

- **Task 22: Configure iptables/UFW**
→ Example: `sudo ufw allow 80` (open port 80), `sudo ufw deny 21` (block FTP).
- **Task 23: Block Ports & Verify**
→ Block ports and verify using Nmap that they are closed.
- **Task 24: Enable SSL/TLS**
→ Generate a self-signed certificate and enable HTTPS for Apache or Nginx.

Task 22: Configure iptables/UFW

→ Example: `sudo ufw allow 80` (open port 80), `sudo ufw deny 21` (block FTP).

Option 1: Using UFW (on CentOS)

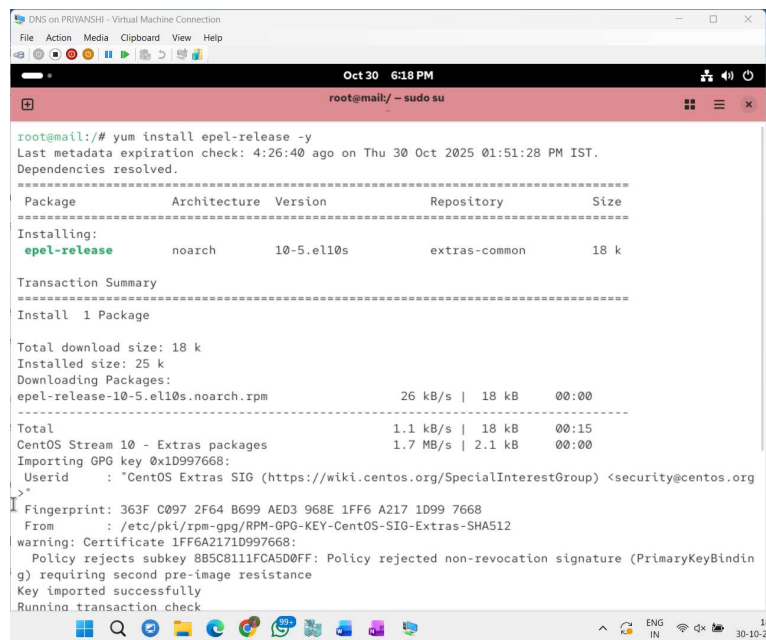
Step 1: Install and enable UFW

```
sudo yum install epel-release -y
```

```
sudo yum install ufw -y
```

```
sudo systemctl enable ufw
```

```
sudo systemctl start ufw
```

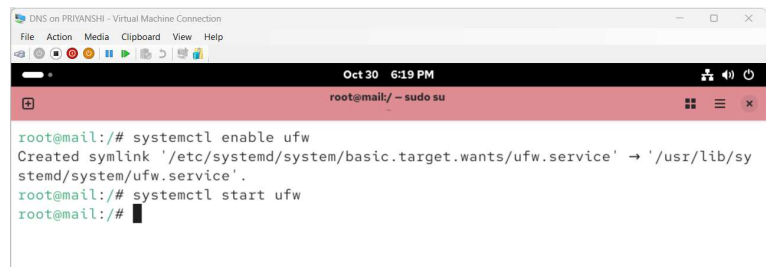


```

root@mail:/# yum install epel-release -y
Last metadata expiration check: 4:26:40 ago on Thu 30 Oct 2025 01:51:28 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
Downloading Packages:
epel-release-10-5.el10s.noarch.rpm                26 kB/s | 18 kB    00:00
-----
Total                                           1.1 kB/s | 18 kB    00:15
CentOS Stream 10 - Extras packages              1.7 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
warning: Certificate 1FF6A2171D997668:
Policy rejects subkey 8B5C8111FCA5D0FF: Policy rejected non-revocation signature (PrimaryKeyBindin
g) requiring second pre-image resistance
Key imported successfully
Running transaction check

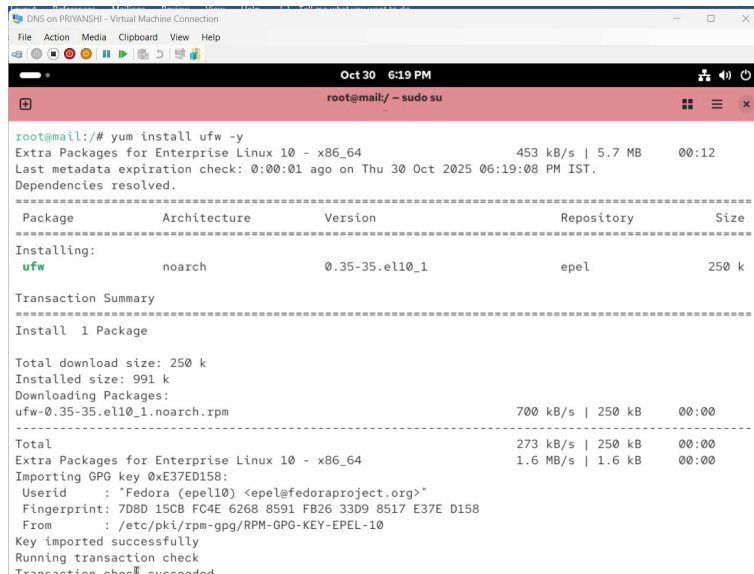
```



```

root@mail:/# systemctl enable ufw
Created symlink '/etc/systemd/system/basic.target.wants/ufw.service' → '/usr/lib/sy
stemd/system/ufw.service'.
root@mail:/# systemctl start ufw
root@mail:/#

```



```
root@mail:/# yum install ufw -y
Extra Packages for Enterprise Linux 10 - x86_64          453 kB/s | 5.7 MB    00:12
Last metadata expiration check: 0:00:01 ago on Thu 30 Oct 2025 06:19:08 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

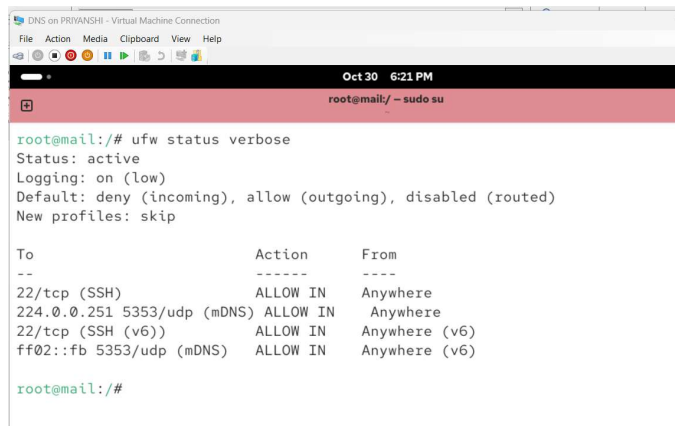
Total download size: 250 k
Installed size: 991 k
Downloading Packages:
ufw-0.35-35.el10_1.noarch.rpm                          700 kB/s | 250 kB    00:00
-----
Total                                                    273 kB/s | 250 kB    00:00
Extra Packages for Enterprise Linux 10 - x86_64          1.6 MB/s | 1.6 kB    00:00
Importing GPG key 0xE37ED158:
Userid   : "Fedora (epel10) <epel@fedoraproject.org>"
Fingerprint: 7D8D 15CB 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.
```

Step 2: Check status

`sudo ufw status verbose`

If it shows “inactive”, enable it:

`sudo ufw enable`



```
root@mail:/# 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/tcp (SSH (v6)) ALLOW IN Anywhere (v6)
ff02::fb 5353/udp (mDNS) ALLOW IN Anywhere (v6)

root@mail:/#
```

Step 3: Allow or deny ports

`sudo ufw allow 80` # Allow HTTP (port 80)

`sudo ufw allow 443` # Allow HTTPS (port 443)

`sudo ufw deny 21` # Block FTP (port 21)

`sudo ufw allow 22` # Allow SSH (port 22)

