

ELEVATE LABS

TASK 10

Firewall Configuration & Testing

Student Name: Chalumuri Sri Venkata Srinivas

University: Aditya University

Domain: Cybersecurity

Duration: 1st January 2026 to 30th April 2026

1. Learn firewall concepts.

A firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predefined security rules.

Its main purpose is to protect a computer or network from unauthorized access, attacks, and threats.

Why is a Firewall Needed?

Firewalls are used to:

- Prevent **unauthorized access**
- Block **malicious traffic**
- Protect against **hackers, malware, and intrusions**
- Control which services and ports are allowed
- Enforce **security policies**

How Does a Firewall Work?

A firewall:

1. Inspects network traffic (packets)
2. Checks traffic against defined rules
3. **Allows or blocks** traffic based on:
 - IP address
 - Port number
 - Protocol (TCP/UDP/ICMP)
 - Application or service

Types of Firewalls

Packet Filtering Firewall is the most basic type, which examines individual data packets and allows or blocks them based on source IP address, destination IP address, port number, and protocol. It works at the network layer and is fast, but it does not inspect the packet contents, making it less secure against advanced attacks.

Stateful Inspection Firewall improves upon packet filtering by tracking the state of active connections. It monitors whether a packet is part of an existing, established session and allows only legitimate traffic, providing better security and control.

Application Layer Firewall, also known as a proxy firewall, operates at the application layer and inspects the actual content of network traffic. It acts as an intermediary between the user and the server, filtering requests based on application-specific rules such as HTTP or FTP commands. This type offers high security by blocking malicious payloads but may reduce performance due to deep inspection.

Next-Generation Firewall (NGFW) combines traditional firewall functions with advanced security features such as deep packet inspection, intrusion prevention systems (IPS), application awareness, and malware protection. NGFWs are widely used in modern enterprise networks to defend against sophisticated cyber threats.

2. Configure rules

Firewall rules are instructions that tell a firewall which traffic to allow or block.

These rules are based on parameters such as IP address, port number, protocol (TCP/UDP), direction (inbound/outbound), and application. By configuring rules, administrators control network access and reduce the attack surface.

A typical firewall rule answers these questions:

- Who can communicate? (source IP)
- To whom? (destination IP)
- How? (protocol)
- Through which port? (port number)
- Allow or deny?

Rules usually follow a top-down order, meaning the firewall checks rules sequentially and applies the first matching rule. Best practice is to allow only required traffic and block everything else by default (principle of least privilege).

PRACTICAL 1: Configure Rules using UFW

Step 1: Check UFW Status

```
sudo ufw status
```

```
└─(kali㉿kali)-[~]
└$ sudo ufw status
Status: inactive
```

If inactive, enable it:

```
sudo ufw enable
```

```
└─(kali㉿kali)-[~]
└$ sudo ufw enable
Firewall is active and enabled on system startup
```

Step 2: Allow a Specific Port (Example: SSH – Port 22)

```
sudo ufw allow 22
```

```
└─(kali㉿kali)-[~]
└$ sudo ufw allow 22
Rule added
Rule added (v6)
```

Step 3: Allow HTTP and HTTPS

```
sudo ufw allow 80
```

```
sudo ufw allow 443
```

```
└─(kali㉿kali)-[~]
└$ sudo ufw allow 80
sudo ufw allow 443
Rule added
Rule added (v6)
Rule added
Rule added (v6)
```

Step 4: Block a Port (Example: Telnet – Port 23)

```
sudo ufw deny 23
```

```
└─(kali㉿kali)-[~]
└$ sudo ufw deny 23
Rule added
Rule added (v6)
```

Step 5: Allow Traffic from a Specific IP

sudo ufw allow from ip address

```
[~] (kali㉿kali)-[~]
$ sudo ufw allow from 192.168.10.3
Rule added
```

