

Task 4: Firewall setup and usage on CENTOS

OBJECTIVE

To configure and test basic firewall rules to allow or block network traffic using UFW (Uncomplicated Firewall) on CentOS.

For today's task I am using the following:

- CentOS Virtual Machine
- UFW (Uncomplicated Firewall)

1. Installing UFW on CentOS

Command

“sudo yum install epel-release”

```
reyanshelocalhost:~$ sudo yum install epel-release
Last metadata expiration check: 0:11:45 ago on Fri 27 Jun 2025 07:22:54 PM IST.
Dependencies resolved.
=====
Package                Architecture Version      Repository      Size
=====
Installing:
epel-release            noarch      10-5.el10s    extras-common    18 k
Transaction Summary
=====
Install 1 Package

Total download size: 18 k
Installed size: 25 k
Is this ok [y/N]: y
Downloading Packages:
epel-release-10-5.el10s.noarch.rpm                28 kB/s | 18 kB    00:00
-----
Total                                              16 kB/s | 18 kB    00:01
CentOS Stream 10 - Extras packages                2.1 MB/s | 2.1 kB    00:00
Importing GPG key 0x1D997668:
  Userid      : "CentOS Extras SIG (https://wiki.centos.org/SpecialInterestGroup) <security@centos.org>"
  Fingerprint: 363F C097 2F64 B699 AED3 968E 1FF6 A217 1D99 7668
  From        : /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-SIG-Extras-SHA512
Is this ok [y/N]: y
warning: Certificate 1FF6A2171D997668:
Policy rejects subkey 885C8111FCA5D0FF: Policy rejected non-revocation signature (PrimaryKeyBinding) requiring second pre-image resistance
```

“sudo yum install ufw -y”

```
Complete!
reyansh@localhost:~$ sudo yum install ufw -y
[sudo] password for reyansh:
Last metadata expiration check: 0:12:31 ago on Fri 27 Jun 2025 07:34:58 PM IST.
Dependencies resolved.
=====
Package                Architecture          Version               Repository            Size
=====
Installing:
ufw                    noarch                0.35-35.el10_1       epel                   250 k
=====
Transaction Summary
=====
Install 1 Package

Total download size: 250 k
Installed size: 991 k
Downloading Packages:
ufw-0.35-35.el10_1.noarch.rpm
-----
168 kB/s | 250 kB | 00:01
-----
Total
74 kB/s | 250 kB | 00:03
Extra Packages for Enterprise Linux 10 - x86_64
1.6 MB/s | 1.6 kB | 00:00
Importing GPG key 0xE37ED158:
  Userid : "Fedora (epel10) <epel@fedoraproject.org>"
  Fingerprint: 7080 1568 5C45 6268 8501 F826 3300 8517 F37E D158
```

Explanation

- "epel-release" gives access to extra software packages that are not available in the default CentOS repository.
- "ufw" is to install the firewall tool that we are installing for managing firewall rules easily.

2. Enabling and Starting UFW service

Command

“sudo systemctl enable ufw”

”sudo systemctl start ufw”

```
1.6 MB/s | 1.6 kB    00:00
Importing GPG key 0xE37ED158:
  Userid      : "Fedora (epel10) <epel@fedoraproject.org>"
  Fingerprint: 7D8D 15C8 FC4E 6268 8591 FB26 33D9 8517 E37E D158
  From        : /etc/pki/rpm-gpg/RPM-GPG-KEY-EPEL-10
Key imported successfully
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      :                                1/1
  Installing     : ufw-0.35-35.el10_1.noarch      1/1
  Running scriptlet: ufw-0.35-35.el10_1.noarch    1/1

Installed:
  ufw-0.35-35.el10_1.noarch

Complete!
reyansh@localhost:~$ sudo systemctl enable ufw
Created symlink '/etc/systemd/system/basic.target.wants/ufw.service' → '/usr/lib/systemd/system/ufw.service'.
reyansh@localhost:~$ sudo systemctl start ufw
reyansh@localhost:~$
```