`sudo ufw allow 5666` # Allow Nagios NRPE communication

`sudo ufw allow proto icmp` # Allow ping

```
DNS on PRIVANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:22 PM
root@mail/~ sudo su

root@mail:/# ufw allow 80
Rule added
Rule added (v6)
root@mail:/# ufw allow 443
Rule added
Rule added (v6)
root@mail:/# ufw allow 21
Rule added
Rule added (v6)
root@mail:/# ufw allow 22
Rule added
Rule added (v6)
root@mail:/# ufw allow 5666
Rule added
Rule added (v6)
root@mail:/# ufw allow proto icmp
ERROR: Need 'to' or 'from' clause
root@mail:/#
```

Step 4: Verify rules

sudo ufw status numbered

```
DNS on PRIVANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:24 PM
root@mail/~ sudo su

root@mail:/# ufw status numbered
Status: active

      To Action From
      --
[ 1] SSH ALLOW IN Anywhere
[ 2] 224.0.0.251 mDNS ALLOW IN Anywhere
[ 3] 80 ALLOW IN Anywhere
[ 4] 443 ALLOW IN Anywhere
[ 5] 21 ALLOW IN Anywhere
[ 6] 22 ALLOW IN Anywhere
[ 7] 5666 ALLOW IN Anywhere
[ 8] SSH (v6) ALLOW IN Anywhere (v6)
[ 9] ff02::fb mDNS ALLOW IN Anywhere (v6)
[10] 80 (v6) ALLOW IN Anywhere (v6)
[11] 443 (v6) ALLOW IN Anywhere (v6)
[12] 21 (v6) ALLOW IN Anywhere (v6)
[13] 22 (v6) ALLOW IN Anywhere (v6)
[14] 5666 (v6) ALLOW IN Anywhere (v6)

root@mail:/#
```

Step 5: Delete a rule if needed

sudo ufw delete <rule_number>

```
DNS on PRIYANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:27 PM
root@mail:/ ~ sudo su

root@mail:/# ufw status numbered
Status: active

      To Action From
      --
[ 1] SSH ALLOW IN Anywhere
[ 2] 224.0.0.251 mDNS ALLOW IN Anywhere
[ 3] 80 ALLOW IN Anywhere
[ 4] 443 ALLOW IN Anywhere
[ 5] 21 ALLOW IN Anywhere
[ 6] 22 ALLOW IN Anywhere
[ 7] 5666 ALLOW IN Anywhere
[ 8] ff02::fb mDNS ALLOW IN Anywhere (v6)
[ 9] 80 (v6) ALLOW IN Anywhere (v6)
[10] 443 (v6) ALLOW IN Anywhere (v6)
[11] 21 (v6) ALLOW IN Anywhere (v6)
[12] 22 (v6) ALLOW IN Anywhere (v6)
[13] 5666 (v6) ALLOW IN Anywhere (v6)

root@mail:/# ufw delete 4
Deleting:
allow 443
Proceed with operation (y/n)? y
Rule deleted
root@mail:/# ufw delete 6
Deleting:
allow 5666
Proceed with operation (y/n)? y
Rule deleted
```

Step 6: Disable UFW if required

sudo ufw disable

```
DNS on PRIYANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:28 PM
root@mail:/ ~ sudo su

root@mail:/# ufw disable
Firewall stopped and disabled on system startup
root@mail:/#
```

Option 2: Using iptables (CentOS)

Step 1: Install and enable iptables

sudo yum install iptables-services -y

sudo systemctl enable iptables

sudo systemctl start iptables

```
DNS on PRIYANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:34 PM
root@mail:/ ~ sudo su

root@mail:/# yum install iptables-services -y
Last metadata expiration check: 0:15:20 ago on Thu 30 Oct 2025 06:19:08 PM IST.
Dependencies resolved.
=====
Package Arch Version Repository Size
=====
Installing:
iptables-nft-services noarch 1.8.11-11.el10 appstream 21 k

Transaction Summary
=====
Install 1 Package

Total download size: 21 k
Installed size: 30 k
Downloading Packages:
Waiting for process with pid 10953 to finish.
```

```
DNS on PRIVANSHI - Virtual Machine Connection
File Action Media Clipboard View Help

Oct 30 6:42 PM
root@mail/ ~ sudo su

=====
Package Arch Version Repository Size
=====
Installing:
iptables-nft-services noarch 1.8.11-11.el10 appstream 21 k

Transaction Summary
=====
Install 1 Package

Total download size: 21 k
Installed size: 30 k
Downloading Packages:
iptables-nft-services-1.8.11-11.el10.noarch.rpm 20 kB/s | 21 kB 00:01
-----
Total 1.7 kB/s | 21 kB 00:12

Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing : 1/1
Installing : iptables-nft-services-1.8.11-11.el10.noarch 1/1
Running scriptlet: iptables-nft-services-1.8.11-11.el10.noarch 1/1

Installed:
iptables-nft-services-1.8.11-11.el10.noarch

Complete!
```

```
DNS on PRIVANSHI - Virtual Machine Connection
File Action Media Clipboard View Help

Oct 30 6:43 PM
root@mail/ ~ sudo su

root@mail:/# systemctl enable iptables.service
Created symlink '/etc/systemd/system/multi-user.target.wants/iptables.service' -> '/usr/lib/systemd/system/iptables.service'.
root@mail:/# systemctl start iptables.service
root@mail:/#
```

Step 2: Check current rules

`sudo iptables -L -n -v`

This shows all currently active firewall rules.

```
DNS on PRIVANSHI - Virtual Machine Connection
File Action Media Clipboard View Help

Oct 30 6:44 PM
root@mail/ ~ sudo su

root@mail:/# iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination state
4 304 ACCEPT all -- * * 0.0.0.0/0 0.0.0.0/0 state RELATED,ESTABLISHED
0 0 ACCEPT icmp -- * * 0.0.0.0/0 0.0.0.0/0
0 0 ACCEPT all -- 10 * 0.0.0.0/0 0.0.0.0/0
0 0 ACCEPT tcp -- * * 0.0.0.0/0 0.0.0.0/0 state NEW tcp dpt:22
2 474 REJECT all -- * * 0.0.0.0/0 0.0.0.0/0 reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
0 0 REJECT all -- * * 0.0.0.0/0 0.0.0.0/0 reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT 6 packets, 778 bytes)
pkts bytes target prot opt in out source destination
root@mail:/#
```

Step 3: Add firewall rules

`sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT # Allow HTTP`

`sudo iptables -A INPUT -p tcp --dport 443 -j ACCEPT # Allow HTTPS`

`sudo iptables -A INPUT -p tcp --dport 21 -j DROP # Deny FTP`

`sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT # Allow SSH`

```
DNS on PRIYANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:46 PM
root@mail:/ ~ sudo su

root@mail:/# iptables -A INPUT -p tcp --dport 80 -j ACCEPT
root@mail:/# iptables -A INPUT -p tcp --dport 443 -j ACCEPT
root@mail:/# iptables -A INPUT -p tcp --dport 21 -j ACCEPT
root@mail:/# iptables -A INPUT -p tcp --dport 22 -j ACCEPT
root@mail:/#
```

Step 4: Save iptables rules

`sudo service iptables save`

You should see:

Saving firewall rules to /etc/sysconfig/iptables: [OK]

```
DNS on PRIYANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:47 PM
root@mail:/ ~ sudo su

root@mail:/# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables: [ OK ]
root@mail:/#
```

Step 5: Restart iptables service

`sudo systemctl restart iptables`

```
DNS on PRIYANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:48 PM
root@mail:/ ~ sudo su

root@mail:/# systemctl restart iptables.service
root@mail:/#
```