Step 6: Allow a Port from a Specific IP

sudo ufw allow from 192.168.10.3 to any port 22

```
[~] (kali㉿kali)-[~]
$ sudo ufw allow from 192.168.1.10 to any port 22
Rule added
```

Step 7: Delete a Rule

sudo ufw delete allow 23

```
[~] (kali㉿kali)-[~]
$ sudo ufw delete allow 23
Rule deleted
Rule deleted (v6)
```

Step 8: View Rules with Numbers

sudo ufw status numbered

```
[~] (kali㉿kali)-[~]
$ sudo ufw status numbered
Status: active

          To          Action    From
          --          --
[ 1] 22          ALLOW IN  Anywhere
[ 2] 80          ALLOW IN  Anywhere
[ 3] 443         ALLOW IN  Anywhere
[ 4] Anywhere    ALLOW IN  192.168.10.3
[ 5] 22          ALLOW IN  192.168.1.10
[ 6] 22 (v6)    ALLOW IN  Anywhere (v6)
[ 7] 80 (v6)    ALLOW IN  Anywhere (v6)
[ 8] 443 (v6)   ALLOW IN  Anywhere (v6)

[~] (kali㉿kali)-[~]
```

3. Allow/deny ports

A port is a logical communication endpoint used by network services to send and receive data. Common examples include HTTP (port 80), HTTPS (port 443), SSH (port 22), and FTP (port 21). Firewalls control network access by allowing or denying ports based on security requirements. Allowing a port means permitting traffic for a specific service, while denying a port blocks all traffic attempting to use that service. Proper port management reduces exposure

to attacks by ensuring that only necessary ports are open and all unused or risky ports are blocked.

Firewalls apply rules using protocols such as TCP or UDP and directions such as inbound or outbound. Most security policies follow a default deny approach, where all ports are blocked by default and only required ports are explicitly allowed. This practice minimizes vulnerabilities and protects systems from unauthorized access and network-based attacks.

Allow a Port (Example: SSH – Port 22)

sudo ufw allow 22

```
└─(kali㉿kali)-[~]
└$ sudo ufw allow 22
Rule updated
Rule updated (v6)
```

Allow a Port with Protocol (HTTP – TCP 80)

sudo ufw allow 80/tcp

```
└─(kali㉿kali)-[~]
└$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
```

Deny a Port (Telnet – Port 23)

sudo ufw deny 23

```
└─(kali㉿kali)-[~]
└$ sudo ufw deny 23
Rule added
Rule added (v6)

└─(kali㉿kali)-[~]
```

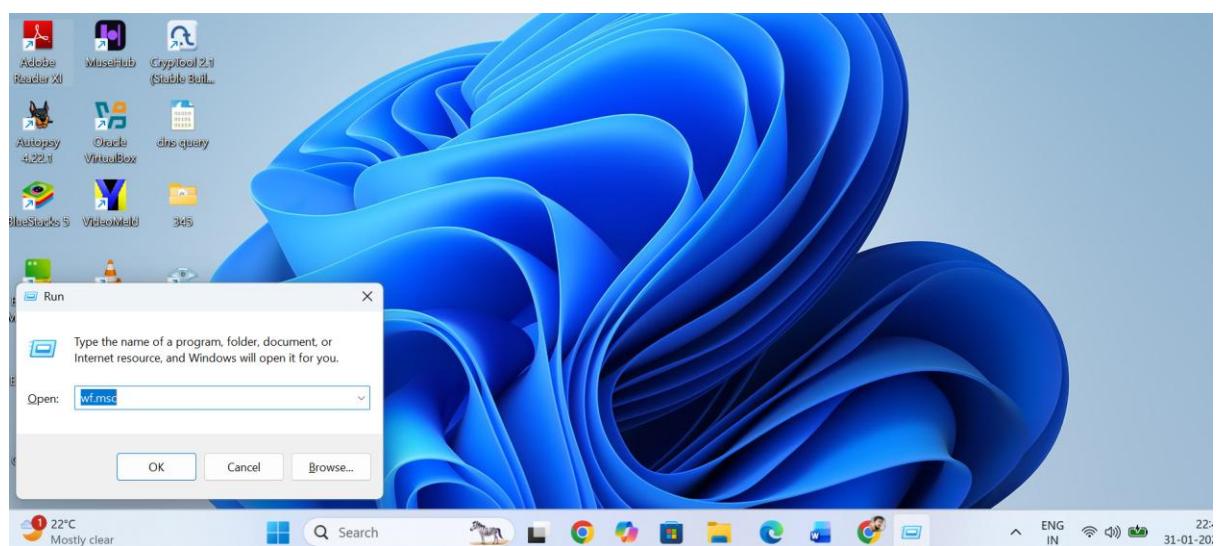
Deny a Port with Protocol

```
└─(kali㉿kali)-[~]
└$ sudo ufw deny 21/tcp
Rule added
Rule added (v6)
```