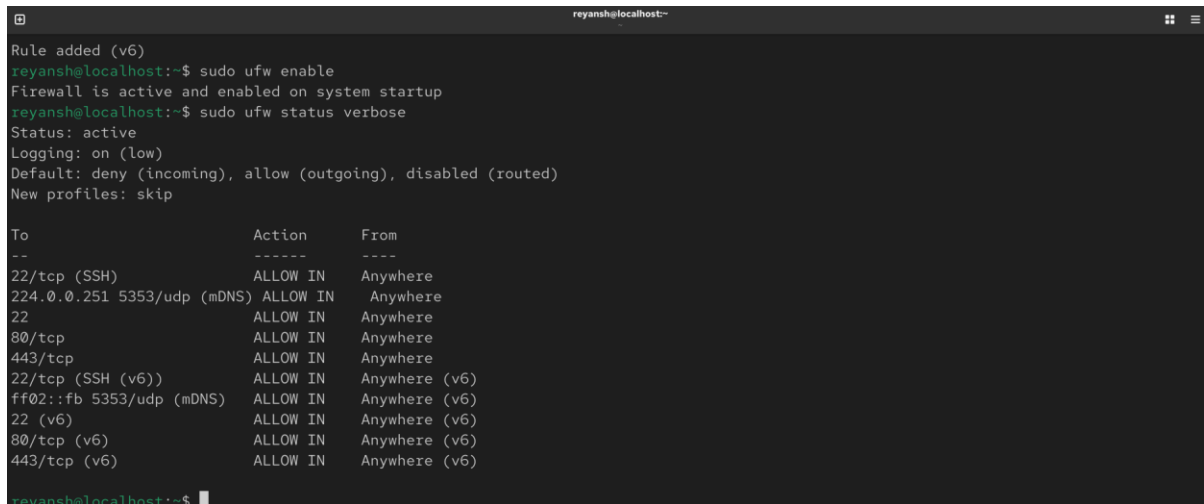
Explanation

- "systemctl enable ufw" makes sure UFW will start automatically every time the system boots
- "systemctl start ufw" starts the UFW service right away

3. Checking UFW Status and Rules

Command

“sudo ufw status verbose”

A terminal window titled 'reyansh@localhost' showing the execution of 'sudo ufw status verbose'. The output indicates that the firewall is active and shows a list of rules. The rules table has three columns: 'To', 'Action', and 'From'.

```
Rule added (v6)
reyansh@localhost:~$ sudo ufw enable
Firewall is active and enabled on system startup
reyansh@localhost:~$ sudo ufw status verbose
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To                Action            From
--                -
22/tcp (SSH)      ALLOW IN         Anywhere
224.0.0.251 5353/udp (mDNS) ALLOW IN         Anywhere
22                ALLOW IN         Anywhere
80/tcp            ALLOW IN         Anywhere
443/tcp           ALLOW IN         Anywhere
22/tcp (SSH (v6)) ALLOW IN         Anywhere (v6)
ff02::fb 5353/udp (mDNS) ALLOW IN         Anywhere (v6)
22 (v6)           ALLOW IN         Anywhere (v6)
80/tcp (v6)       ALLOW IN         Anywhere (v6)
443/tcp (v6)      ALLOW IN         Anywhere (v6)
reyansh@localhost:~$
```

Explanation

- This shows all the active firewall rules in a clear format, along with details like default policies and whether logging is enabled.

