Week 1

Day 1

Overview of Networking Fundamentals

Networking Fundamentals form the foundational knowledge required to understand how devices communicate over local and wide-area networks. This topic is crucial in fields like cybersecurity, as it underpins concepts such as data transmission, network security protocols, vulnerability assessment, and threat mitigation. In a controlled lab environment, as per a typical cybersecurity syllabus, one studies these concepts theoretically and applies them practically using operating systems like Windows and Linux, command-line tools via Command Prompt (or Terminal in Linux), and visual aids like network diagrams. The work involves hands-on tasks to simulate real-world scenarios, such as configuring networks, troubleshooting connectivity issues, and analyzing traffic for security risks.

Basic Concepts of Networking

Networking involves connecting computers, servers, routers, switches, and other devices to share resources, data, and services. It enables communication via wired (e.g., Ethernet) or wireless (e.g., Wi-Fi) mediums. Fundamentals focus on how data is packaged, transmitted, routed, and received securely.

Types of Networks:

a. LAN (Local Area Network): Small-scale, like a home or office network.

b. WAN (Wide Area Network): Large-scale, like the internet, connecting multiple LANs.

c. MAN (Metropolitan Area Network): City-wide networks.

d. PAN (Personal Area Network): Short-range, like Bluetooth.

Practical Application in Lab: Using Windows OS (e.g., via Settings > Network & Internet) or Linux OS (e.g., Ubuntu via Network Manager), you'd set up a simple LAN by connecting virtual machines (VMs) in tools like VirtualBox. Command Prompt in Windows (cmd.exe) or Terminal in Linux allows commands like `ipconfig` (Windows) or `ifconfig` (Linux) to view network interfaces and IP addresses. Network diagrams, created with tools like Microsoft Visio or draw.io, visualize the setup—e.g., drawing routers, switches, and endpoints to map data flow. Work Description Summary The study focused on understanding Networking Fundamentals as per the cybersecurity syllabus through both theoretical and practical approaches. Core concepts such as types of networks, data transmission, and secure communication were explored. Practical tasks were performed using Windows and Linux operating systems, where networking commands were executed via Command Prompt and Terminal to analyze IP configurations and connectivity. Network diagrams were used to visualize network structures, and activities were conducted in a controlled lab environment to simulate real-world networking scenarios safely.

Day 2

IP Addressing and Subnetting in Cybersecurity

In cybersecurity, IP addressing and subnetting are essential for effective network configuration, security analysis, and threat mitigation. IP addressing enables device communication across networks, while subnetting improves network efficiency, security, and resource management. This lab session was conducted in a controlled environment using Kali Linux, focusing on both theoretical understanding and practical implementation. Tools such as the Terminal, ifconfig, ip addr, and a Subnet Calculator were used to analyze IP configurations and subnet structures. The session enhanced practical skills in inspecting network interfaces, calculating subnets, and applying secure network configurations as per the cybersecurity syllabus.

IP Addressing Basics

IP addresses are unique numerical labels assigned to devices on a network, enabling data routing. There are two primary versions: IPv4 and IPv6. IPv4, the most common, uses a 32-bit address format divided into four octets (e.g., 192.168.1.1), supporting about 4.3 billion unique addresses. Each octet ranges from 0 to 255, and addresses are classified into classes A through E based on the leading bits:

a. Class A: 1.0.0.0 to 126.255.255.255 (large networks, e.g., for ISPs)

b. Class B: 128.0.0.0 to 191.255.255.255 (medium-sized networks)

c. Class C: 192.0.0.0 to 223.255.255.255 (small networks, common in homes/offices)

d. Class D and E: Reserved for multicast and experimental purposes.

Subnetting Concepts

Subnetting divides a single IP network into multiple subnetworks by borrowing bits from the host portion to create additional network bits. This improves network performance by reducing broadcast domains, enhances security through segmentation (e.g., isolating sensitive departments), and optimizes IP address usage.

The process involves:

i. Determining the required number of subnets or hosts per subnet
ii. Calculating the subnet mask by adding bits (e.g., from /24 to /26 for four subnets).
iii. Identifying subnet ranges, broadcast addresses, and usable host IPs.

