Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

Session - 35 Agenda:
1. Manage Firewall Using Ansible

Manage Firewall in Linux:
A firewall can be defined as a system of network security that controls and filters the traffic on the rule's predefined set. It is an intermediary system between the Internet and the device.

The kernel of Linux contains a subsystem, i.e., Netfilter. It is used for deciding or manipulating the network traffic fate headed through or into our server. All latest solutions of Linux firewall apply this system for a process which is known as "packet filtering".

firewalld is a firewall management tool for Linux operating systems. It provides firewall features by acting as a front-end for the Linux kernel's netfilter framework. firewalld's current default backend is nftables. Prior to v0.6.0, iptables was the default backend. Through its abstractions, firewalld acts as an alternative to nft and iptables command line programs. The name firewalld adheres to the Unix convention of naming system daemons by appending the letter "d".

firewalld is written in Python. It was intended to be ported to C++, but the porting project was abandoned in January 2015.

 # dnf repolist all
 # rpm -qa | grep firewalld
 # dnf remove firewalld -y
 # rpm -qa | grep firewalld
 # systemctl status firewalld
OR
 # firewall-cmd --state
 # dnf install firewalld -y
 # systemctl enable firewalld --now
 # systemctl status firewalld

We can see which zone is currently selected as the default by typing:
 # firewall-cmd --get-default-zone
 # firewall-cmd --list-all

Adding a Service to your Zones:
The most straight forward method is to add the services or ports you need to the zones you are using. You can get a list of the available service definitions with the --get-services option:
 # firewall-cmd --get-services

For instance, if we are running a web server serving conventional HTTP traffic, we can temporarily allow this traffic for interfaces in our public zone by typing:
 # firewall-cmd --zone=public --add-service=http

You can leave out the --zone= flag if you wish to modify the default zone. We can verify the operation was successful by using the --list-all or --list-services operations:
 # firewall-cmd --zone=public --list-services

To get a list of the available zones, type:

 # firewall-cmd --get-zones

Changing the Zone of an Interface:
You can move an interface between zones during a session by using the --zone= parameter in combination with the --change-interface= parameter. As with all commands that modify the firewall, you will need to use sudo.

For instance, we can move our eth0 interface to the home zone by typing this:
 # firewall-cmd --zone=home --change-interface=ens160

We can verify that this was successful by asking for the active zones again:
 # firewall-cmd --get-active-zones

Firewall script files location in RHEL 9 Linux.
 # cd /usr/lib/firewalld/
 # ls -l
 # cd services/
 # ls -l

Firewall is managed by files present in the /etc/firewalld directory.
 # cd /etc/firewalld/
 # ls -l

Services are collections of ports with an associated name and description. Using services is easier to administer than ports, but requires a bit of up-front work. The easiest way to start is to copy an existing script (found in /usr/lib/firewalld/services) to the /etc/firewalld/services directory where the firewall looks for non-standard definitions.

For instance, we could copy the SSH service definition to use for our example service definition like this. The filename minus the .xml suffix will dictate the name of the service within the firewall services list:
 # cp /usr/lib/firewalld/services/ssh.xml /etc/firewalld/services/example.xml

Now, you can adjust the definition found in the file you copied. First open it in your favorite text editor. We'll use vi here:
 # vim /etc/firewalld/services/example.xml

To start, the file will contain the SSH definition that you copied:
/etc/firewalld/services/example.xml

```
<?xml version="1.0" encoding="utf-8"?>
<service>
  <short>SSH</short>
  <description>Secure Shell (SSH) is a protocol for logging into and executing commands on remote machines. It provides secure encrypted communications. If you plan on accessing your machine remotely via SSH over a firewalled interface, enable this option. You need the openssh-server package installed for this option to be useful.</description>
  <port protocol="tcp" port="22"/>
</service>
```

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

The majority of this definition is actually metadata. You will want to change the short name for the service within the <short> tags. This is a human-readable name for your service. You should also add a description so that you have more information if you ever need to audit the service. The only configuration you need to make that actually affects the functionality of the service will likely be the port definition where you identify the port number and protocol you wish to open. Multiple <port/> tags can be specified.

