Practical Assignment Report

Nmap & Scapy Lab Documentation

1. Introduction

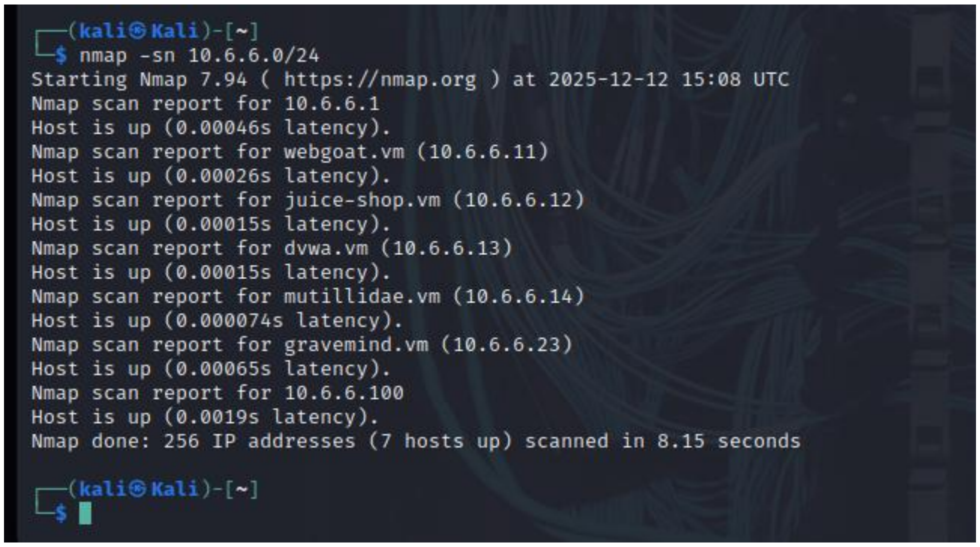
This assignment documents the reproduction of the Nmap and Scapy practical labs covered during class. The goal was to rerun all demonstrations, understand how each tool functions, and summarize the learning outcomes, challenges, and relevance of these tools in real world cybersecurity operations.

2. Nmap Lab Activities

2.1 Host Discovery

Commands practiced:

- nmap -sn <target>



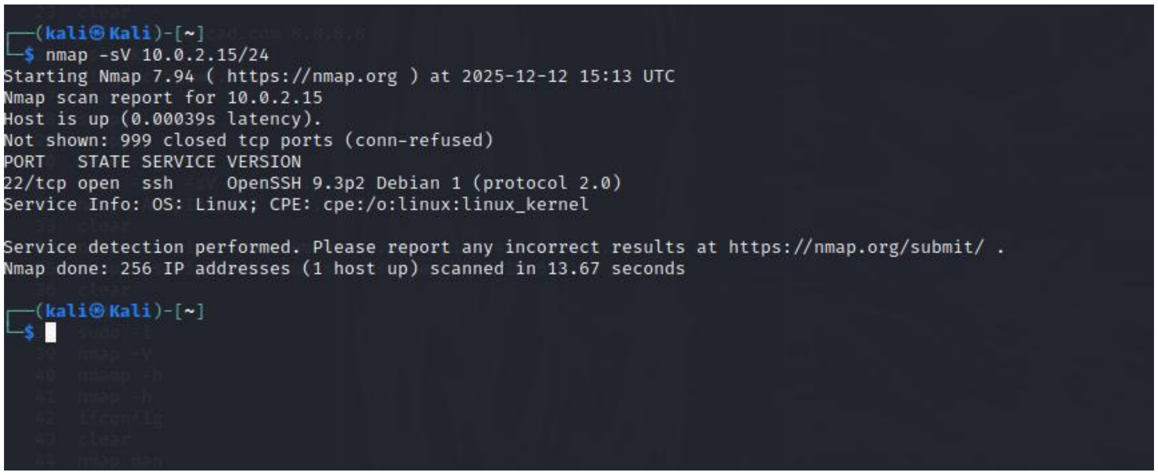
```
┌──(kali㉿Kali)-[~]
└─$ nmap -sn 10.6.6.0/24
Starting Nmap 7.94 ( https://nmap.org ) at 2025-12-12 15:08 UTC
Nmap scan report for 10.6.6.1
Host is up (0.00046s latency).
Nmap scan report for webgoat.vm (10.6.6.11)
Host is up (0.00026s latency).
Nmap scan report for juice-shop.vm (10.6.6.12)
Host is up (0.00015s latency).
Nmap scan report for dvwa.vm (10.6.6.13)
Host is up (0.00015s latency).
Nmap scan report for mutillidae.vm (10.6.6.14)
Host is up (0.000074s latency).
Nmap scan report for gravemind.vm (10.6.6.23)
Host is up (0.00065s latency).
Nmap scan report for 10.6.6.100
Host is up (0.0019s latency).
Nmap done: 256 IP addresses (7 hosts up) scanned in 8.15 seconds

┌──(kali㉿Kali)-[~]
└─$ 
```

Purpose: Identify live hosts on a network without scanning ports.
Observation: ARP and ICMP ping sweeps are efficient for mapping active devices.

