | eb } <table> <chain> <args>
*To query if a rule with the arguments <args> exists in chain <chain> in table <table>. Returns "yes" if true, "no" otherwise:
firewall-cmd --direct --query-rule { ipv4 | ipv6 | eb } <table> <chain> <args>
*To get all rules added to chain <chain> in table <table> as a newline separated list of arguments:
firewall-cmd --direct --get-rules { ipv4 | ipv6 | eb } <table> <chain>
*To get all direct rules:
firewall-cmd --direct --get-all-rules