For our example service, imagine that we need to open up port 7777 for TCP and 8888 for UDP. We can modify the existing definition with something like this:

/etc/firewalld/services/example.xml

```
<?xml version="1.0" encoding="utf-8"?>
<service>
 <short>example service</short>
 <description>This is just an example service. It probably shouldn't be used on a real system.</description>
 <port protocol="tcp" port="7777"/>
 <port protocol="udp" port="8888"/>
</service>
```

Save and close the file.

Reload your firewall to get access to your new service:

 # firewall-cmd --reload

You can see that it is now among the list of available services:

 # firewall-cmd --get-services

OR

 # firewall-cmd --get-services | grep -o example

Example of web server working & how firewall is important for it.

 # dnf install httpd -y
 # systemctl enable httpd --now
 # systemctl status httpd
 # cd /var/www/html
 # vim index.html
Nehra Classes Are Awesome !!!
 # curl localhost
 # ip a

Access webpage from another machine.

Open web browser and mention the web server IP Address. Firewall will not allow the traffic and we will not be able to see the web page, until we won't allow the http traffic in the firewall.

 # firewall-cmd --permanent --add-service=http
 # firewall-cmd --reload
 # firewall-cmd --list-all

Open web browser and mention the web server IP Address. This time firewall will allow the traffic and we will be able to see the web page.

 # firewall-cmd --permanent --remove-service=http

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

 # firewall-cmd --reload
 # firewall-cmd --list-all

Open web browser and mention the web server IP Address. Firewall will not allow the traffic and we will not be able to see the web page.

Add port for http in the firewall.
 # firewall-cmd --permanent --add-port=80/tcp
 # firewall-cmd --reload
 # firewall-cmd --list-all

Open web browser and mention the web server IP Address. This time firewall will allow the traffic and we will be able to see the web page.

To use firewall GUI utility.
 # dnf install -y firewall-config

Go to GUI and search for firewall, easily manage firewall from here.

Managing firewall zones.
 # firewall-cmd --permanent --zone=public --add-service=http
 # firewall-cmd --reload
 # firewall-cmd --list-all

Basic concepts of FirewallD:
firewalld simplifies the concepts of network traffic management. You have two main ideas as follows when it comes to firewalld on RHEL 9.

1. zones
Firewalld zones are nothing but predefined sets of rules. You can see all zones by running the following ls command:
 # ls -l /usr/lib/firewalld/zones/

Use the cat command to view drop zone:
 # cat /usr/lib/firewalld/zones/drop.xml

Understanding predefined zones:
block – All incoming network connections rejected. Only network connections initiated from within the system are possible.
dmz – Classic demilitarized zone (DMZ) zone that provided limited access to your LAN and only allows selected incoming ports.
drop – All incoming network connections dropped, and only outgoing network connections allowed.
external – Useful for router type of connections. You need LAN and WAN interfaces too for masquerading (NAT) to work correctly.
home – Useful for home computers such as laptops and desktops within your LAN where you trust other computers. Allows only selected TCP/IP ports.
internal – For use on internal networks when you mostly trust the other servers or computers on the LAN.
public – You do not trust any other computers and servers on the network. You only allow the required ports and services. For cloud servers or server hosted at your place always use public zone.
trusted – All network connections are accepted. I do not recommend this zone for dedicated servers or VMs connected to WAN.
work – For use at your workplace where you trust your coworkers and other servers.