Step 6: Verify rules

`sudo iptables -L -n -v`

```
DNS on PRIYANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:48 PM
root@mail:/ ~ sudo su

root@mail:/# iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source                 destination
  4 304 ACCEPT     all  --  *      *       0.0.0.0/0             0.0.0.0/0
  0 0 ACCEPT     icmp --  *      *       0.0.0.0/0             0.0.0.0/0
  0 0 ACCEPT     all  --  *      *       0.0.0.0/0             0.0.0.0/0
  0 0 ACCEPT     tcp  --  *      *       0.0.0.0/0             0.0.0.0/0
  0 0 REJECT     all  --  *      *       0.0.0.0/0             0.0.0.0/0
  0 0 ACCEPT     tcp  --  *      *       0.0.0.0/0             0.0.0.0/0
  0 0 ACCEPT     tcp  --  *      *       0.0.0.0/0             0.0.0.0/0
  0 0 ACCEPT     tcp  --  *      *       0.0.0.0/0             0.0.0.0/0
  0 0 ACCEPT     tcp  --  *      *       0.0.0.0/0             0.0.0.0/0

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source                 destination
  0 0 REJECT     all  --  *      *       0.0.0.0/0             0.0.0.0/0

Chain OUTPUT (policy ACCEPT 4 packets, 304 bytes)
  pkts bytes target     prot opt in     out     source                 destination
root@mail:/#
```

Step 7: Delete a rule (optional)

If you want to remove a specific rule:

sudo iptables -D INPUT 8

sudo iptables -D INPUT 6

```
DNS on PRIVANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:53 PM
root@mail/ - sudo su

root@mail:/# iptables -L INPUT -n --line-numbers
Chain INPUT (policy ACCEPT)
num target prot opt source destination
1 ACCEPT all -- 0.0.0.0/0 0.0.0.0/0 state RELATED,ESTABLISHED
2 ACCEPT icmp -- 0.0.0.0/0 0.0.0.0/0
3 ACCEPT all -- 0.0.0.0/0 0.0.0.0/0
4 ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 state NEW tcp dpt:22
5 REJECT all -- 0.0.0.0/0 0.0.0.0/0 reject-with icmp-host-prohibited
6 ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:80
7 ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:443
8 ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:21
9 ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:22

root@mail:/# iptables -D INPUT 8
root@mail:/# iptables -D INPUT 6
root@mail:/# iptables -L -n -v

Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
20 1520 ACCEPT all -- * * 0.0.0.0/0 0.0.0.0/0 state RELATED,ESTABLISHED
0 0 ACCEPT icmp -- * * 0.0.0.0/0 0.0.0.0/0
0 0 ACCEPT all -- lo * 0.0.0.0/0 0.0.0.0/0
0 0 ACCEPT tcp -- * * 0.0.0.0/0 0.0.0.0/0 state NEW tcp dpt:22
4 948 REJECT all -- * * 0.0.0.0/0 0.0.0.0/0 reject-with icmp-host-prohibited
0 0 ACCEPT tcp -- * * 0.0.0.0/0 0.0.0.0/0 tcp dpt:443
0 0 ACCEPT tcp -- * * 0.0.0.0/0 0.0.0.0/0 tcp dpt:22

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
pkts bytes target prot opt in out source destination
0 0 REJECT all -- * * 0.0.0.0/0 0.0.0.0/0 reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT 24 packets, 2468 bytes)
pkts bytes target prot opt in out source destination
root@mail:/#
```

Step 8: Disable iptables (optional)

sudo systemctl stop iptables

sudo systemctl disable iptables

```
DNS on PRIVANSHI - Virtual Machine Connection
File Action Media Clipboard View Help
Oct 30 6:54 PM
root@mail/ - sudo su

root@mail:/# systemctl stop iptables.service
root@mail:/# systemctl disable iptables.service
Removed '/etc/systemd/system/multi-user.target.wants/iptables.service'.
root@mail:/#
```

Task 23: Block Ports & Verify

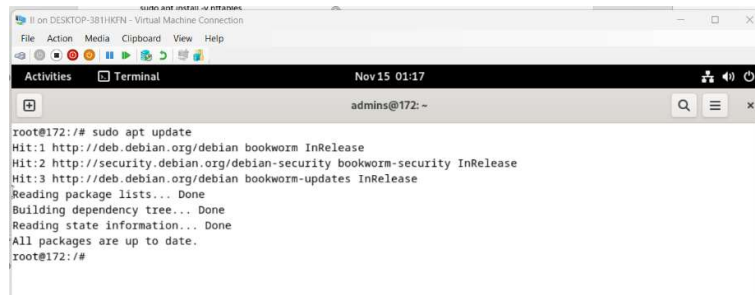
Block ports and verify using Nmap that they are closed.

STEP 1 — Install nftables

Command:

```
sudo apt update
```

```
sudo apt install -y nftables
```



```
root@172: /# sudo apt update
Hit:1 http://deb.debian.org/debian bookworm InRelease
Hit:2 http://security.debian.org/debian-security bookworm-security InRelease
Hit:3 http://deb.debian.org/debian bookworm-updates InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
root@172: /#
```



```
root@172: /# sudo apt install -y nftables
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nftables is already the newest version (1.0.6-2+deb12u2).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@172: /#
```

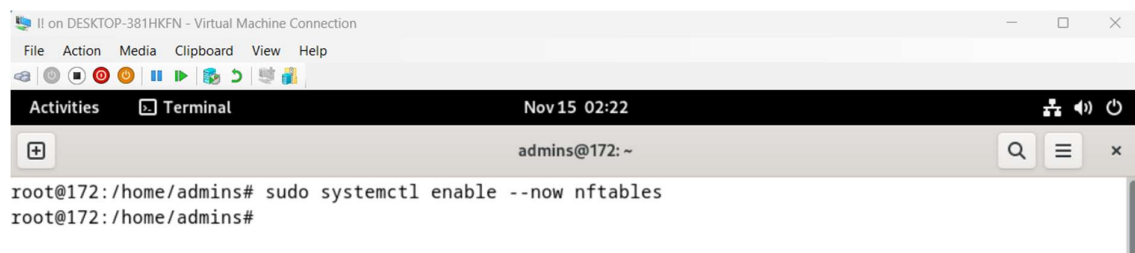
Explanation:

- apt update refreshes your Debian package list so it knows the latest available versions
- nftables is the modern Linux firewall system (it replaces older iptables).
- Installing it ensures the firewall commands (nft) work properly.

STEP 2 — Enable and start the nftables service

Command:

```
sudo systemctl enable --now nftables
```



```
root@172: /home/admins# sudo systemctl enable --now nftables
root@172: /home/admins#
```