4. Setting UFW to Deny All Incoming and allowing all outgoing by Default

Command

“sudo ufw default deny incoming”

“sudo ufw default allow outgoing”

```
reyansh@localhost:~$ sudo ufw default deny incoming
Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)
reyansh@localhost:~$ sudo ufw default allow outgoing
Default outgoing policy changed to 'allow'
(be sure to update your rules accordingly)
reyansh@localhost:~$
```

Explanation

- Denying incoming means all connections coming into the machine are blocked by default, which adds more security
- Allowing outgoing means the machine can still connect to the internet or other external services for updates and access

5. Allowing essential ports

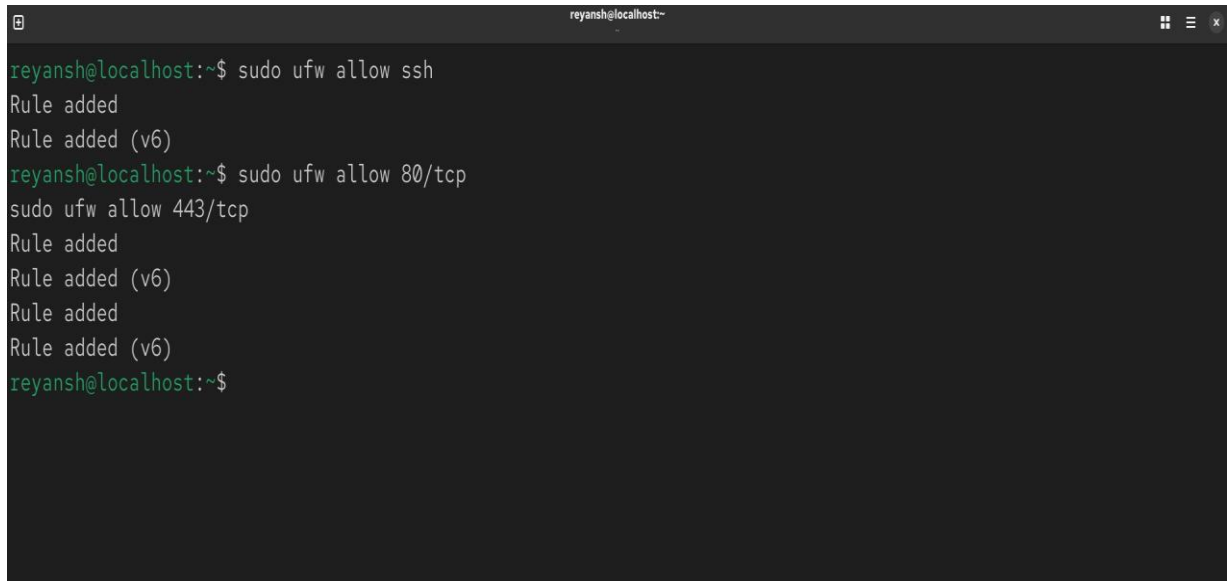
Command

“sudo ufw allow ssh”- for remote access

```
reyansh@localhost:~$ sudo ufw allow ssh
Rule added
Rule added (v6)
reyansh@localhost:~$
```

“sudo ufw allow 80/tcp”- for http service

“sudo ufw allow 443/tcp”- for https service

A terminal window with a dark background and light green text. The window title is 'reyansh@localhost:~'. The terminal shows the following commands and output: 'sudo ufw allow ssh' followed by 'Rule added' and 'Rule added (v6)'; 'sudo ufw allow 80/tcp' followed by 'Rule added'; 'sudo ufw allow 443/tcp' followed by 'Rule added', 'Rule added (v6)', 'Rule added', and 'Rule added (v6)'. The prompt 'reyansh@localhost:~\$' is shown at the end.

```
reyansh@localhost:~$ sudo ufw allow ssh
Rule added
Rule added (v6)
reyansh@localhost:~$ sudo ufw allow 80/tcp
Rule added
reyansh@localhost:~$ sudo ufw allow 443/tcp
Rule added
Rule added (v6)
Rule added
Rule added (v6)
reyansh@localhost:~$
```

Explanation

- This allows port 22, which is used for SSH access
- If we are connected through SSH, blocking this port would disconnect us and lock us out of the machine
- Port 80 is used for regular web traffic (HTTP)
- Port 443 is used for secure web traffic (HTTPS)
- Allowing these ports lets web traffic come into the machine

