TASK : 7

Port Scanner – PoC

This Proof of Concept demonstrates a simple **Bash script** to scan specific ports on a user-supplied IP address using `nc` (netcat) with `timeout`.

Objective

- Scan **ports 20–25** (FTP, SSH, SMTP range) on a given IP.
- Report which ports are **open** or **closed**.
- Keep it simple, lightweight, and scriptable.

Script: portscan.sh

```
#!/bin/bash

TARGET=$1
echo "🔍 Scanning ports 20-25 on $TARGET"

for PORT in {20..25}; do
  timeout 1 bash -c "echo > /dev/tcp/$TARGET/$PORT" 2>/dev/null &&
    echo "✅ Port $PORT is OPEN" ||
    echo "❌ Port $PORT is CLOSED"
done
```

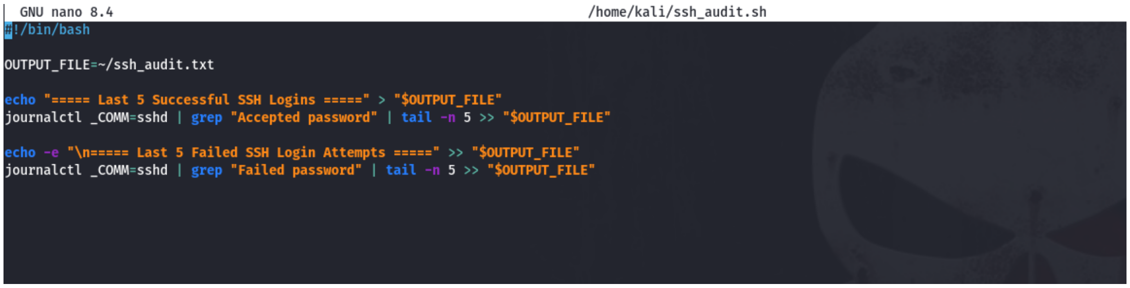
Alternative (using `nc`):

```
#!/bin/bash

TARGET=$1
echo "🔍 Scanning ports 20-25 on $TARGET"

for PORT in {20..25}; do
  timeout 1 nc -zv $TARGET $PORT &>/dev/null
```

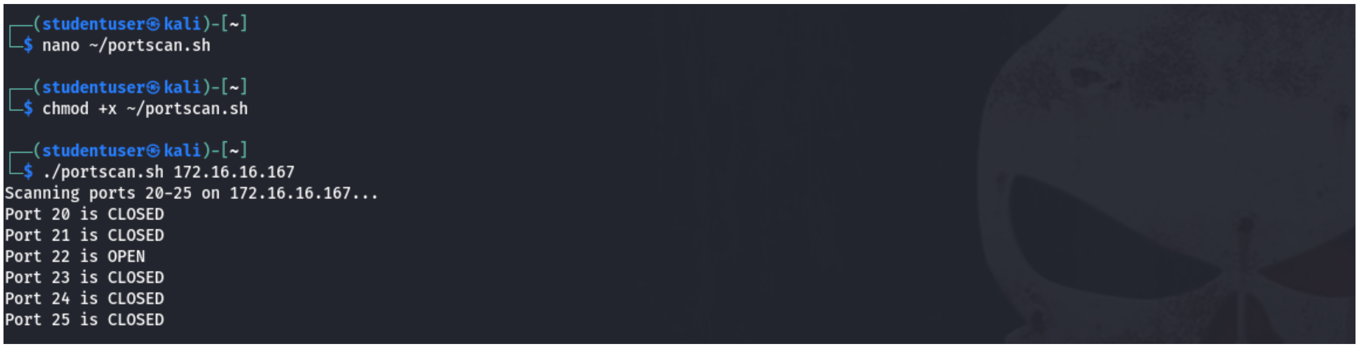
```
    if [ $? -eq 0 ]; then
        echo "✅ Port $PORT is OPEN"
    else
        echo "❌ Port $PORT is CLOSED"
    fi
done
```

```
chmod +x portscan.sh
./portscan.sh <target-ip>
```

Setup & Run



```
  GNU nano 8.4                                              /home/kali/ssh_audit.sh
#!/bin/bash

OUTPUT_FILE=~/ssh_audit.txt

echo "===== Last 5 Successful SSH Logins =====" > "$OUTPUT_FILE"
journalctl _COMM=sshd | grep "Accepted password" | tail -n 5 >> "$OUTPUT_FILE"

echo -e "\n===== Last 5 Failed SSH Login Attempts =====" >> "$OUTPUT_FILE"
journalctl _COMM=sshd | grep "Failed password" | tail -n 5 >> "$OUTPUT_FILE"
```



```
┌──(studentuser㉿kali)-[~]
└─$ nano ~/portscan.sh

┌──(studentuser㉿kali)-[~]
└─$ chmod +x ~/portscan.sh

┌──(studentuser㉿kali)-[~]
└─$ ./portscan.sh 172.16.16.167
Scanning ports 20-25 on 172.16.16.167...
Port 20 is CLOSED
Port 21 is CLOSED
Port 22 is OPEN
Port 23 is CLOSED
Port 24 is CLOSED
Port 25 is CLOSED
```