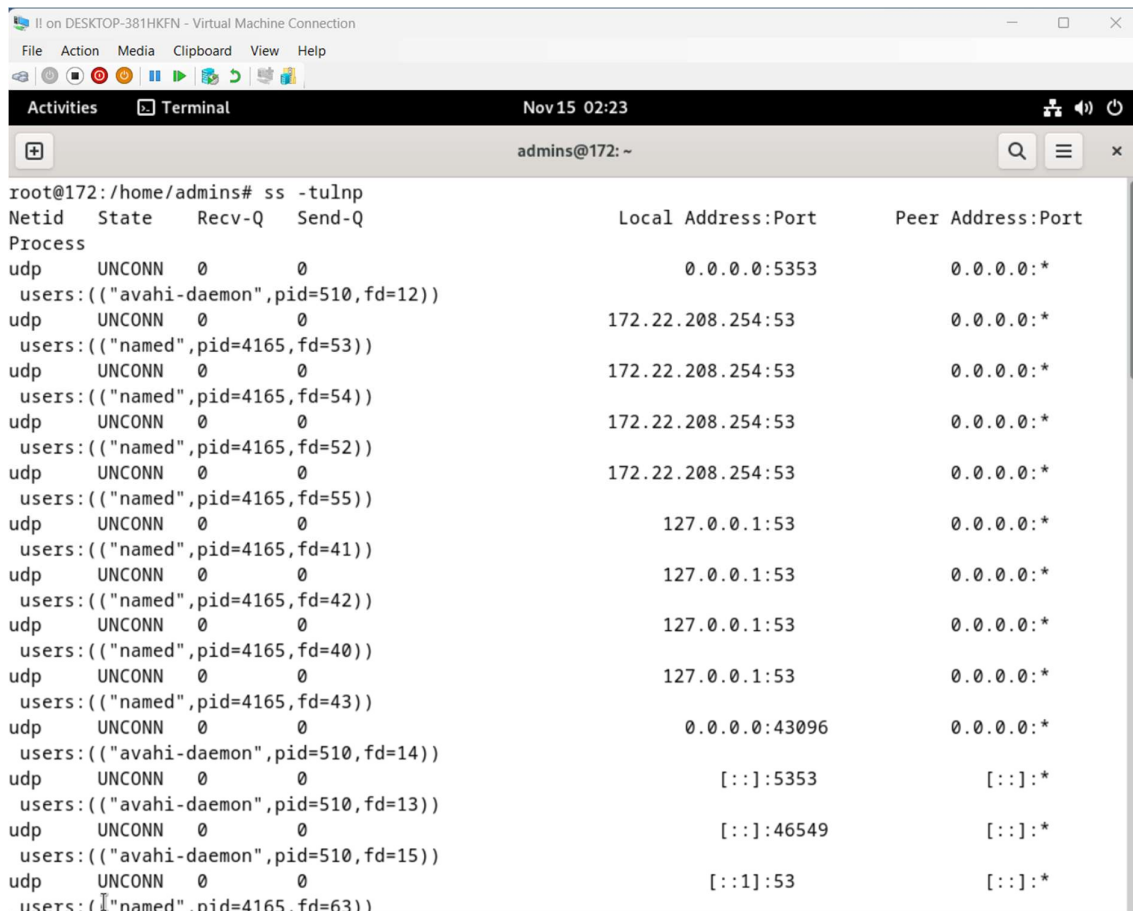

Explanation:

- systemctl enable → makes nftables start every time the system boots.
- --now → starts the nftables service immediately.
- Without enabling this, your firewall rules may not work after a reboot.
- This is required so that /etc/nftables.conf gets loaded automatically when the system starts.

STEP 3 — Check current open ports (optional but important)

Command:

ss -tulnp



```
root@172: /home/admins# ss -tulnp
Netid  State  Recv-Q  Send-Q           Local Address:Port           Peer Address:Port
Process
udp    UNCONN  0        0           0.0.0.0:5353                0.0.0.0:*
users: ( ("avahi-daemon", pid=510, fd=12) )
udp    UNCONN  0        0        172.22.208.254:53          0.0.0.0:*
users: ( ("named", pid=4165, fd=53) )
udp    UNCONN  0        0        172.22.208.254:53          0.0.0.0:*
users: ( ("named", pid=4165, fd=54) )
udp    UNCONN  0        0        172.22.208.254:53          0.0.0.0:*
users: ( ("named", pid=4165, fd=52) )
udp    UNCONN  0        0        172.22.208.254:53          0.0.0.0:*
users: ( ("named", pid=4165, fd=55) )
udp    UNCONN  0        0          127.0.0.1:53                0.0.0.0:*
users: ( ("named", pid=4165, fd=41) )
udp    UNCONN  0        0          127.0.0.1:53                0.0.0.0:*
users: ( ("named", pid=4165, fd=42) )
udp    UNCONN  0        0          127.0.0.1:53                0.0.0.0:*
users: ( ("named", pid=4165, fd=40) )
udp    UNCONN  0        0          127.0.0.1:53                0.0.0.0:*
users: ( ("named", pid=4165, fd=43) )
udp    UNCONN  0        0           0.0.0.0:43096              0.0.0.0:*
users: ( ("avahi-daemon", pid=510, fd=14) )
udp    UNCONN  0        0           [::]:5353                  [::]:*
users: ( ("avahi-daemon", pid=510, fd=13) )
udp    UNCONN  0        0           [::]:46549                 [::]:*
users: ( ("avahi-daemon", pid=510, fd=15) )
udp    UNCONN  0        0           [::1]:53                   [::]:*
users: ( ("named", pid=4165, fd=63) )
```

Explanation:

- ss (socket statistics) shows what services are currently listening.
- -t → TCP
- -u → UDP
- -l → listening ports
- -n → show port numbers instead of service names
- -p → show the process name and PID

Example output:

```
LISTEN 0 128 0.0.0.0:22 *:* users:(("sshd",pid=700,fd=3))
```

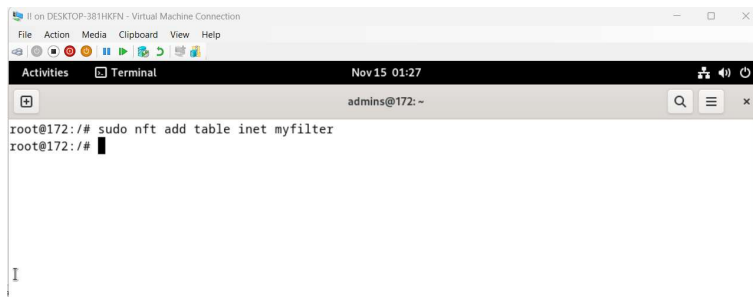
This shows SSH is listening on port 22.

Even if a service is listening, firewall can block it — firewall rules override listening services.

STEP 4 — Create a new nftables table

Command:

```
sudo nft add table inet myfilter
```

A screenshot of a terminal window titled "it on DESKTOP-381HKFN - Virtual Machine Connection". The terminal shows the command `root@172:/# sudo nft add table inet myfilter` being entered and executed. The prompt changes to `root@172:/#` after the command is run. The terminal window has a menu bar with "File", "Action", "Media", "Clipboard", "View", and "Help". The top status bar shows "Nov 15 01:27" and "admins@172: ~".

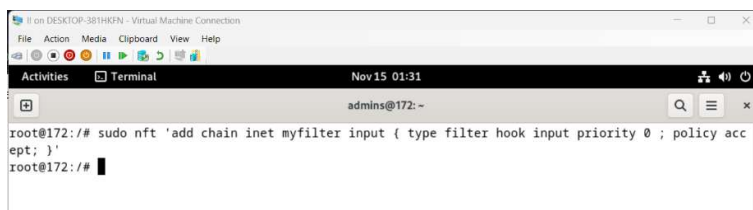
Explanation:

- You are creating a new firewall table named myfilter.
- inet means the table works for both IPv4 and IPv6.
- A table is like a folder where firewall chains and rules live.
- This keeps your firewall rules organized and clean.

STEP 5 — Create INPUT chain with policy

Command:

```
sudo nft 'add chain inet myfilter input { type filter hook input priority 0 ; policy accept; }'
```

A screenshot of a terminal window titled "it on DESKTOP-381HKFN - Virtual Machine Connection". The terminal shows the command `root@172:/# sudo nft 'add chain inet myfilter input { type filter hook input priority 0 ; policy accept; }'` being entered and executed. The prompt changes to `root@172:/#` after the command is run. The terminal window has a menu bar with "File", "Action", "Media", "Clipboard", "View", and "Help". The top status bar shows "Nov 15 01:31" and "admins@172: ~".

Explanation:

- A chain is a list of rules that traffic passes through.
- input chain handles incoming traffic to your system (important).
- type filter → this chain is used for filtering packets.
- hook input → attaches this chain to the Linux kernel's input handling.
- priority 0 → normal priority for filtering.
- policy accept → default action is allow (we will explicitly block only required ports).

This structure ensures:

- System remains safe.
- You do not accidentally block everything.
- Rules work in the correct order.