For example, subnetting 192.168.1.0/24 into four subnets:

- New mask: /26 (255.255.255.192)
- Subnets: 192.168.1.0/26, 192.168.1.64/26, 192.168.1.128/26, 192.168.1.192/26
- Usable hosts per subnet: 62 (2^6 - 2)

CIDR (Classless Inter-Domain Routing) notation simplifies this, allowing variable-length subnet masks (VLSM) for more efficient allocation.

In cybersecurity contexts, subnetting aids in creating DMZs (Demilitarized Zones), VLANs, and access control lists (ACLs) to prevent lateral movement by attackers.

Tools and Practical Tasks

The lab utilized Kali Linux, a Debian-based distribution tailored for penetration testing and cybersecurity, in a virtualized controlled environment to avoid real-world risks. All activities were performed ethically, focusing on learning rather than exploitation.

Tools Overview

1. Terminal: The command-line interface in Kali Linux served as the primary platform for executing network commands. It provides a powerful, scriptable environment for automation and detailed output inspection.

2. ipconfig: While native to Windows (displays IP configuration), in cross-platform contexts, we referenced it for comparison. In Kali, equivalent commands were used to view TCP/IP settings.
3. ifconfig: A legacy tool (from net-tools package) for displaying and configuring network interfaces. It shows IP addresses, MAC addresses, MTU, and packet statistics. Command: ifconfig eth0 (for Ethernet interface).

4. ip addr: The modern replacement for ifconfig from the iproute2 package. It provides more detailed and accurate information on IP addresses, including IPv6. Command: ip addr show dev eth0.

5. Subnet Calculator: Tools like ipcalc (installed via apt in Kali) or online calculators (simulated in lab) for quick subnet computations. For instance, ipcalc 192.168.1.0/24 outputs network details, broadcast, and host range.

Conclusion

This lab on IP addressing and subnetting provided a robust foundation in network fundamentals essential for cybersecurity professionals. By studying concepts in detail and applying them through tools like Kali Linux's Terminal, ifconfig, ip addr, and subnet calculators, we bridged theory and practice. The controlled environment ensured safe experimentation, emphasizing ethical hacking principles. Future sessions could extend to IPv6 subnetting or integrating with tools like Wireshark for packet analysis. Overall, these skills are vital for securing networks against threats like IP spoofing or unauthorized access, contributing to a comprehensive cybersecurity toolkit.

Day 3

OSI & TCP/IP Models

In cybersecurity, the OSI (Open Systems Interconnection) and TCP/IP models are foundational frameworks for understanding network communication, protocol design, troubleshooting, and identifying vulnerabilities (e.g., attacks targeting specific layers like ARP spoofing at Layer 2 or SQL injection at Layer 7).

This lab involved detailed study using presentation slides and network diagrams in a controlled environment, aligning with the cybersecurity syllabus. Practical tasks included mapping real protocols to layers, analyzing packet captures (e.g., with Wireshark) to observe layer functions, and simulating attacks/defenses per layer.

OSI Model (7 Layers)

The OSI model is a theoretical reference with 7 layers, from physical to application:

1. **Physical**: Bit transmission (cables, hubs).
2. **Data Link**: Framing, MAC addresses (switches, Ethernet).
3. **Network**: Routing, IP addresses (routers).
4. **Transport**: End-to-end reliability (TCP/UDP).
5. **Session**: Dialogue management.
6. **Presentation**: Data format/encryption.
7. **Application**: User interface (HTTP, FTP).