Simply run the following command to see all zones:
 # firewall-cmd --get-zones

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

How to find out your default zone:
One can assign network interface and source to a zone. One of these zones set as the default zone. To get your default zone run:
 # firewall-cmd --get-default-zone

To see your network interface names run either ip command or nmcli command:
 # ip link show
OR
 # nmcli device status

When new interface connection added (such as eth0 or ens3) to NetworkManager, they are attached to the default zone. Verify it by running the following command:
 # firewall-cmd --get-active-zones

2. services
A service is nothing but a list of local ports, protocols, source ports, destinations, and firewall helper modules. Some examples:
Port – 80
Service – SSH
Protocols – ICMP
How to see firewall rules or services associated with the public zone Run:
 # firewall-cmd --list-all
OR
 # firewall-cmd --list-all --zone=public

List only services:
 # firewall-cmd --list-services

List only ports:
 # firewall-cmd --list-ports

How to see which services are allowed in the current zone
 # firewall-cmd --list-services
OR
 # firewall-cmd --list-services --zone=public
 # firewall-cmd --list-services --zone=home

How to find of list of services supported by firewalld
 # firewall-cmd --get-services
 # firewall-cmd --get-services | grep -o mysql
 # ls -l /usr/lib/firewalld/services/
 # cat /usr/lib/firewalld/services/dns.xml

Firewalld rule sets examples:
How to add a service to your zone. Add dns service (TCP/UDP port 53):
 # firewall-cmd --zone=public --add-service=dns --permanent

How to remove (delete) service from your zone. Delete vnc server service (TCP port range 5900-5903):
 # firewall-cmd --zone=public --remove-service=vnc-server --permanent

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

How to allow/open TCP/UDP port/protocol. Open TCP port # 9009:
 # firewall-cmd --zone=public --add-port=9009/tcp --permanent

To view added ports, run:
$ firewall-cmd --zone=internal --list-ports

How to deny/block TCP/UDP port/protocol. Open TCP port # 23:
 # firewall-cmd --zone=public --remove-port=23/tcp --permanent

How to write port forwarding firewalld rule. Forward TCP port 443 to 8080 on the same server:
 # firewall-cmd --zone=public --add-forward-port=port=80:proto=tcp:toport=8080 --permanent
 # firewall-cmd --reload

Verify port forwarding.
 # firewall-cmd --list-forward-ports

To delete above port forwarding, run
 # firewall-cmd --zone=public --remove-forward-port=port=80:proto=tcp:toport=8080
 # firewall-cmd --reload

Turn on masquerading if you need to forward traffic (port 443) to lxd server/container hosted at 192.168.2.42 port 443:
 # firewall-cmd --zone=public --add-masquerade
 # firewall-cmd --zone=public --add-forward-port=port=443:proto=tcp:toport=443:toaddr=192.168.2.42 --permanent
 # firewall-cmd --reload

Verify it.
 # firewall-cmd --list-all --zone=public

To delete above masquerading rules, run:
 # firewall-cmd --zone=public --remove-masquerade
 # firewall-cmd --zone=public --remove-forward-port=port=443:proto=tcp:toport=443:toaddr=192.168.2.42 --permanent
 # firewall-cmd --reload

As usual use the following to list rules:
 # firewall-cmd --zone=public --list-all --permanent

Rich rule example:
Say you want to allow access to SSH port 22 only from 10.8.0.8 IP address, run:
 # firewall-cmd --permanent --zone=public --add-rich-rule 'rule family="ipv4" source address="10.8.0.8" port port=22 protocol=tcp accept'
 # firewall-cmd --reload

To verify new rules, run:
 # firewall-cmd --list-rich-rules --permanent

You can delete rich rules as follows:
 # firewall-cmd --remove-rich-rule 'rule family="ipv4" source address="10.8.0.8" port port=22 protocol=tcp accept' --permanent
 # firewall-cmd --remove-rich-rule 'rule family="ipv4" source address="192.168.1.0/24" port port="11211" protocol="tcp" accept' --permanent
 # firewall-cmd --reload

Again verify it:
 # firewall-cmd --list-rich-rules --permanent

Creating a New Zone:
An entirely new zone may be created by running the following command. Once created, the zone may be managed in the same way as any of the predefined zones:
 # firewall-cmd --get-zones
 # firewall-cmd --permanent --new-zone=myoffice