2.3 Service Version Detection

Command:

- nmap -sV <target>



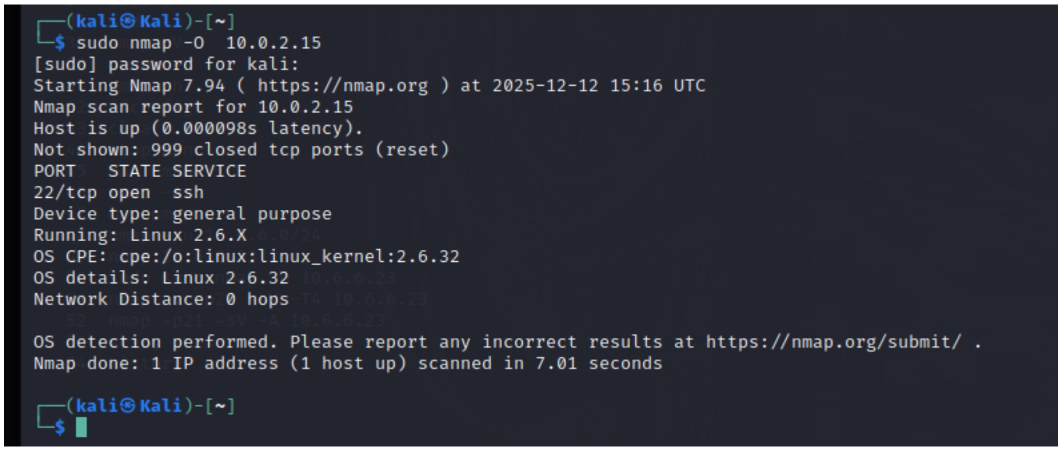
```
┌──(kali㉿Kali)-[~]
└─$ nmap -sV 10.0.2.15/24
Starting Nmap 7.94 ( https://nmap.org ) at 2025-12-12 15:13 UTC
Nmap scan report for 10.0.2.15
Host is up (0.00039s latency).
Not shown: 999 closed tcp ports (conn-refused)
PORT   STATE SERVICE VERSION
22/tcp open  ssh     OpenSSH 9.3p2 Debian 1 (protocol 2.0)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 256 IP addresses (1 host up) scanned in 13.67 seconds

┌──(kali㉿Kali)-[~]
└─$ 
```

Purpose: Identify versions of running services.
Observation: Helps assess vulnerabilities linked to outdated software.

2.4 OS Fingerprinting

Command:

- nmap -O <target>



```
┌──(kali㉿Kali)-[~]
└─$ sudo nmap -O  10.0.2.15
[sudo] password for kali:
Starting Nmap 7.94 ( https://nmap.org ) at 2025-12-12 15:16 UTC
Nmap scan report for 10.0.2.15
Host is up (0.000098s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE SERVICE
22/tcp open  ssh
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.32
OS details: Linux 2.6.32
Network Distance: 0 hops

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.01 seconds

┌──(kali㉿Kali)-[~]
└─$ 
```

Purpose: Detect operating system based on TCP/IP behavior.
Observation: Useful for attacker profiling and target defense strategies.

2.5 Aggressive Scan

Command:

- nmap -A <target>

Purpose: Combine OS detection, service detection, scripts, and traceroute.
Observation: Powerful but noisy should be used carefully in real environments.

2.6 Basic Nmap Scripting Engine (NSE) Usage

3. Scapy Lab Activities

3.1 Packet Crafting

Example:

packet = IP(dst="192.168.1.10")/TCP(dport=80, flags="S")
send(packet)

Purpose: Create custom packets for testing firewalls or protocols.