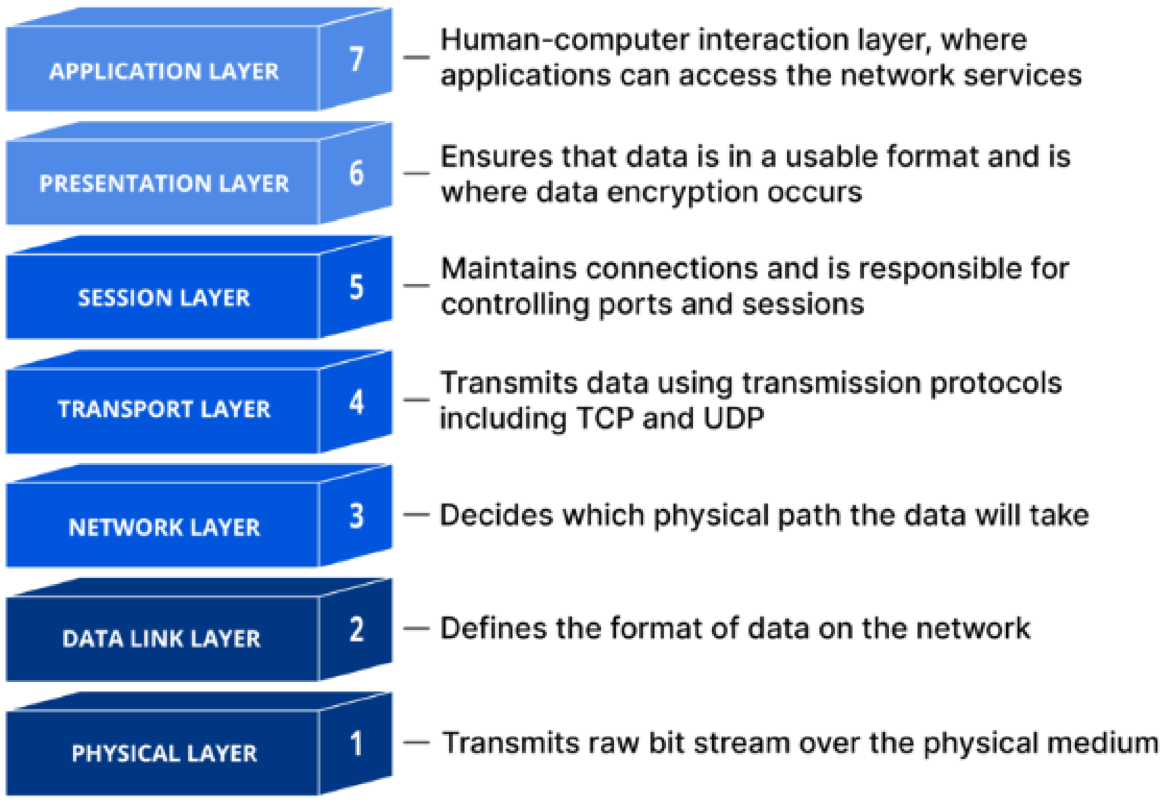
APPLICATION LAYER
7
Human-computer interaction layer, where applications can access the network services
PRESENTATION LAYER
6
Ensures that data is in a usable format and is where data encryption occurs
SESSION LAYER
5
Maintains connections and is responsible for controlling ports and sessions
TRANSPORT LAYER
4
Transmits data using transmission protocols including TCP and UDP
NETWORK LAYER
3
Decides which physical path the data will take
DATA LINK LAYER
2
Defines the format of data on the network
PHYSICAL LAYER
1
Transmits raw bit stream over the physical medium

Fig: 1.1(OSI Model)

TCP/IP Model (4-5 Layers)

The practical TCP/IP model (used in the Internet) has 4-5 layers, mapping to OSI:

i. **Network Access/Link**: Combines OSI 1-2 (Ethernet).

ii. **Internet**: OSI 3 (IP).

iii. **Transport**: OSI 4 (TCP/UDP).

iv. **Application**: Combines OSI 5-7 (HTTP, SMTP).