Allow / Deny Ports using Windows Firewall

Step 1: Open Firewall Console

- Press Win + R
- Type wf.msc
- Press Enter

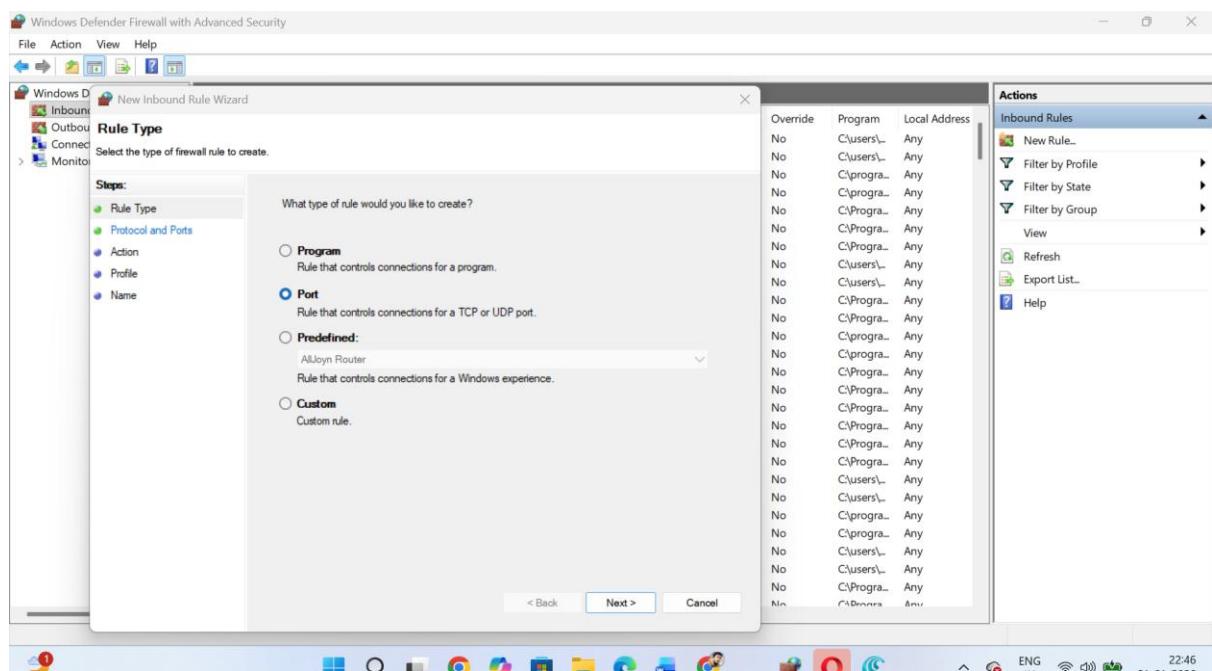


Allow a Port

Click Inbound Rules

Select New Rule

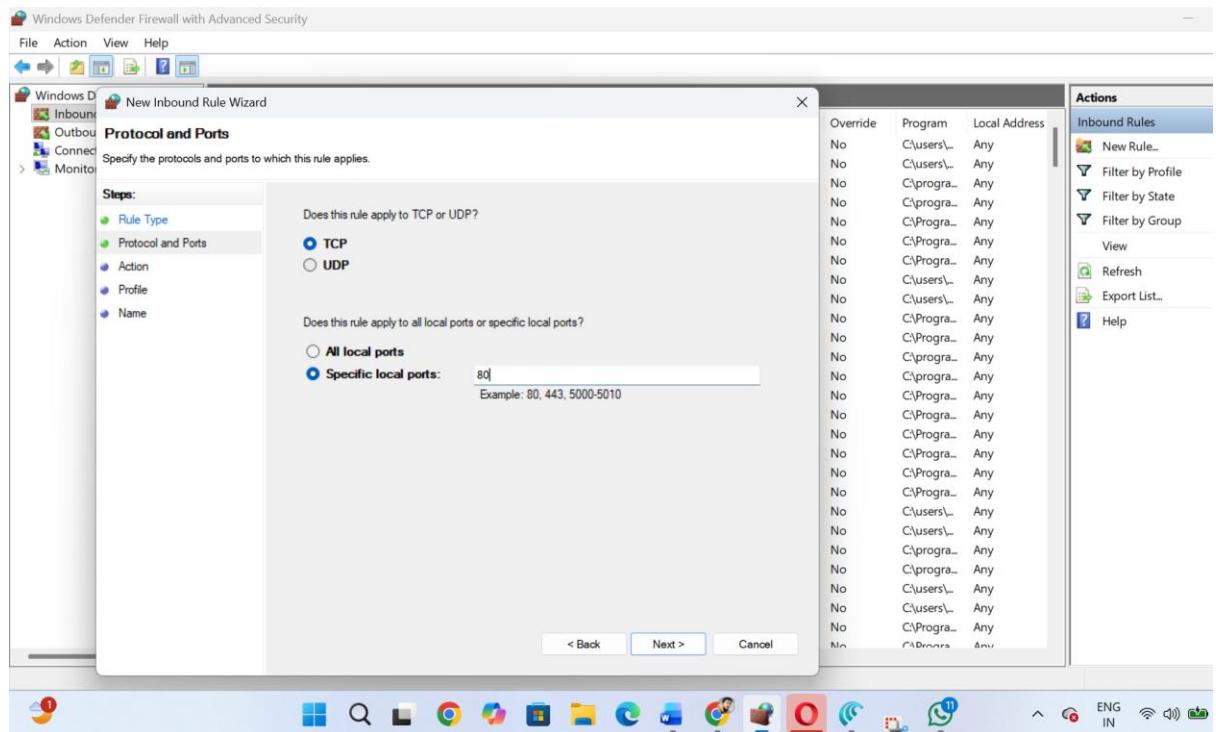
Choose Port



Select **TCP**

Enter port **80**

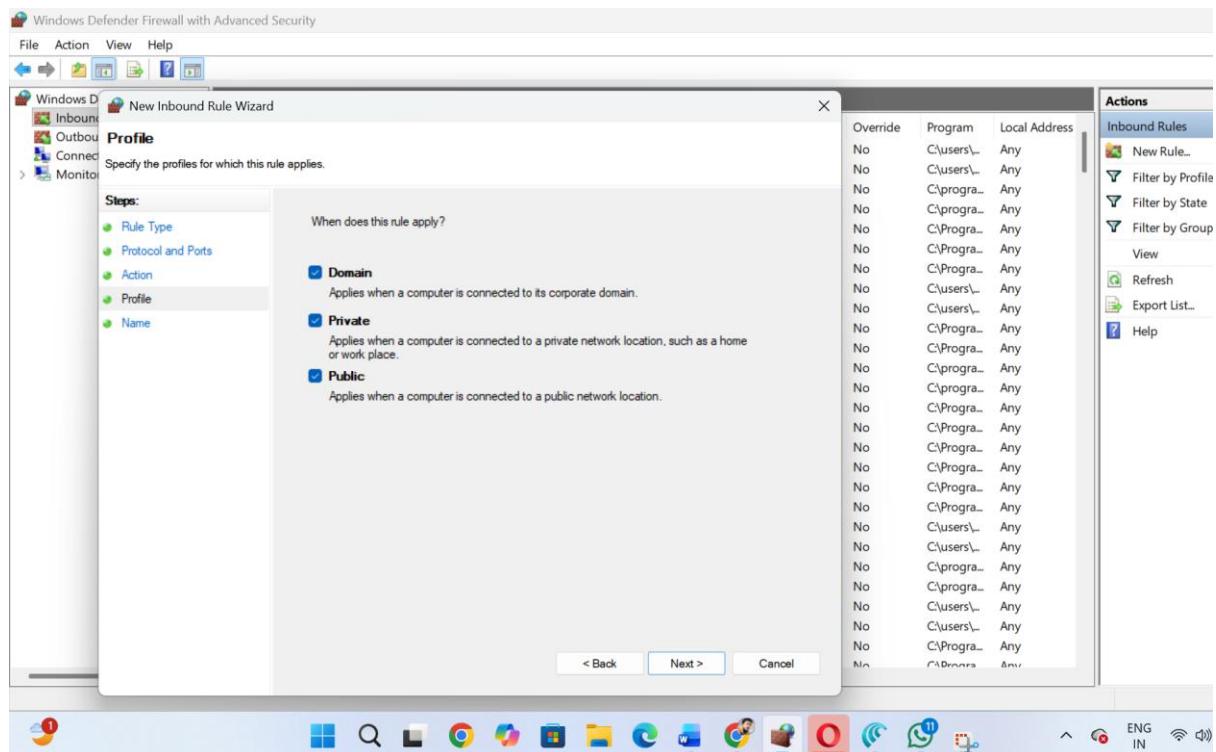
Choose **Allow the connection**

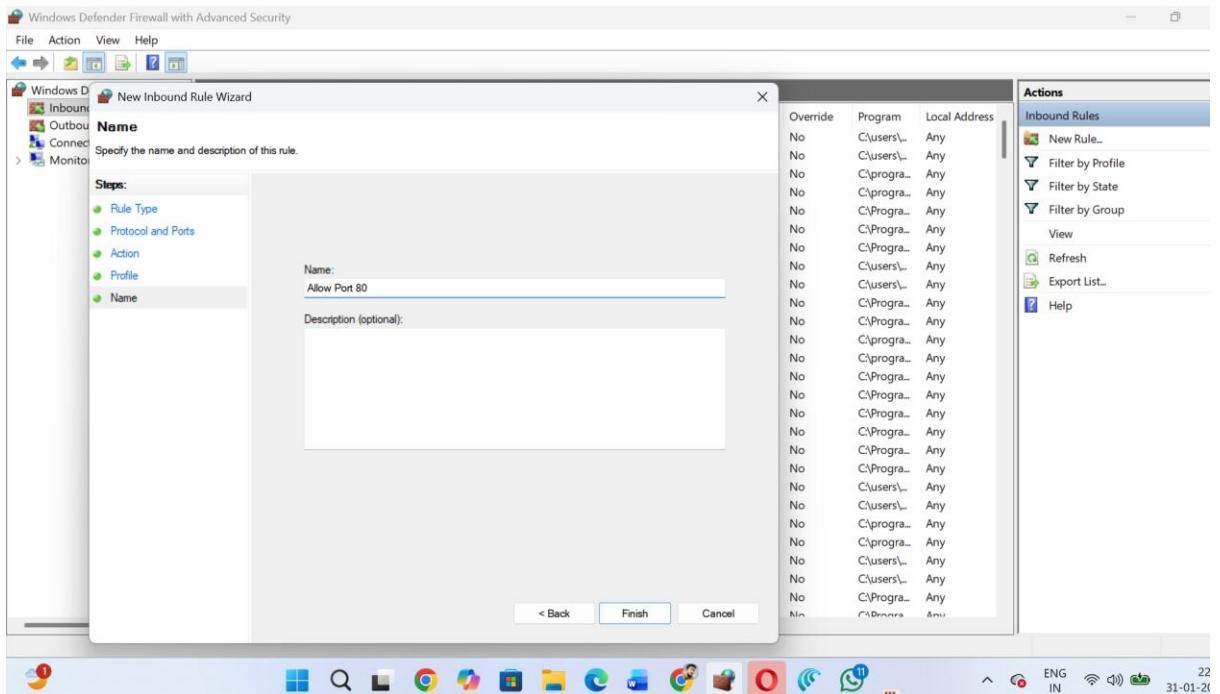


Choose **Allow the connection**

Select profiles (Domain/Private/Public)

Name rule: Allow Port 80





4. Test connectivity.

Testing connectivity is the process of verifying whether network communication between systems is successful after configuring firewall rules. Once ports are allowed or denied, connectivity testing confirms that permitted services are reachable and blocked services are inaccessible. This step ensures that firewall rules are working as intended and have not accidentally blocked legitimate traffic.