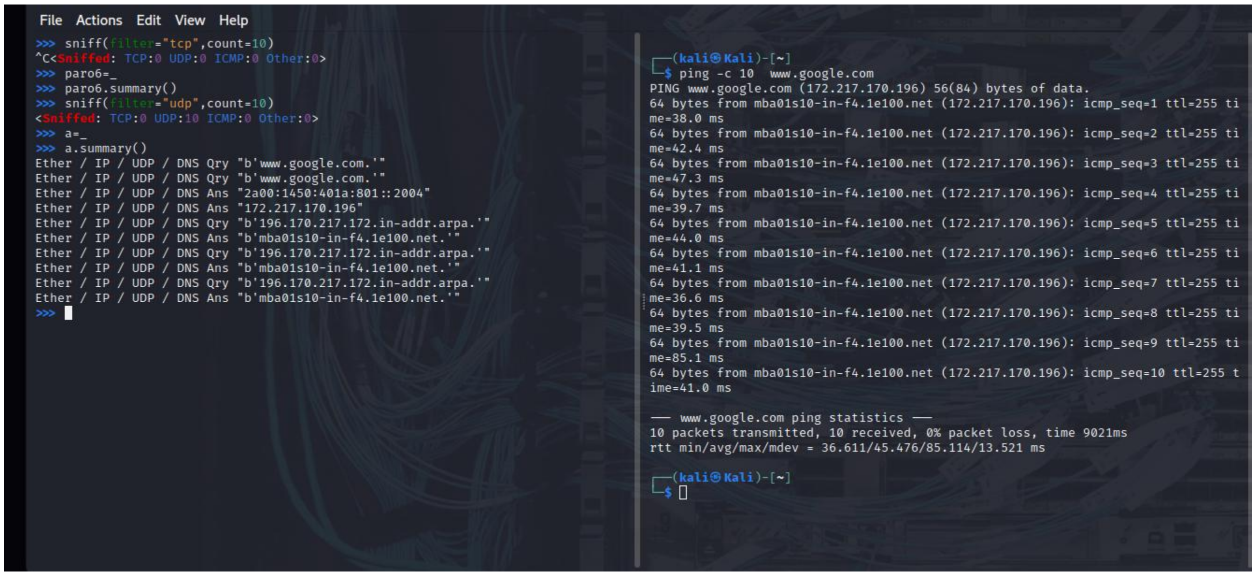
3.2 Packet Sending

- Used send(), sr(), sr1()
  Observation: Allows low-level control over packet transmission.

3.3 Packet Sniffing

Example:

sniff(filter="tcp", count=10)



```
>>> sniff(filter="tcp",count=10)
^C<Sniffed: TCP:0 UDP:0 ICMP:0 Other:0>
>>> paro6=_
>>> paro6.summary()
>>> sniff(filter="udp",count=10)
<Sniffed: TCP:0 UDP:10 ICMP:0 Other:0>
>>> a=_
>>> a.summary()
Ether / IP / UDP / DNS Qry "b'www.google.com.'"
Ether / IP / UDP / DNS Qry "b'www.google.com.'"
Ether / IP / UDP / DNS Ans "2a00:1450:401a:801::2004"
Ether / IP / UDP / DNS Ans "172.217.170.196"
Ether / IP / UDP / DNS Qry "b'196.170.217.172.in-addr.arpa.'"
Ether / IP / UDP / DNS Ans "b'mba01s10-in-f4.1e100.net.'"
Ether / IP / UDP / DNS Qry "b'196.170.217.172.in-addr.arpa.'"
Ether / IP / UDP / DNS Ans "b'mba01s10-in-f4.1e100.net.'"
Ether / IP / UDP / DNS Qry "b'196.170.217.172.in-addr.arpa.'"
Ether / IP / UDP / DNS Ans "b'mba01s10-in-f4.1e100.net.'"
>>>
```

```
┌──(kali㉿Kali)-[~]
└─$ ping -c 10  www.google.com
PING www.google.com (172.217.170.196) 56(84) bytes of data.
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=1 ttl=255 ti
me=38.0 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=2 ttl=255 ti
me=42.4 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=3 ttl=255 ti
me=47.3 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=4 ttl=255 ti
me=39.7 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=5 ttl=255 ti
me=44.0 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=6 ttl=255 ti
me=41.1 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=7 ttl=255 ti
me=36.6 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=8 ttl=255 ti
me=39.5 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=9 ttl=255 ti
me=85.1 ms
64 bytes from mba01s10-in-f4.1e100.net (172.217.170.196): icmp_seq=10 ttl=255 t
ime=41.0 ms

— www.google.com ping statistics —
10 packets transmitted, 10 received, 0% packet loss, time 9021ms
rtt min/avg/max/mdev = 36.611/45.476/85.114/13.521 ms

┌──(kali㉿Kali)-[~]
└─$
```

Purpose: Capture and analyze live network traffic.
Observation: Required elevated permissions; useful for debugging and intrusion analysis.

4. Summary & Reflection

4.1 What I Learned

- How to enumerate hosts, services, and operating systems using Nmap.
- How to craft, send, and sniff packets using Scapy.
- How to analyze network traffic and interpret protocol behaviors.
- How Nmap and Scapy complement each other one for scanning, one for packet-level control.

4.2 What Was Challenging

- Running Scapy sniffing required root/administrator privileges.
- Understanding some advanced Nmap flags and NSE scripts.
- Crafting packets correctly without breaking protocol structure.
- Interpreting raw packet output during Scapy analysis.

4.3 Real World Cybersecurity Importance

- **Nmap** is essential for penetration testing, vulnerability assessment, network auditing, and asset discovery.
- **Scapy** is crucial for incident response, protocol testing, exploit development, and understanding how attacks like spoofing or scanning work at the packet level.
- Both tools help cybersecurity professionals detect weaknesses, verify defenses, and understand attacker behavior.

5. Conclusion

Reproducing the Nmap and Scapy labs strengthened my technical skills in network scanning, reconnaissance, packet crafting, and analysis. These tools play a critical role in modern cybersecurity practices, and the hands on experience provided deeper insight into both offensive and defensive security operations.