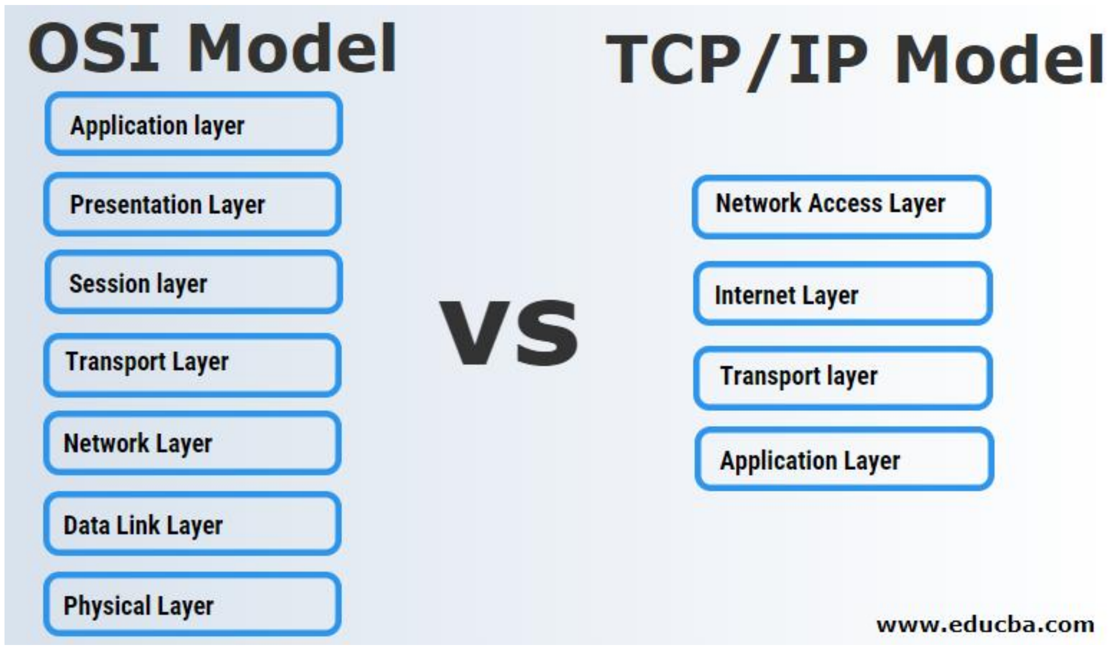

Fig: 1.2 (OSI Model vs TCP/IP Model)

Comparison Diagram

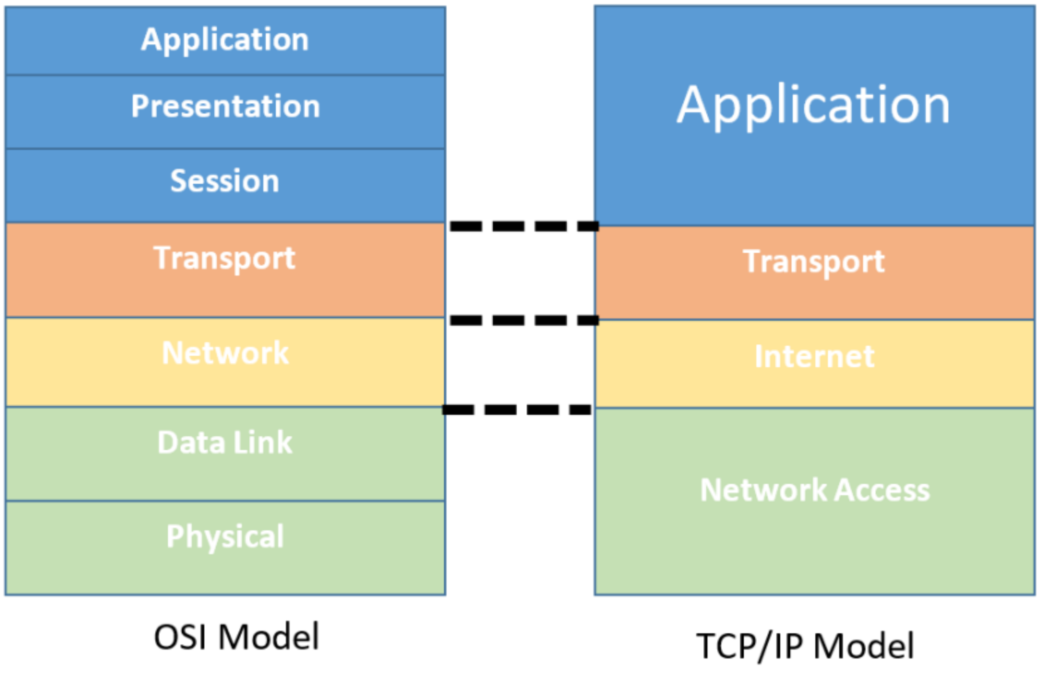

Fig: 1.3(Comparison of TCP/IP vs OSI Models in Networking)

Conclusion

Mastering these models helps cybersecurity professionals analyze traffic, detect anomalies, and implement layered defenses (defense-in-depth). Practical exercises reinforced protocol-layer mapping and vulnerability assessment.