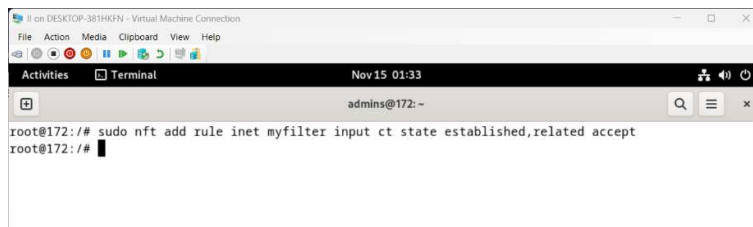
STEP 6 — Add essential allow rules (SAFETY RULES)

These rules prevent system break or accidental lockout.

6.1 Allow ESTABLISHED and RELATED connections

Command:

```
sudo nft add rule inet myfilter input ct state established,related accept
```

A screenshot of a terminal window titled "Terminal" with a timestamp of "Nov 15 01:33". The prompt is "admins@172: ~". The command entered is "root@172:/# sudo nft add rule inet myfilter input ct state established,related accept". The output shows the command being executed successfully, with the prompt returning to "root@172:/#".

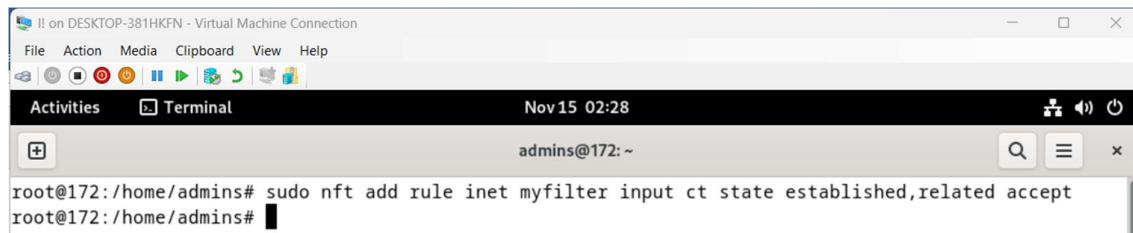
Explanation:

- This rule allows ongoing connections (SSH session, updates, downloads).
- ct state established → packets that belong to an existing connection.
- related → packets related to existing connections (like FTP data).
- This prevents your internet or SSH session from suddenly disconnecting.

6.2 Allow loopback interface

Command:

```
sudo nft add rule inet myfilter input iif lo accept
```

A screenshot of a terminal window titled "Terminal" with a timestamp of "Nov 15 02:28". The prompt is "admins@172: ~". The command entered is "root@172:/home/admins# sudo nft add rule inet myfilter input ct state established,related accept". The output shows the command being executed successfully, with the prompt returning to "root@172:/home/admins#".

Explanation:

- The loopback interface (lo) is your system's internal network (127.0.0.1).
- Many applications (e.g., MySQL, Apache, systemd) talk to themselves using this interface.
- Blocking this would break internal system communication.

So we explicitly allow it to keep system stable.

6.3 Allow SSH (VERY IMPORTANT)

Command:

`sudo nft add rule inet myfilter input tcp dport 22 accept`

A screenshot of a terminal window titled "on DESKTOP-381HGFN - Virtual Machine Connection". The terminal shows a user prompt "admins@172: ~" and three lines of commands being executed: "root@172:/# sudo nft add rule inet myfilter input iif lo accept", "root@172:/# sudo nft add rule inet myfilter input tcp dport 22 accept", and "root@172:/#". The terminal output is empty, indicating successful execution.

```
admins@172: ~
root@172:/# sudo nft add rule inet myfilter input iif lo accept
root@172:/# sudo nft add rule inet myfilter input tcp dport 22 accept
root@172:/#
```

Explanation:

- If you are connected via SSH (22), blocking port 22 would lock you out permanently.
- This rule ensures SSH remains open even after you start blocking other ports.
- Always allow SSH first in any firewall configuration.

STEP 7 — BLOCK the ports you want

7.1 Block port 80 (HTTP)

Command:

`sudo nft add rule inet myfilter input tcp dport 80 drop`

A screenshot of a terminal window titled "on DESKTOP-381HGFN - Virtual Machine Connection". The terminal shows a user prompt "admins@172: ~" and two lines of commands being executed: "root@172:/# sudo nft add rule inet myfilter input tcp dport 80 drop" and "root@172:/#". The terminal output is empty, indicating successful execution.

```
admins@172: ~
root@172:/# sudo nft add rule inet myfilter input tcp dport 80 drop
root@172:/#
```

Explanation:

- `tcp dport 80` → matches traffic coming to port 80.
- `drop` → silently discard packets (no reply to sender).
- When dropped, Nmap will show these ports as filtered.
- This is useful in secure environments—attackers don't know what's behind the firewall.

7.2 Block port 3306 (MySQL)

Command:

`sudo nft add rule inet myfilter input tcp dport 3306 drop`



```
root@172:/# sudo nft add rule inet myfilter input tcp dport 3306 drop
root@172:/#
```

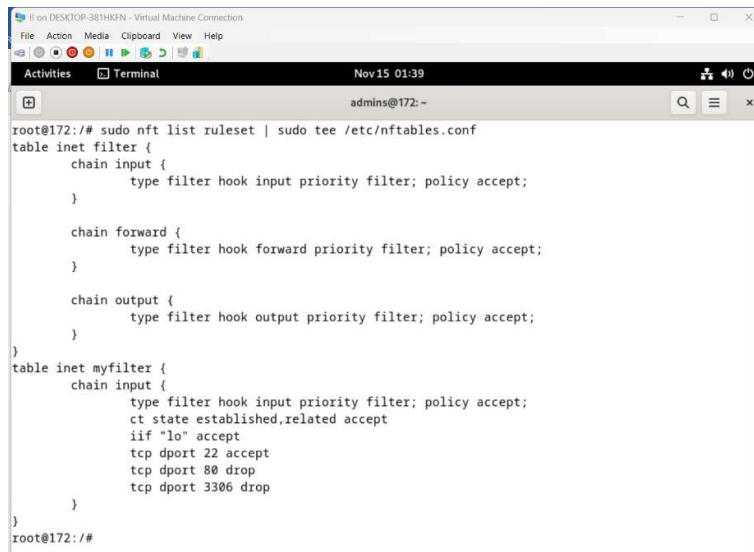
Explanation:

- Same as above, but targeting MySQL port.
- Recommended because exposing DB ports publicly is a major security risk.
- Dropping the port ensures remote systems cannot scan or access it.

STEP 8 — Save your rules permanently

Command:

`sudo nft list ruleset | sudo tee /etc/nftables.conf`



```
root@172:/# sudo nft list ruleset | sudo tee /etc/nftables.conf
table inet filter {
    chain input {
        type filter hook input priority filter; policy accept;
    }

    chain forward {
        type filter hook forward priority filter; policy accept;
    }

    chain output {
        type filter hook output priority filter; policy accept;
    }
}
table inet myfilter {
    chain input {
        type filter hook input priority filter; policy accept;
        ct state established,related accept
        iif "lo" accept
        tcp dport 22 accept
        tcp dport 80 drop
        tcp dport 3306 drop
    }
}
root@172:/#
```

Explanation:

- `nft list ruleset` prints the entire active firewall configuration.
- `tee /etc/nftables.conf` writes it to the main config file.
- `/etc/nftables.conf` is loaded on every boot automatically.