Connectivity testing typically uses network diagnostic tools such as ping, telnet, netcat (nc), curl, and Nmap. These tools help identify whether a host is reachable, whether specific ports are open or closed, and how a system responds to connection attempts. Proper testing is essential to avoid misconfigurations that could cause service downtime or security gaps.

Check Host Reachability

```
(kali㉿kali)-[~]
$ ping 192.168.1.1
PING 192.168.1.1 (192.168.1.1) 56(84) bytes of data.

^C
--- 192.168.1.1 ping statistics ---
58 packets transmitted, 0 received, 100% packet loss, time 59795ms
```

Test Allowed Port (Example: HTTP – Port 80)

```
(kali㉿kali)-[~]
$ telnet localhost 80
Trying ::1 ...
Connected to localhost.
Escape character is '^]'.
```

Test Blocked Port (Telnet – Port 23)

```
connection closed by foreign host
(kali㉿kali)-[~]
$ telnet localhost 23
Trying ::1 ...
Connection failed: Connection refused
Trying 127.0.0.1 ...
telnet: Unable to connect to remote host: Connection refused
(kali㉿kali)-[~]
```

5. Observe logs.

Firewalls act as a protective barrier between your system and external networks by controlling incoming and outgoing traffic based on predefined rules.

Observing firewall logs is crucial for system security, as it helps administrators detect suspicious activity, unauthorized access attempts, and misconfigured rules. On Linux, **UFW (Uncomplicated Firewall)** is a front-end for iptables that simplifies firewall management; its logs record allowed or blocked network connections. On Windows, the **Windows Firewall** (or Windows Defender Firewall) logs events such as connection attempts, rule violations, and blocked programs. Regular monitoring of these logs helps in auditing, troubleshooting network issues, and proactively identifying potential attacks, such as port scans, brute-force attempts, or malware communication.

Step 1: Check UFW status

```
sudo ufw status verbose
```

```
Session Actions Edit View Help
zsh: corrupt history file /home/kali/.zsh_history
└─(kali㉿kali)-[~]
└─$ sudo ufw status verbose
[sudo] password for kali:
Sorry, try again.
[sudo] password for kali:
Status: active
Logging: on (medium)
Default: deny (incoming), allow (outgoing), deny (routed)
New profiles: skip

To           Action    From
--           _____
22           ALLOW IN  Anywhere
80           ALLOW IN  Anywhere
443          ALLOW IN  Anywhere
Anywhere     ALLOW IN  192.168.10.3
22           ALLOW IN  192.168.1.10
80/tcp        ALLOW IN  Anywhere
23           DENY IN   Anywhere
21/tcp        DENY IN   Anywhere
Anywhere     DENY IN   192.168.1.50
22 (v6)      ALLOW IN  Anywhere (v6)
80 (v6)      ALLOW IN  Anywhere (v6)
443 (v6)     ALLOW IN  Anywhere (v6)
80/tcp (v6)  ALLOW IN  Anywhere (v6)
23 (v6)      DENY IN   Anywhere (v6)
21/tcp (v6)  DENY IN   Anywhere (v6)
```

