

Proof of Concept (PoC) Report

Task 3: Firewall & Network Security

Objective:

Demonstrate how an improperly configured firewall can expose services and how to implement network security measures to protect the system.

1 Setup: Installing & Configuring a Web Server with Weak Security

Step 1: Install Apache Web Server

Command:

- `sudo apt update && sudo apt install apache2 -y`(command for installing)
- Then we begin by starting and enabling apache2 and available to use

```
(kali@kali)-[~]  
$ sudo systemctl start apache2  
  
(kali@kali)-[~]  
$ sudo systemctl enable apache2  
Synchronizing state of apache2.service with SysV service script with /usr/lib  
/systemd/systemd-sysv-install.  
Executing: /usr/lib/systemd/systemd-sysv-install enable apache2  
Created symlink '/etc/systemd/system/multi-user.target.wants/apache2.service'  
→ '/usr/lib/systemd/system/apache2.service'.
```

What Does It Do?

- Installs and enables the Apache web server.
- Starts the Apache service on boot.

Step 2: Disable Firewall (Allow All Traffic)

Command:

- if the ufw tool is not installed use command
(sudo apt update && sudo apt install ufw -y) then enable it
-y:without asking yes or no prompt for installation

```
Firewall is active and enabled on system startup
(kali@kali)-[~]
$ sudo ufw disable
Firewall stopped and disabled on system startup
```

Security Risk:

- Without firewall protection, all ports remain open to attackers.
- Exposed services can be easily discovered and exploited.

2 Exploit: Scanning for Open Ports & Services

Step 3: Perform an Nmap Scan

Command:

```
(kali@kali)-[~]
$ nmap 192.168.64.2
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-03-16 14:39 EDT
Nmap scan report for 192.168.64.2
Host is up (0.0000020s latency).
Not shown: 998 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
80/tcp    open  http
Nmap done: 1 IP address (1 host up) scanned in 0.19 seconds
```

What Does It Do?

- Detects open ports and running services
- Identifies software versions and potential vulnerabilities.

● Critical Risk: Attackers can discover and target these services.

Step 4: Use Netcat to Enumerate Services

Command:

```
(kali@kali)-[~]  
$ nc -nzv 192.168.64.2 80  
(UNKNOWN) [192.168.64.2] 80 (http) open
```

Flag Breakdown

- **-n** → No DNS resolution (avoids delays from hostname lookups).
- **-z** → Zero-I/O mode (only checks if a port is open, without sending data).
- **-v** → Verbose mode (provides detailed output)

What Does It Do?

- Connects to the web server to check for service responses.
- Can be used to manually interact with exposed ports.

● Issue: Attackers can send malicious requests to exploit vulnerabilities.

③ Mitigation: Securing the Network with a Firewall

Step 5: Enable UFW and Restrict Traffic

Command:(setting rules)

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

```
(kali@kali)-[~]
$ sudo ufw default deny incoming
sudo: Default incoming policy changed to 'deny'
(be sure to update your rules accordingly)

(kali@kali)-[~]
$ sudo ufw default allow incoming
Default incoming policy changed to 'allow'
(be sure to update your rules accordingly)

(kali@kali)-[~]
$ sudo ufw allow ssh
Rule added
Rule added (v6)

(kali@kali)-[~]
$ sudo ufw allow http
Rule added
Rule added (v6)
```

What Does It Do?

1. Enables firewall protection.
2. Allows only SSH (22) and HTTP (80) traffic.
3. Blocks all other unauthorized connections.
4. Reload the ufw

```
(kali@kali)-[~]
$ sudo ufw reload
Firewall reloaded
```

Step 6: Implement iptables Rules

Command:

```
(kali㉿kali)-[~]
$ sudo iptables -P INPUT DROP

(kali㉿kali)-[~]
$ sudo iptables -P FORWARD DROP

(kali㉿kali)-[~]
$ sudo iptables -P OUTPUT ACCEPT

(kali㉿kali)-[~]
$ sudo iptables -A INPUT -m --ctstate ESTABLISHED,RELATED -j ACCEPT
iptables v1.8.10 (nf_tables): Couldn't load match '--ctstate':No such file or
directory

Try `iptables -h' or 'iptables --help' for more information.

(kali㉿kali)-[~]
$ sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACE
EPT
iptables v1.8.10 (nf_tables): Chain 'ACEEPT' does not exist
Try `iptables -h' or 'iptables --help' for more information.

(kali㉿kali)-[~]
$ sudo iptables -A INPUT -m conntrack --ctstate ESTABLISHED,RELATED -j ACC
EPT

(kali㉿kali)-[~]
$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT

(kali㉿kali)-[~]
$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT

(kali㉿kali)-[~]
$ sudo iptables-save | sudo tee /etc/iptables/rules.v4
```

```
kali@kali: ~  
File Actions Edit View Help  
$ sudo iptables -A INPUT -p tcp --dport 22 -j ACCEPT  
(kali@kali)-[~]  
$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT  
(kali@kali)-[~]  
$ sudo iptables-save | sudo tee /etc/iptables/rules.v4  
tee: /etc/iptables/rules.v4: No such file or directory  
# Generated by iptables-save v1.8.10 (nf_tables) on Sun Mar 16 15:00:57 2025  
*filter  
:INPUT DROP [0:0]  
:FORWARD DROP [0:0]  
:OUTPUT ACCEPT [0:0]  
:ufw-after-forward - [0:0]  
:ufw-after-input - [0:0]  
:ufw-after-logging-forward - [0:0]  
:ufw-after-logging-input - [0:0]  
:ufw-after-logging-output - [0:0]  
:ufw-after-output - [0:0]  
:ufw-before-forward - [0:0]  
:ufw-before-input - [0:0]  
:ufw-before-logging-forward - [0:0]  
:ufw-before-logging-input - [0:0]  
:ufw-before-logging-output - [0:0]  
:ufw-before-output - [0:0]  
:ufw-logging-allow - [0:0]  
:ufw-logging-deny - [0:0]  
:ufw-not-local - [0:0]  
:ufw-reject-forward - [0:0]  
:ufw-reject-input - [0:0]  
:ufw-reject-output - [0:0]  
:ufw-skip-to-policy-forward - [0:0]  
:ufw-skip-to-policy-input - [0:0]  
:ufw-skip-to-policy-output - [0:0]  
:ufw-track-forward - [0:0]
```

What Does It Do?

- 1.Allows only SSH and HTTP traffic.
- 2.Drops all other incoming connections, preventing unauthorized access.



Conclusion:

Exploitation: Demonstrated how an open firewall exposes services.

Mitigation: Applied firewall rules to limit access.

Outcome: System security enhanced, reducing risk of network attacks.



Status: Fixed & Secured 