STEP 9 — Verify your rules

Command:

This lets you confirm that:

- Table created
- Input chain exists
- SSH allowed

- Loopback allowed
- Established connections allowed
- Port 80 & 3306 blocked

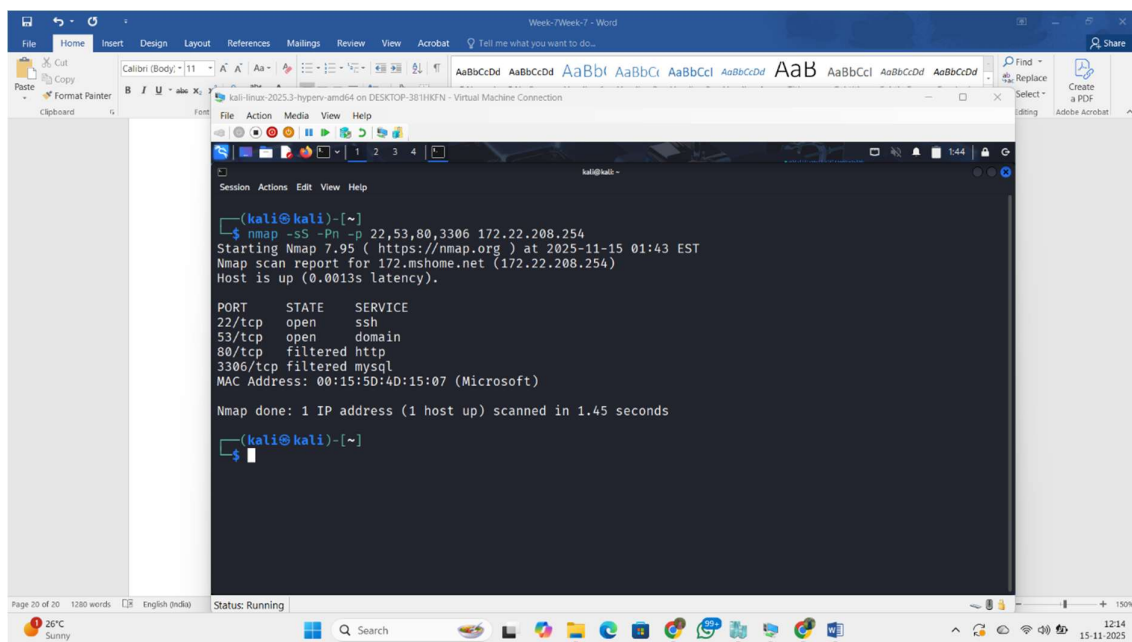
If everything looks correct → firewall is configured properly.

STEP 10 — Verify using Nmap

Run from another machine (NOT from same Debian system):

Command:

`nmap -sS -Pn -p 22,80,3306 <your_server_ip>`



```

(kali@kali)-[~]
└─$ nmap -sS -Pn -p 22,53,80,3306 172.22.208.254
Starting Nmap 7.95 ( https://nmap.org ) at 2025-11-15 01:43 EST
Nmap scan report for 172.mshome.net (172.22.208.254)
Host is up (0.0013s latency).

PORT      STATE      SERVICE
22/tcp    open      ssh
53/tcp    open      domain
80/tcp    filtered   http
3306/tcp   filtered   mysql
MAC Address: 00:15:5D:4D:15:07 (Microsoft)

Nmap done: 1 IP address (1 host up) scanned in 1.45 seconds

(kali@kali)-[~]
└─$
  
```

Explanation:

- -sS → SYN scan (stealth scan — fastest & standard)
- -Pn → skip ping (firewall may block ping)
- -p → specify ports to check

Expected Output:

```

22/tcp    open
80/tcp    filtered
3306/tcp   filtered
  
```

Meaning:

- open → allowed
- filtered → blocked by firewall

Task 24: Enable SSL/TLS

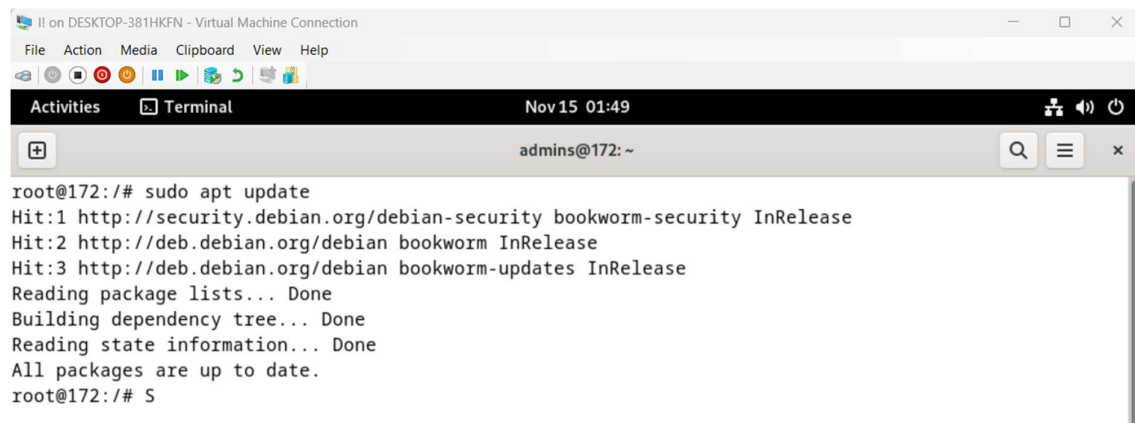
→ Generate a self-signed certificate and enable HTTPS for Apache or Nginx.

You will:

- Install SSL module
- Create a self-signed certificate
- Create secure virtual host
- Enable HTTPS
- Test HTTPS

Step 1: Install SSL module in Apache

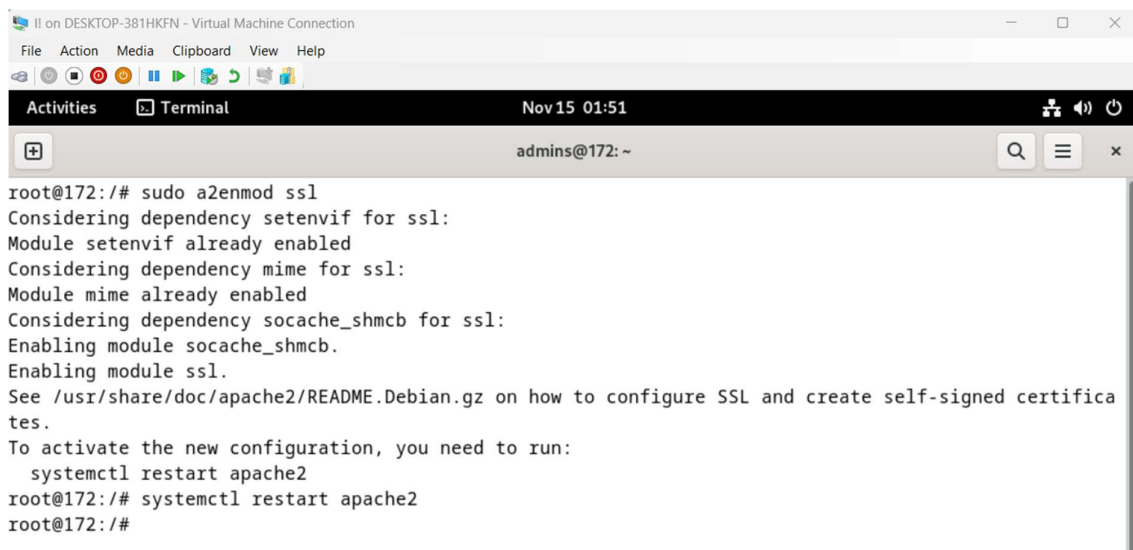
```
sudo apt update
sudo apt install openssl
sudo a2enmod ssl
```

A terminal window titled "II on DESKTOP-381HKFN - Virtual Machine Connection" with a menu bar (File, Action, Media, Clipboard, View, Help) and a toolbar. The terminal shows the command "sudo apt update" being executed. The output lists several security updates from Debian and bookworm, including "bookworm-security InRelease", "bookworm InRelease", and "bookworm-updates InRelease". It also shows "Reading package lists... Done", "Building dependency tree... Done", and "Reading state information... Done". The final message is "All packages are up to date." followed by the prompt "root@172:/# S".

```
root@172:/# sudo apt update
Hit:1 http://security.debian.org/debian-security bookworm-security InRelease
Hit:2 http://deb.debian.org/debian bookworm InRelease
Hit:3 http://deb.debian.org/debian bookworm-updates InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
root@172:/# S
```

A terminal window titled "II on DESKTOP-381HKFN - Virtual Machine Connection" with a menu bar (File, Action, Media, Clipboard, View, Help) and a toolbar. The terminal shows the command "sudo apt install openssl" being executed. The output shows "Reading package lists... Done", "Building dependency tree... Done", and "Reading state information... Done". It then states "openssl is already the newest version (3.0.17-1~deb12u3)." and "openssl set to manually installed." followed by "0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded." and the prompt "root@172:/#".

```
root@172:/# sudo apt install openssl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
openssl is already the newest version (3.0.17-1~deb12u3).
openssl set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
root@172:/#
```

A terminal window titled "II on DESKTOP-381HKFN - Virtual Machine Connection" with a menu bar (File, Action, Media, Clipboard, View, Help) and a toolbar. The terminal shows the command 'sudo a2enmod ssl' being executed, which outputs information about enabling the SSL module and its dependencies (setenvif, mime, socache_shmcb). It then prompts to run 'systemctl restart apache2', which is also executed. The prompt returns to 'root@172: /#'.