Step 2: Enable UFW logging

```
└─(kali㉿kali)-[~]
└─$ sudo ufw logging on
Logging enabled
```

To set log level:

sudo ufw logging low

```
└─(kali㉿kali)-[~]
└─$ sudo ufw logging low
Logging enabled
```

Step 3: Generate some traffic (to create logs)

Option A: Block a port

sudo ufw deny 23

```
└─(kali㉿kali)-[~]
└─$ sudo ufw deny 23
Skipping adding existing rule
Skipping adding existing rule (v6)
```

Option B: Test from same system

```
nc localhost 23
```

```
└─(kali㉿kali)-[~]
└─$ nc localhost 23
localhost [127.0.0.1] 23 (telnet) : Connection refused
```

Step 4: View UFW logs (MAIN STEP)

UFW logs are stored in:

```
/var/log/ufw.log
```

Or use

```
sudo journalctl -f | grep UFW
```

Step 5: Understand log entries (example)

```
└─(kali㉿kali)-[~]
└─$ sudo journalctl -f | grep UFW
Jan 31 19:45:36 kali sudo[5814]:      kali : TTY=pts/0 ; PWD=/home/kali ; USER=root ; COMMAND=/usr/bin/grep UFW /var/log/syslog
```

Step 6: Stop logging

```
sudo ufw logging off
```

```
└─(kali㉿kali)-[~]
└─$ sudo ufw logging off
```

```
Logging disabled
```

6. Block Malicious IP

Blocking a malicious IP address is a defensive firewall technique used to prevent unauthorized access, brute-force attacks, malware communication, and denial-of-service attempts. Firewalls such as UFW (Uncomplicated Firewall) in Linux and Windows Defender Firewall allow administrators to define rules that deny all incoming or outgoing traffic from a specific IP address. Once blocked, any packets originating from the malicious IP are dropped and logged, helping in incident response and forensic analysis. This control reduces attack surface and protects system resources by stopping threats at the network layer before they reach applications.

Step 1: Check UFW status

```
(kali㉿kali)-[~]
$ sudo ufw status verbose

Status: active
Logging: off
Default: deny (incoming), allow (outgoing), deny (routed)
New profiles: skip

To                         Action      From
--                         --          --
22                         ALLOW IN   Anywhere
80                         ALLOW IN   Anywhere
443                        ALLOW IN   Anywhere
Anywhere                    ALLOW IN   192.168.10.3
22                         ALLOW IN   192.168.1.10
80/tcp                     ALLOW IN   Anywhere
23                         DENY IN    Anywhere
21/tcp                      DENY IN    Anywhere
Anywhere                    DENY IN    192.168.1.50
22 (v6)                   ALLOW IN   Anywhere (v6)
80 (v6)                   ALLOW IN   Anywhere (v6)
443 (v6)                  ALLOW IN   Anywhere (v6)
80/tcp (v6)                ALLOW IN   Anywhere (v6)
23 (v6)                   DENY IN    Anywhere (v6)
21/tcp (v6)                DENY IN    Anywhere (v6)
```

If inactive:

sudo ufw enable

Step 2: Enable logging

```
(kali㉿kali)-[~]
$ sudo ufw logging on
Logging enabled
```

Step 3: Block malicious IP (Incoming)

sudo ufw deny from 192.168.1.100

```
(kali㉿kali)-[~]
$ sudo ufw deny from 192.168.1.100
```