After adding a new zone, firewalld will need to be restarted before the zone becomes available:
 # firewall-cmd --reload

Verify it.
 # firewall-cmd --get-zones
OR
 # firewall-cmd --get-zones | grep -o myoffice

Changing Zone/Interface Assignments:
As previously discussed, each interface on the system must be assigned to a zone. The zone to which an interface is assigned can also be changed using the firewall-cmd tool. In the following example, the eth0 interface is assigned to the public zone:
 # firewall-cmd --zone=public --change-interface=eth0

Disable firewall.
 # systemctl stop firewalld
 # systemctl disable firewalld
 # systemctl status firewalld

=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=
=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=x=

Manage Firewall Using Ansible:
There are two ways of managing the firewall using Ansible on the managed nodes:
1. Ansible Ad-Hoc Commands
2. Ansible Playbooks

1. Ansible Ad-Hoc Commands:
We can execute ansible ad-hoc command on the managed nodes using command & firewalld modules to manage the firewall rules.
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
$ ansible node1 -m command -a 'firewall-cmd --list-ports' -b
$ ansible node1 -m command -a 'firewall-cmd --list-services' -b
$ ansible node1 -m command -a 'firewall-cmd --get-zones' -b
$ ansible node1 -m command -a 'firewall-cmd --get-default-zone' -b
$ ansible node1 -m command -a 'firewall-cmd --permanent --add-service=http' -b

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

```
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-services' -b
$ ansible node1 -m command -a 'firewall-cmd --permanent --remove-service=http' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-services' -b
$ ansible node1 -m command -a 'firewall-cmd --permanent --add-port=80/tcp' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-ports' -b
$ ansible node1 -m command -a 'firewall-cmd --permanent --remove-port=80/tcp' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-ports' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=home' -b
$ ansible node1 -m command -a 'firewall-cmd --permanent --add-service=ftp --zone=home' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=home' -b
$ ansible-galaxy collection install ansible.posix
$ ansible node1 -m command -a 'firewall-cmd --list-services' -b
$ ansible node1 -m firewalld -a 'protocol=ospf permanent=true state=enabled' -b
$ ansible node1 -m command -a 'firewall-cmd --list-services' -b
$ ansible node1 -m command -a 'firewall-cmd --list-protocols' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-protocols' -b
$ ansible node1 -m firewalld -a 'protocol=ospf permanent=true state=disabled' -b
$ ansible node1 -m command -a 'firewall-cmd --list-protocols' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-protocols' -b
$ ansible node1 -m command -a 'firewall-cmd --list-services' -b
$ ansible node1 -m firewalld -a 'service=https permanent=true state=enabled immediate=yes' -b
$ ansible node1 -m command -a 'firewall-cmd --list-services' -b
$ ansible node1 -m firewalld -a 'service=https permanent=true state=disabled immediate=yes' -b
$ ansible node1 -m command -a 'firewall-cmd --list-services' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
$ ansible node1 -m firewalld -a 'port=443/tcp permanent=true state=enabled immediate=yes' -b
$ ansible node1 -m command -a 'firewall-cmd --list-ports' -b
$ ansible node1 -m firewalld -a 'port=443/tcp permanent=true state=disabled immediate=yes' -b
$ ansible node1 -m command -a 'firewall-cmd --list-ports' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=drop' -b
$ ansible node1 -m firewalld -a 'zone=drop state=enabled permanent=true icmp_block_inversion=true immediate=yes' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=drop' -b
$ ansible node1 -m firewalld -a 'zone=drop state=enabled permanent=true icmp_block_inversion=false immediate=yes' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=drop' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
$ ansible node1 -m firewalld -a 'rich_rule="rule family=ipv4 forward-port port=443 protocol=tcp to-port=8443" zone=public
permanent=true state=enabled immediate=yes' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
```

2. Ansible Playbooks:

We can write Ansible playbooks using firewalld module to manage the firewall rules on the managed nodes.