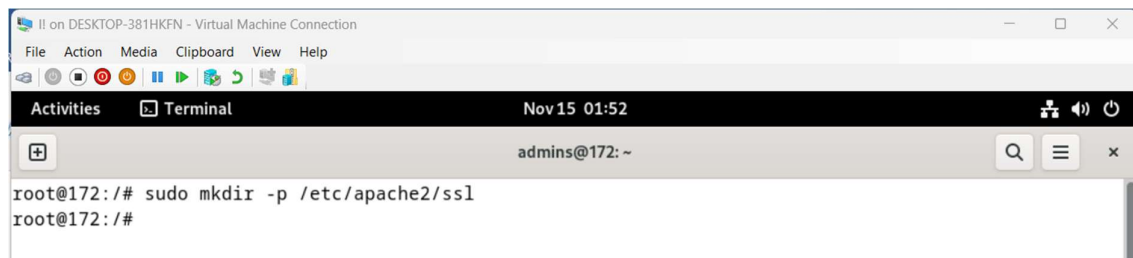
```
root@172: /# sudo a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Enabling module socache_shmcb.
Enabling module ssl.
See /usr/share/doc/apache2/README.Debian.gz on how to configure SSL and create self-signed certificates.
To activate the new configuration, you need to run:
    systemctl restart apache2
root@172: /# systemctl restart apache2
root@172: /#
```

Explanation:

- openssl → Tool used to generate certificates
- a2enmod ssl → Enables SSL module in Apache so Apache can serve HTTPS traffic.

Step 2: Create a Directory for SSL Certificates

`sudo mkdir -p /etc/apache2/ssl`

A terminal window titled "II on DESKTOP-381HKFN - Virtual Machine Connection" with a menu bar (File, Action, Media, Clipboard, View, Help) and a toolbar. The terminal shows the command 'sudo mkdir -p /etc/apache2/ssl' being executed, and the prompt returns to 'root@172: /#'.

```
root@172: /# sudo mkdir -p /etc/apache2/ssl
root@172: /#
```

Explanation:

This is the folder where your key and certificate will be stored.

Step 3: Generate Self-Signed SSL Certificate

```
sudo openssl req -x509 -nodes -days 365 \
-newkey rsa:2048 \
-keyout /etc/apache2/ssl/selfsigned.key \
-out /etc/apache2/ssl/selfsigned.crt
```


- -x509 → Create certificate
- -nodes → No password protection
- -days 365 → Valid for 1 year
- -newkey rsa:2048 → Create private key
- selfsigned.key → Private key
- selfsigned.crt → Public certificate

You will be asked some details:

Just press ENTER for all except Common Name:

Common Name:

Enter your server IP, for example:

172.22.208.254

Step 4: Create HTTPS Virtual Host

Open SSL configuration file:

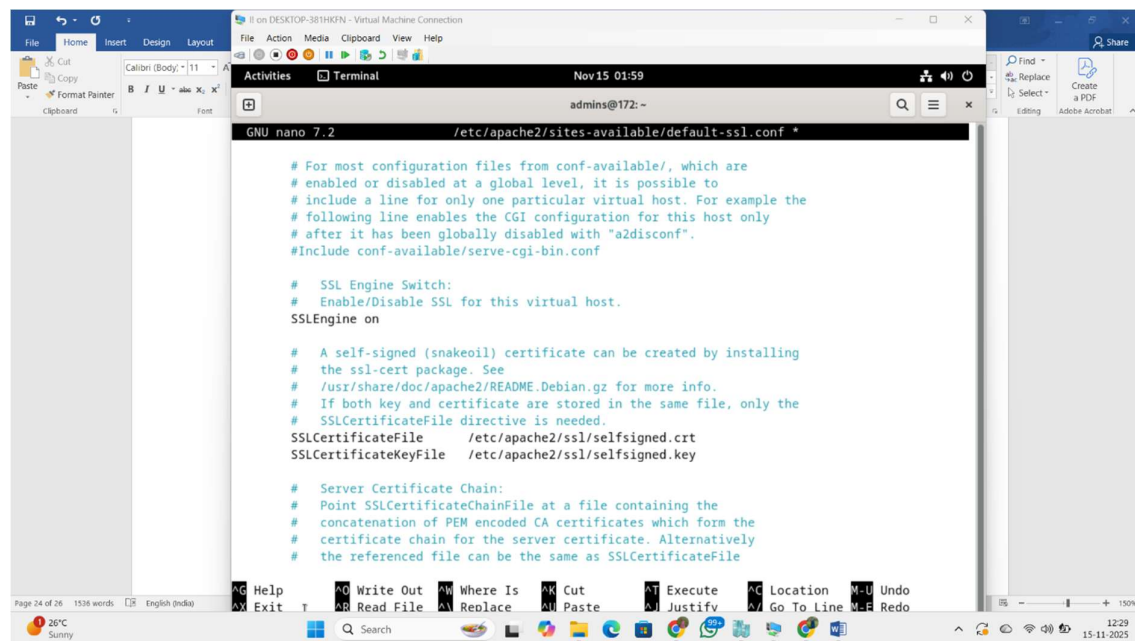
`sudo nano /etc/apache2/sites-available/default-ssl.conf`

Modify these lines:

SSL Engine on

SSLCertificateFile /etc/apache2/ssl/selfsigned.crt

SSLCertificateKeyFile /etc/apache2/ssl/selfsigned.key



The screenshot shows a terminal window titled "on DESKTOP-381H0FN - Virtual Machine Connection" with a date and time of "Nov 15 01:59". The user is logged in as "admins@172: ~". The terminal shows the command `GNU nano 7.2 /etc/apache2/sites-available/default-ssl.conf *` and the following configuration file content:

```
# For most configuration files from conf-available/, which are
# enabled or disabled at a global level, it is possible to
# include a line for only one particular virtual host. For example the
# following line enables the CGI configuration for this host only
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf

#
# SSL Engine Switch:
# Enable/Disable SSL for this virtual host.
SSL Engine on

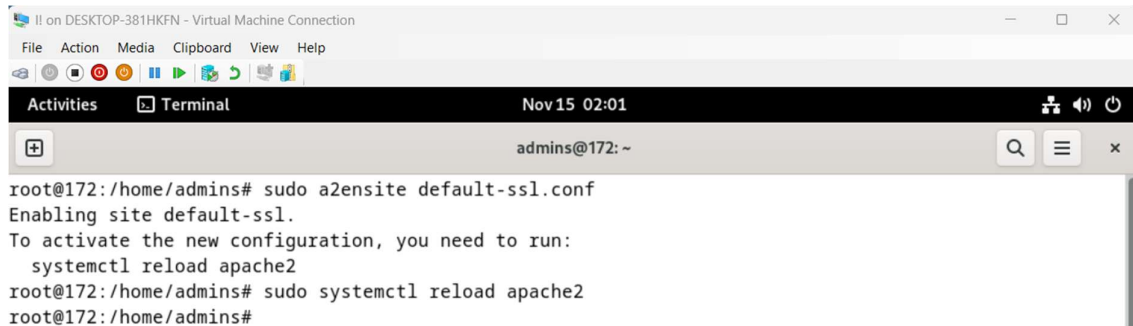
#
# A self-signed (snakeoil) certificate can be created by installing
# the ssl-cert package. See
# /usr/share/doc/apache2/README.Debian.gz for more info.
# If both key and certificate are stored in the same file, only the
# SSLCertificateFile directive is needed.
SSLCertificateFile /etc/apache2/ssl/selfsigned.crt
SSLCertificateKeyFile /etc/apache2/ssl/selfsigned.key

#
# Server Certificate Chain:
# Point SSLCertificateChainFile at a file containing the
# concatenation of PEM encoded CA certificates which form the
# certificate chain for the server certificate. Alternatively
# the referenced file can be the same as SSLCertificateFile
```