Rule added

To block **both incoming & outgoing**:

sudo ufw deny from 192.168.1.100 to any

```
(kali㉿kali)-[~]
$ sudo ufw deny from 192.168.1.100 to any
Skipping adding existing rule
```

Step 4: Verify rule

sudo ufw status numbered

```
(kali㉿kali)-[~]
$ sudo ufw status numbered

Status: active

To                         Action      From
--                         --          --
[ 1] 22                      ALLOW IN   Anywhere
[ 2] 80                      ALLOW IN   Anywhere
[ 3] 443                     ALLOW IN   Anywhere
[ 4] Anywhere                DENY IN    192.168.10.3
[ 5] 22                      ALLOW IN   192.168.1.10
[ 6] 80/tcp                  ALLOW IN   Anywhere
[ 7] 23                      DENY IN    Anywhere
[ 8] 21/tcp                  DENY IN    Anywhere
[ 9] Anywhere                DENY IN    192.168.1.50
[10] 22 (v6)                ALLOW IN   Anywhere (v6)
[11] 80 (v6)                ALLOW IN   Anywhere (v6)
[12] 443 (v6)               ALLOW IN   Anywhere (v6)
[13] 80/tcp (v6)            ALLOW IN   Anywhere (v6)
[14] 23 (v6)                DENY IN    Anywhere (v6)
[15] 21/tcp (v6)             DENY IN    Anywhere (v6)
```

Step 5: Test (from attacker machine)

ping <your-ip>

```
(kali㉿kali)-[~]
$ ping 192.168.1.100
PING 192.168.1.100 (192.168.1.100) 56(84) bytes of data.
^C
--- 192.168.1.100 ping statistics ---
13 packets transmitted, 0 received, 100% packet loss, time 12291ms
```

7. Document rules.

Documenting firewall rules is the process of systematically recording all configured allow and deny policies applied to a firewall. Proper documentation includes details such as rule number, source and destination IP addresses, ports, protocols, direction of traffic, and the purpose of each rule. This practice ensures transparency, simplifies troubleshooting, supports security audits, and helps administrators understand and manage firewall behavior over time. Well-maintained firewall documentation reduces misconfiguration risks and improves incident response efficiency.

Step 1: List all firewall rules

sudo ufw status numbered

```
(kali㉿kali)-[~]
$ sudo ufw status numbered
Status: active

To                         Action      From
--                         --          --
[ 1] 22                      ALLOW IN   Anywhere
[ 2] 80                      ALLOW IN   Anywhere
[ 3] 443                     ALLOW IN   Anywhere
[ 4] Anywhere                DENY IN    192.168.10.3
[ 5] 22                      ALLOW IN   192.168.1.10
[ 6] 80/tcp                  ALLOW IN   Anywhere
[ 7] 23                      DENY IN    Anywhere
[ 8] 21/tcp                  DENY IN    Anywhere
[ 9] Anywhere                DENY IN    192.168.1.50
[10] Anywhere                DENY IN    192.168.1.100
[11] 22 (v6)                 ALLOW IN   Anywhere (v6)
[12] 80 (v6)                 ALLOW IN   Anywhere (v6)
[13] 443 (v6)                ALLOW IN   Anywhere (v6)
[14] 80/tcp (v6)              ALLOW IN   Anywhere (v6)
[15] 23 (v6)                 DENY IN    Anywhere (v6)
[16] 21/tcp (v6)              DENY IN    Anywhere (v6)
```

8. Explain Impact

Firewall configuration and testing have a significant impact on system security and network reliability. Properly configured firewall rules restrict unauthorized access, block malicious traffic, and allow only legitimate services to operate. Testing connectivity ensures that allowed services function correctly while blocked services remain inaccessible. Observing logs helps detect attack attempts and validate rule effectiveness. Overall, firewalls reduce the attack surface, prevent data breaches, and ensure secure and controlled network communication.