6. Adding a Rule to Block (Telnet – Port 23) Command

“sudo ufw deny 23”

```
reyansh@localhost:~$ sudo ufw status
Status: active
Logging: on (low)
Default: deny (incoming), allow (outgoing), disabled (routed)
New profiles: skip

To Action From
--
22/tcp (SSH) ALLOW IN Anywhere
224.0.0.251 5353/udp (mDNS) ALLOW IN Anywhere
22 ALLOW IN Anywhere
80/tcp ALLOW IN Anywhere
443/tcp ALLOW IN Anywhere
22/tcp (SSH (v6)) ALLOW IN Anywhere (v6)
ff02::fb 5353/udp (mDNS) ALLOW IN Anywhere (v6)
22 (v6) ALLOW IN Anywhere (v6)
80/tcp (v6) ALLOW IN Anywhere (v6)
443/tcp (v6) ALLOW IN Anywhere (v6)

reyansh@localhost:~$ sudo ufw deny 23
[sudo] password for reyansh:
Rule added
Rule added (v6)
reyansh@localhost:~$
```

Explanation

- This blocks Telnet, which uses port 23 and is known to be insecure
- It shows how to block services that are not needed or may be risky

7. Enabling UFW

Command

“sudo ufw enable”

```
reyansh@localhost:~$ sudo ufw allow ssh
Rule added
Rule added (v6)
reyansh@localhost:~$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
reyansh@localhost:~$ sudo ufw enable
Firewall is active and enabled on system startup
reyansh@localhost:~$
```

Explanation

- This command turns on the firewall and applies all the rules we set earlier

8. Testing firewall rules

Test-1 Web Access (HTTP and HTTPS)

Command

“curl http://<centos-ip>”

“curl https://<centos-ip>”

A terminal window on a Kali Linux machine with a keyboard background. The user runs two curl commands. The first command, curl http://[redacted], fails with the message: 'curl: (7) Failed to connect to [redacted] port 80 after 2 ms: Could not connect to server'. The second command, curl https://[redacted], also fails with the message: 'curl: (7) Failed to connect to [redacted] port 443 after 0 ms: Could not connect to server'. The prompt returns to the user after each failure.

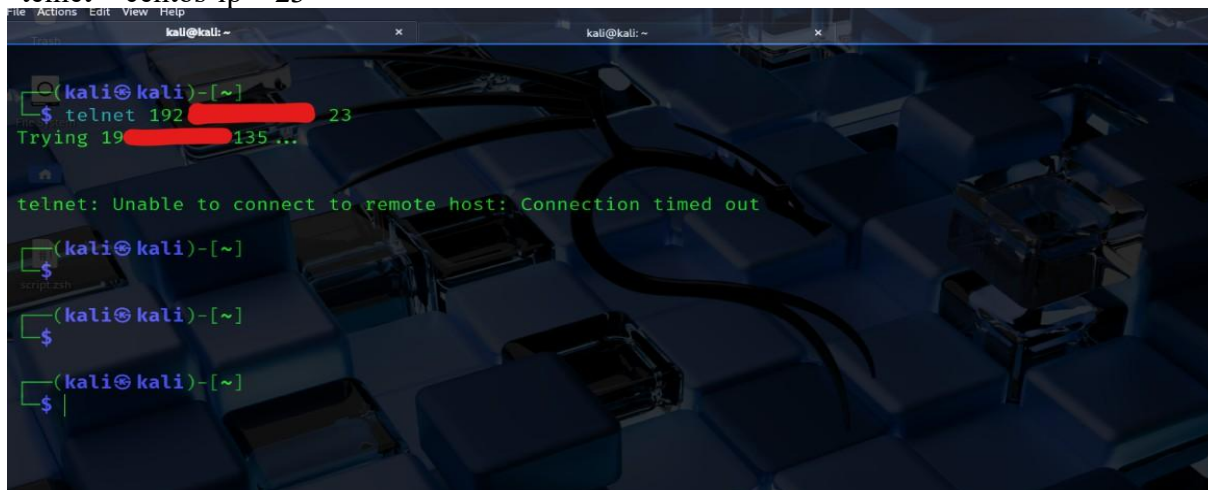
```
(kali@kali)-[~]
$ curl http://[redacted]
curl: (7) Failed to connect to [redacted] port 80 after 2 ms: Could not connect to server
(kali@kali)-[~]
$ curl https://[redacted]
curl: (7) Failed to connect to [redacted] port 443 after 0 ms: Could not connect to server
(kali@kali)-[~]
$
```