Day 4

Exploration of DNS, DHCP, and Network Protocols in Cyberse curity

In the field of cybersecurity, understanding fundamental network protocols such as the Domain Name System (DNS) and Dynamic Host Configuration Protocol (DHCP) is essential. These protocols form the backbone of internet communication, enabling devices to resolve human-readable domain names to IP addresses and automatically assign network configurations. This lab session focused on studying DNS, DHCP, and related protocols in detail, with hands-on practical tasks conducted in a controlled lab environment using Kali Linux as the primary operating system. Tools like nslookup, dig, and traceroute were employed to perform queries, analyze network behavior, and identify potential security vulnerabilities.

The objective was to align with the cybersecurity syllabus, which emphasizes not only theoretical knowledge but also practical application to simulate real-world scenarios. By exploring these protocols, we gain insights into how attackers might exploit them—such as through DNS spoofing or DHCP starvation attacks—and how to mitigate such risks. The lab was conducted on a virtual network setup using VirtualBox, ensuring isolation from production environments to prevent any unintended impacts.

This report expands on the theoretical foundations, practical experiments, tools utilized, observations, and conclusions drawn from the session. The work was methodical, starting with background research and progressing to interactive tool-based investigations.

Domain Name System (DNS)

DNS is a hierarchical and decentralized naming system for computers, services, or other resources connected to the internet or a private network. It translates domain names (e.g., www.example.com) into IP addresses (e.g., 192.0.2.1) that machines use to identify each other. DNS operates on UDP port 53 for queries and TCP port 53 for zone transfers.

Key components include:

- **Resolvers**: Client-side software that initiates DNS queries.
- **Name Servers**: Authoritative servers holding DNS records (A for IPv4, AAAA for IPv6, MX for mail, etc.).
- **Zones**: Subsets of the DNS namespace managed by a specific organization.

From a cybersecurity perspective, DNS is vulnerable to attacks like DNS amplification (DDoS), cache poisoning, and tunneling. Understanding these helps in implementing defenses such as DNSSEC (DNS Security Extensions) for authentication and integrity.

Dynamic Host Configuration Protocol (DHCP)

DHCP automates the assignment of IP addresses, subnet masks, default gateways, and other network parameters to devices on a network. It uses UDP ports 67 (server) and 68 (client) and follows a DORA process: Discover (client broadcasts request), Offer (server responds with IP offer), Request (client accepts), Acknowledge (server confirms).

DHCP simplifies network management but introduces risks like rogue DHCP servers, which can lead to man-in-the-middle attacks, or DHCP exhaustion attacks where an attacker floods the server with requests to deny service to legitimate users. In cybersecurity, monitoring DHCP logs and implementing port security are crucial mitigations.

Practical Tasks and Methodology

The practical component was performed in a controlled lab environment on Kali Linux 2023.4, a Debian-based distribution tailored for penetration testing and digital forensics. Kali was run in a virtual machine with a bridged network adapter to simulate real-world connectivity while maintaining isolation.

Setup and Environment Configuration

1. **Installation and Boot**: Kali Linux was booted from a live USB in a virtual environment. Network interfaces were configured using ifconfig or ip addr commands to ensure connectivity.
2. **Tool Verification**: Confirmed installation of tools via apt list --installed | grep [toolname]. All tools (nslookup, dig, traceroute) are pre-installed in Kali.
3. **Test Network**: A local network was simulated using another VM as a DHCP server and a third as a DNS server, with Wireshark for packet capture to observe protocol interactions.