Step 5: Enable the SSL Site

`sudo a2ensite default-ssl.conf`

`sudo systemctl reload apache2`

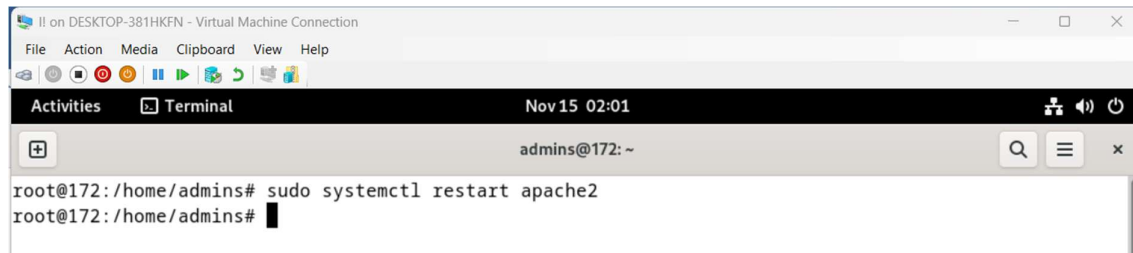


A terminal window titled "II on DESKTOP-381HKFN - Virtual Machine Connection" showing the following commands and output:

```
root@172:/home/admins# sudo a2ensite default-ssl.conf
Enabling site default-ssl.
To activate the new configuration, you need to run:
    systemctl reload apache2
root@172:/home/admins# sudo systemctl reload apache2
root@172:/home/admins#
```

Step 6: Restart Apache

`sudo systemctl restart apache2`



A terminal window titled "II on DESKTOP-381HKFN - Virtual Machine Connection" showing the following command and output:

```
root@172:/home/admins# sudo systemctl restart apache2
root@172:/home/admins#
```

Step 7: Test HTTPS

Open your browser and type:

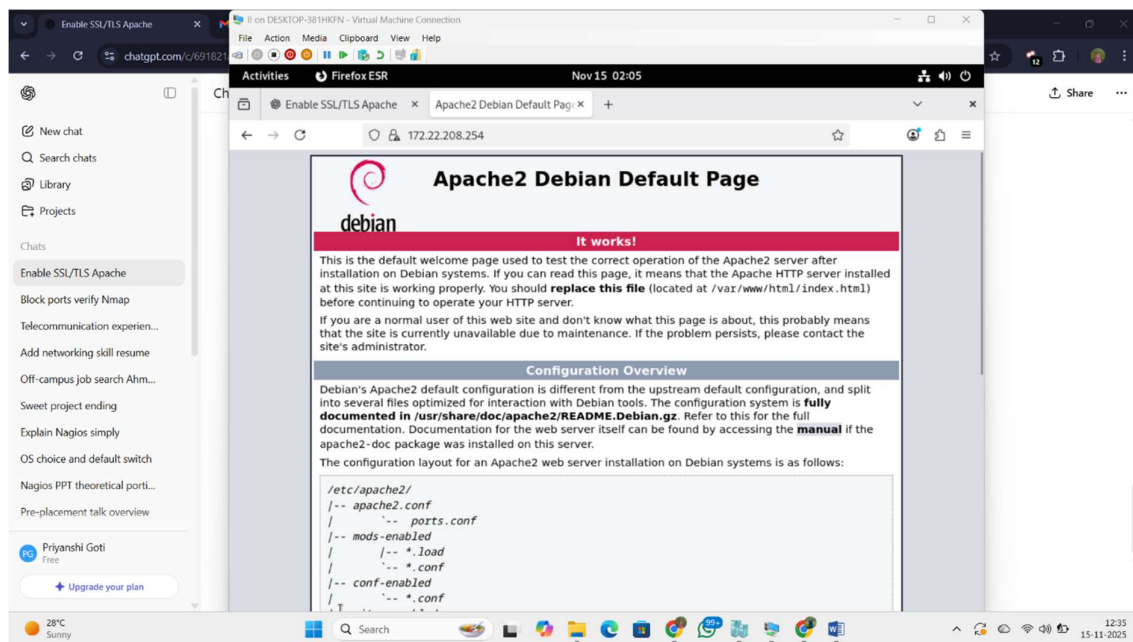
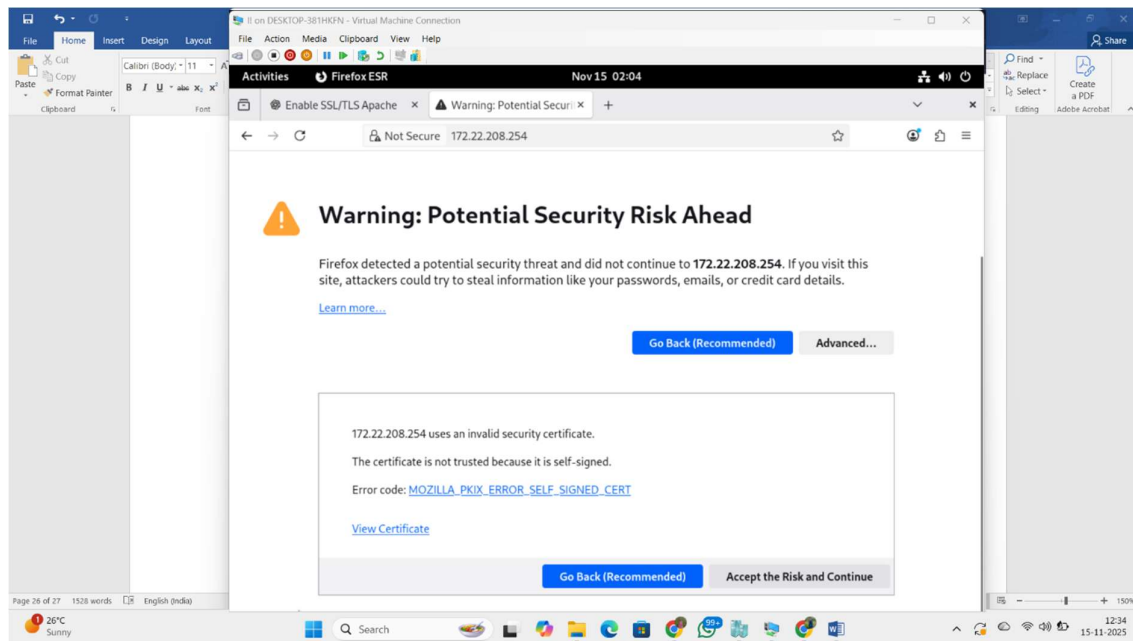
`https://192.168.10.11`

You will get a browser warning:

⚠ "Your connection is not private"

This is normal for self-signed certificates.

Click → Advanced → Proceed



RESULT: HTTPS successfully enabled!

Now your Apache site supports:

- Encrypted traffic
- Secure Nagios/Apache monitoring and Modern TLS security layer