The result shows us that the firewall allowed the connection on ports 80 and 443, but the request failed because no web service was running on the CentOS machine to respond.

Test-2 Telnet Access (Port 23)

Command

“telnet <centos-ip> 23”

A terminal window on a Kali Linux machine with a keyboard background. The user runs the telnet command: telnet 192.[redacted].23. The terminal shows 'Trying 192.[redacted].135...' followed by the error message: 'telnet: Unable to connect to remote host: Connection timed out'. The prompt returns to the user.

```
(kali@kali)-[~]
$ telnet 192.[redacted].23
Trying 192.[redacted].135...
telnet: Unable to connect to remote host: Connection timed out
(kali@kali)-[~]
$
(kali@kali)-[~]
$
(kali@kali)-[~]
$
```

The Telnet connection on port 23 failed as expected because the firewall rule explicitly blocked incoming traffic on that port.

9. Testing Rules via Kali Linux

We are testing what ports are open using “nmap” from our kali machine.

Command (via Kali Linux)

nmap <centos>

A terminal window from a Kali Linux machine showing the output of an nmap scan. The background of the terminal is a dark blue image of a computer keyboard. The text is green on a black background. The scan results for three hosts are shown, with IP addresses redacted by red bars. The first host (192.168.1.1) has ports 3306/tcp (mysql) and 6646/tcp (unknown) open. The second host (192.168.1.2) has port 53/tcp (domain) filtered. The third host (192.168.1.35) has ports 22/tcp (ssh) open, and ports 80/tcp (http) and 443/tcp (https) closed. The fourth host (192.168.1.54) is ignored.

```
(kali㉿kali)-[~]
└─$ nmap 192.168.1.24
Starting Nmap 7.95 ( https://nmap.org ) at 2025-06-27 11:19 EDT
Stats: 0:00:01 elapsed; 0 hosts completed (0 up), 255 undergoing ARP Ping Scan
ARP Ping Scan Timing: About 50.78% done; ETC: 11:19 (0:00:01 remaining)
Nmap scan report for 192.168.1.1
Host is up (0.0010s latency).
Not shown: 998 filtered tcp ports (no-response)
PORT      STATE SERVICE
3306/tcp   open  mysql
6646/tcp   open  unknown
MAC Address: 00:0C:29:00:00:03 (VMware)

Nmap scan report for 192.168.1.2
Host is up (0.00081s latency).
Not shown: 999 closed tcp ports (reset)
PORT      STATE SERVICE
53/tcp     filtered domain
MAC Address: 00:0C:29:00:00:04 (VMware)

Nmap scan report for 192.168.1.35
Host is up (0.0013s latency).
Not shown: 997 filtered tcp ports (no-response)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    closed http
443/tcp   closed https
MAC Address: 00:0C:29:00:00:05 (VMware)

Nmap scan report for 192.168.1.54
Host is up (0.00087s latency).
All 1000 scanned ports on 192.168.1.54 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
```

Summary of Port Testing Results

- **SSH (port 22)** was allowed in the firewall and confirmed to be open and accessible, as the service (SSH daemon) was actively running.
- **HTTP (port 80)** and **HTTPS (port 443)** were also allowed through the firewall, but Nmap showed them as *closed* because no web server was running to listen on those ports.
- The firewall configuration was correct in all cases the test results reflect the availability of services, not firewall blocks.