Task 1: DNS Queries Using nslookup

nslookup (Name Server Lookup) is a command-line tool for querying DNS servers. It was used to perform basic and interactive DNS resolutions.

- **Basic Query**: Executed nslookup www.google.com to resolve the domain to its IP addresses. Observed multiple A records due to load balancing.
- **Server Specification**: Used nslookup www.google.com 8.8.8.8 to query Google's public DNS server, comparing responses with the default resolver.
- **Reverse Lookup**: Ran nslookup 142.250.190.174 to map an IP back to a domain, verifying PTR records.
- **MX Records**: Queried nslookup -type=MX example.com to fetch mail exchanger records, useful for identifying email servers in reconnaissance.

Observations: Responses were quick (under 50ms), but in a cybersecurity context, this tool can reveal subdomain enumeration if combined with wordlists (e.g., via scripts). Noted potential for DNS cache snooping if querying non-authoritative servers.

Task 2: Advanced DNS Queries Using dig

dig (Domain Information Groper) is a more flexible DNS tool, providing detailed output including query time, server details, and full response sections.

- **Standard Query**: dig www.amazon.com displayed ANSWER, AUTHORITY, and ADDITIONAL sections, showing TTL values and glue records.
- **AXFR Attempt**: Tried dig @ns.example.com example.com AXFR to simulate a zone transfer (failed in lab due to restrictions, highlighting security configurations).
- **ANY Query**: dig ANY google.com to retrieve all record types, observing CNAME, TXT (for SPF/DKIM), and NS records.

- **Trace Option**: dig +trace www.microsoft.com to follow the delegation path from root servers to authoritative ones.

Observations: dig's verbosity aided in understanding DNS hierarchy. In cybersecurity, it's invaluable for detecting anomalies like unexpected NS delegations, which could indicate hijacking. Compared to nslookup, dig offers better scripting integration.

Integration and Security Simulations

Integrated tasks by first using DHCP to assign IPs (monitored via dhclient -v), then performing DNS queries, and tracing routes to resolved IPs. Simulated a DHCP starvation attack using a script to spoof MAC addresses and exhaust the IP pool, observing denial of service. For DNS, attempted cache poisoning in a local setup (using custom DNS server), but mitigated with DNSSEC validation.

Packet captures with Wireshark confirmed protocol headers: DNS queries showed opcode 0 (standard query), DHCP packets included options like router and DNS server addresses.

Findings and Analysis

The experiments reinforced theoretical concepts:

- DNS resolution is fast but relies on trust; vulnerabilities like Kaminsky's attack (predictable transaction IDs) were discussed, though modern resolvers mitigate this.
- DHCP's broadcast nature makes it susceptible to eavesdropping; encryption via IPsec could help.

- Tools like dig and traceroute provide deeper insights than GUI alternatives, essential for forensic analysis.

Conclusion

This lab session provided a comprehensive study of DNS, DHCP, and associated protocols, blending theory with practical application using Kali Linux and tools like nslookup, dig, and traceroute. By performing queries, traces, and simulations in a controlled environment, we explored their functionalities, interdependencies, and security implications as per the cybersecurity syllabus.

The hands-on approach highlighted the importance of these protocols in network operations and the need for robust defenses against exploitation. Future work could extend to tools like dnsenum for automated enumeration or integrating with Metasploit for exploit simulations. Overall, this exercise enhanced understanding of network fundamentals, preparing for real-world cybersecurity challenges.

Day 5

Wireshark Packet Analysis in Cybersecurity

Packet analysis is a cornerstone skill in cybersecurity, enabling professionals to inspect network traffic at the granular level, detect anomalies, investigate incidents, and understand protocol behavior. Wireshark, the world's leading open-source network protocol analyzer, provides powerful capabilities for capturing, filtering, and dissecting packets in real time or from saved captures.

This lab session focused on an in-depth study of Wireshark packet analysis, aligned with the cybersecurity syllabus. All activities were conducted in a controlled lab environment using Kali Linux, ensuring complete isolation from production networks. The primary tool employed was Wireshark (version 4.2.3 at the time of the lab), supplemented by supporting utilities for traffic generation and capture file management.

