MODULE 3: INFORMATION GATHERING AND VULNERABILITY SCANNING

Scapy packet-sniffing lab

In this lab I used Scapy as an interactive packet-manipulation tool in Python to discover hosts and observe traffic during a Cisco ethical hacking session with ParoCyber. The key workflow involved listing protocol layers, sniffing packets on a specific interface, and summarizing captured traffic.

- **ls()** and **ls(IP)** were used to inspect Scapy protocol layers and available IP-layer fields, which helps understand how packets are constructed before crafting or analysing them.

- **sniff()** was used several times, first with a default capture and later with options such as **iface="br-internal"** to bind to a lab bridge interface and **filter="icmp"**, **count=3** to capture only a small number of ICMP packets generated by pings.

- Captured packets were stored in variables like par0, par2, and par3, then reviewed with **par0.summary()** etc., which prints a one-line summary per packet and is useful for quick reconnaissance of what protocols and hosts are active on the segment.

During the exercise, ICMP echo requests were generated using commands such as **ping google.com** or **ping -c 4 google.com** so that Scapy could see live traffic flowing over the selected interface. By changing interfaces and filters, it became clear how different subnets and protocols appear in the capture, reinforcing concepts like interfaces, IP addressing, and basic protocol behaviour. Proper stopping of **sniff()** with **Ctrl+C** was also practiced to avoid uncontrolled captures and large in-memory buffers.

Tcpdump and Wireshark capture lab

Another part of the lab focused on using tcpdump to create capture files and then analysing them in Wireshark. IP configuration was checked with **ifconfig** and **/etc/resolv.conf** to verify the local address (for example 10.0.2.15) and DNS settings before starting the capture.

- The command **sudo tcpdump -i eth0 -s0 -w lab.pcap** was used to capture all traffic on interface **eth0** with a snap length of 0, meaning full packets were written to a .pcap file for later analysis.

- After stopping the capture with **Ctrl+C,** ls was used to confirm that **lab.pcap** existed, and then Wireshark was launched to open the file and inspect individual frames, follow streams, and apply display filters.

This workflow demonstrated the difference between command-line capture (tcpdump) and GUI analysis (Wireshark), as well as the importance of storing captures in standard formats such as **PCAP/PCAPNG** for repeatable investigations and reporting.

Nmap host and service enumeration lab

Nmap was used for host discovery, port scanning, OS detection, and SMB enumeration against the lab target **10.6.6.23**. The exercises showed how different scan types reveal increasing amounts of information useful for ethical hacking and penetration testing.

- **nmap -V** verified the installed Nmap version, an important step to know which features and NSE scripts are available.

- **nmap -sn 10.6.6.0/24** performed a ping/host discovery scan over the subnet to identify live hosts without doing a full port scan, which is faster and less noisy.

- **sudo nmap -O 10.6.6.23** enabled OS detection by analyzing TCP/IP stack behaviour, giving an estimate of the operating system and device type running on the target.

A more detailed scan combined several options: **nmap -p21 -sV -A -T4 10.6.6.23**, where -p21 restricted the scan to FTP port 21, -sV attempted service/version detection, -A enabled OS detection and common scripts, and -T4 increased speed at the cost of more aggressive timing. Additional scans like **nmap -A -p139,445 10.6.6.23** and **nmap --script smb-enum-shares.nse -p445 10.6.6.23** focused on SMB ports 139 and 445, using NSE scripts to enumerate Windows file shares and identify SMB-related weaknesses. The final step I used **smbclient //10.6.6.23/print$ -N** to connect anonymously to one of the discovered shares, illustrating how enumeration results directly support exploitation steps in an ethical hacking methodology.

Summary of learning outcomes

From these labs I practiced end-to-end network reconnaissance: capturing raw traffic with **tcpdump** and **Wireshark**, interactively sniffing and examining packets with Scapy, and scanning hosts and services with Nmap plus SMB utilities like **smbclient**. Together, these tools form a basic toolkit for ethical hacking, enabling me to move from information gathering (Nmap), to traffic observation (Scapy/tcpdump), and finally to protocol-specific enumeration and access (SMB scripts and clients) in a structured, defensible penetration-testing workflow.