$ vim firewall.yml

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

```
---
- name: Manage Firewall
  hosts: node1
  become: true
  tasks:
    - name: permit traffic in default zone for https service
      ansible.posix.firewalld:
        service: https
        permanent: true
        state: enabled
...

$ ansible-playbook firewall.yml
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b

$ vim firewall2.yml
---
- name: Manage Firewall
  hosts: node1
  become: true
  tasks:
    - name: permit ospf traffic
      ansible.posix.firewalld:
        protocol: ospf
        permanent: true
        state: enabled
...

$ ansible-playbook firewall2.yml
$ ansible node1 -m command -a 'firewall-cmd --list-protocols' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-protocols' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b

$ vim firewall3.yml
---
- name: Manage Firewall
  hosts: node1
  become: true
  tasks:
    - name: do not permit traffic in default zone on port 8081/tcp
      ansible.posix.firewalld:
        port: 8081/tcp
        permanent: true
        state: disabled
...

$ ansible-playbook firewall3.yml
```

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

```
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b

$ vim firewall4.yml
---
- name: Manage Firewall
  hosts: node1
  become: true
  tasks:
    - name: Allow traffic from UDP ports 161 & 162 in the firewall.
      ansible.posix.firewalld:
        port: 161-162/udp
        permanent: true
        state: enabled
        immediate: true

    - name: Allow the http traffic in the DMZ zone.
      ansible.posix.firewalld:
        zone: dmz
        service: http
        permanent: true
        state: enabled
        immediate: true
...

$ ansible-playbook firewall4.yml
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=dmz' -b

$ vim firewall5.yml
---
- name: Manage Firewall
  hosts: node1
  become: true
  tasks:
    - ansible.posix.firewalld:
        rich_rule: rule service name="ftp" audit limit value="1/m" accept
        permanent: true
        state: enabled
        immediate: true

    - ansible.posix.firewalld:
        zone: trusted
        interface: ens160
        permanent: true
        state: enabled
        immediate: true
...

$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
```

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

```
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=trusted' -b
$ ansible-playbook firewall5.yml
$ ansible node1 -m command -a 'firewall-cmd --list-all' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=trusted' -b

$ vim firewall6.yml
---
- name: Manage Firewall
  hosts: node1
  become: true
  tasks:
    - ansible.posix.firewalld:
        masquerade: true
        state: enabled
        permanent: true
        zone: dmz
        immediate: true

    - ansible.posix.firewalld:
        zone: custom
        state: present
        permanent: true
...

$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=dmz' -b
$ ansible node1 -m command -a 'firewall-cmd --get-zones' -b
$ ansible-playbook firewall6.yml
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=dmz' -b
$ ansible node1 -m command -a 'firewall-cmd --get-zones' -b
$ ansible node1 -m command -a 'firewall-cmd --reload' -b
$ ansible node1 -m command -a 'firewall-cmd --get-zones' -b

$ vim firewall7.yml
---
- name: Manage Firewall
  hosts: node1
  become: true
  tasks:
    - ansible.posix.firewalld:
        zone: public
        state: enabled
        permanent: true
        icmp_block_inversion: true
        immediate: true

    - ansible.posix.firewalld:
        zone: drop
        state: enabled
        permanent: true
        icmp_block: echo-request
```

Manage Firewall Using Ansible

By: Er. Vikas Nehra (M. Tech, B. Tech), Experience: 15 + Years

```
    immediate: true

  - name: Redirect port 443 to 8443 with Rich Rule
    ansible.posix.firewalld:
      rich_rule: rule family=ipv4 forward-port port=443 protocol=tcp to-port=8443
      zone: public
      permanent: true
      immediate: true
      state: enabled
...
```

```
$ ping node1
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=public' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=drop' -b
$ ansible-playbook firewall7.yml
$ ping node1
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=public' -b
$ ansible node1 -m command -a 'firewall-cmd --list-all --zone=drop' -b
```

Thank You