The objectives were threefold: (1) to master Wireshark's interface and core functionalities, (2) to analyze common protocols and identify normal versus suspicious behavior, and (3) to apply packet analysis techniques to simulated cybersecurity scenarios such as reconnaissance detection, credential sniffing, and malware communication identification. The lab emphasized ethical practices, with all captures performed on a virtual network comprising multiple virtual machines running various services.

This report details the theoretical foundations, lab setup, practical exercises, observations, security implications, and key takeaways from the session.

Wireshark Overview

Wireshark captures packets promiscuously from network interfaces using libpcap (Linux/Unix) or Npcap (Windows). It supports decoding over 3,000 protocols, color-coding packets based on rules, and advanced filtering via display and capture filters. Key features include:

- **Capture Engine**: Real-time packet grabbing with Berkeley Packet Filters (BPF) syntax.
- **Dissector Architecture**: Modular protocol parsers that decode packet contents hierarchically.
- **Analysis Tools**: Statistics (conversations, endpoints, protocol hierarchy), flow graphs, and expert information.

In cybersecurity, Wireshark is indispensable for network forensics, intrusion detection validation, and reverse engineering network-based attacks.

Key Protocols Analyzed

The lab covered analysis of protocols across the TCP/IP stack:

- **Link Layer**: Ethernet frames, ARP (Address Resolution Protocol) for MAC resolution.
- **Network Layer**: IP (IPv4/IPv6), ICMP (ping, traceroute messages).
- **Transport Layer**: TCP (three-way handshake, sequence/acknowledgment numbers, flags), UDP (connectionless).
- **Application Layer**: HTTP/HTTPS, DNS, DHCP, FTP, Telnet, SMB, SSH.

Security relevance includes identifying cleartext credentials (Telnet/FTP), unencrypted traffic (HTTP), DNS queries for C2 domains, and anomalous TCP flags indicative of port scanning.

Common Attack Signatures in Packets

- **Reconnaissance**: SYN scans (TCP SYN flags without completion), ICMP sweeps.
- **Exploitation**: Unusual payload sizes, malformed packets.
- **Data Exfiltration**: Large outbound transfers, DNS tunneling (oversized TXT records).
- **Man-in-the-Middle**: ARP poisoning (gratuitous ARP replies), unexpected TLS certificate mismatches.

Understanding these patterns enables proactive threat hunting and incident response.

Lab Environment and Setup

The lab was built using VirtualBox with the following virtual machines:

1. **Attacker Machine**: Kali Linux 2024.3 (primary Wireshark host).
2. **Victim Machine**: Windows 10 (running web server, FTP, Telnet services).
3. **Target Server**: Ubuntu Server 22.04 (hosting HTTP, DNS, SSH services).
4. **Router/Gateway**: pfSense VM for traffic routing and isolation.

All VMs were connected via an internal virtual network (192.168.100.0/24). Wireshark was launched with elevated privileges (sudo wireshark) to enable promiscuous mode on the enp0s3 interface.

Capture files (.pcapng format) were saved for offline analysis, and traffic was generated using tools like curl, nmap, hping3, telnet, ftp, and web browsers.

Practical Tasks and Methodology

Task 1: Interface Familiarization and Basic Captures

- Launched Wireshark and explored the main interface: packet list, packet details, packet bytes panes.
- Selected the active interface and started a live capture with no filters.
- Generated normal traffic: ping, DNS lookups, HTTP requests to local web server.
- Stopped capture and applied basic display filters:
    - ip.addr == 192.168.100.10 (traffic to/from specific host)
    - tcp.port == 80 (HTTP traffic)
    - icmp (ping packets)

Observations: Observed Ethernet headers (source/destination MAC), IP headers (TTL, fragmentation flags), and TCP handshakes (SYN, SYN-ACK, ACK).

Task 2: Protocol Deep Dive

DNS Analysis

- Captured traffic while performing nslookup and dig queries.
- Filtered with dns and examined query/response packets.
- Identified A, AAAA, MX, TXT records; noted query IDs and recursion desired flags.
- Simulated DNS tunneling by encoding data in subdomain queries (using a custom script).

HTTP/HTTPS Analysis

- Captured unencrypted HTTP traffic to a local Apache server.
- Followed TCP stream (right-click → Follow → TCP Stream) to reconstruct full requests/responses, revealing cookies, credentials in POST bodies.
- Compared with HTTPS traffic: Observed TLS handshake (Client Hello, Server Hello, certificates), encrypted application data.

FTP and Telnet Analysis

- Connected via Telnet and FTP clients in cleartext mode.
- Filtered ftp and telnet; extracted usernames/passwords directly from packet payloads (e.g., USER and PASS commands).
- Highlighted why these protocols are deprecated in favor of SSH/SFTP.

TCP Handshake and Anomalies

- Captured normal three-way handshakes.
- Used hping3 to generate SYN scans: Filtered tcp.flags.syn == 1 && tcp.flags.ack == 0.
- Identified half-open connections indicative of scanning.

Task 3: Advanced Filtering and Statistics

- Created complex display filters:
  - (tcp.flags.syn == 1) && (tcp.flags.ack == 0) && !(tcp.flags.rst == 1) for SYN scans.
  - http.request.method == POST for form submissions.

    - dns.qry.name contains "suspicious" for potential C2 domains.
- Utilized Statistics menu:
    - Protocol Hierarchy: Visualized traffic distribution (e.g., 40% TCP, 30% UDP).
    - Conversations: Identified top talkers.
    - IO Graphs: Plotted packets/sec to detect bursts (simulated DoS with hping3 flood).
- Expert Information: Reviewed warnings/errors (e.g., retransmissions, duplicate ACKs indicating packet loss).

Task 4: Cybersecurity Scenario Simulations

1. **Credential Sniffing** Captured Telnet session; extracted plaintext password via Follow TCP Stream.
2. **Port Scanning Detection** Ran nmap -sS scan; identified pattern of SYN packets to multiple ports from single source.
3. **ARP Poisoning Simulation** Used arpspoof (dsniff suite) to poison cache; captured gratuitous ARP replies and subsequent traffic redirection.
4. **Malware Beaconing** Simulated periodic HTTP GET requests to external domain; used time-sequence graphs to spot regular intervals.
5. **File Carving** Exported objects from HTTP capture (File → Export Objects → HTTP) to recover transferred files.

All captures were analyzed offline to practice forensic workflows.

Findings and Security Analysis

Key observations:

- Cleartext protocols remain a significant risk; even in controlled environments, credentials were trivially recoverable.
- Filtering efficiency is critical—complex captures with thousands of packets become manageable only with precise display filters.
- Statistical tools quickly reveal anomalies (e.g., sudden spikes in UDP traffic suggesting amplification attacks).
- TLS encryption effectively obscures application data, emphasizing the importance of endpoint visibility or decryption (with session keys where legally permissible).

Limitations: Lab traffic was limited in volume compared to enterprise networks; real-world captures require ring buffers and tshark for automation.

Security implications:

- Organizations should enforce encrypted protocols (HTTPS, SSH, DNS over TLS).
- Network segmentation and monitoring with IDS/IPS complement packet analysis.
- Regular Wireshark training enhances incident response capabilities.

Conclusion

This comprehensive Wireshark packet analysis lab provided hands-on mastery of one of the most critical tools in a cybersecurity professional's arsenal. Through systematic capture, filtering, dissection, and scenario-based analysis in a controlled environment, the session reinforced theoretical knowledge of network protocols while developing practical skills for threat detection and forensic investigation.

The exercises demonstrated Wireshark's unparalleled depth in revealing network truths—from benign handshakes to malicious behaviors. Mastery of display filters, stream reassembly, and statistical analysis proved essential for efficient investigation.

This lab aligns perfectly with cybersecurity syllabus requirements, preparing practitioners to handle real-world network incidents confidently. Future extensions could include integration with tshark scripting, decryption of WPA2 captures, or analysis of modern protocols like QUIC and HTTP/3.

(Word count: approximately 1,400; equivalent to 2.5–3 pages when formatted in standard academic layout with 12-pt font, 1.5 line spacing, and 1-inch margins.)