

Day - 3: Networking Basics for Cyber Security

Basic Networking Concepts:

Networking is the way computers, phones, servers, IoT devices, etc. talk to each other to share information. All of this communication happens through networks — wired (Ethernet cable), wireless (Wi-Fi), or even mobile data.

IP Address (Internet Protocol Address):

An IP address is a logical (virtual/software-assigned) identifier for a device on a network. It operates at Layer 3 (Network Layer) of the OSI model.

- There are two main versions: IPv4 (e.g., 192.168.1.10 — 32-bit, written as four decimal numbers) and IPv6 (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334 — 128-bit, hexadecimal).
- **Purpose:** Enables routing of packets across different networks (internet/global scope). Routers use IP addresses to forward data from source to destination, even across the world.
- **Types:** Static (manually set, doesn't change) vs. Dynamic (assigned temporarily by DHCP).
- **Key point for cybersecurity:** IP addresses can be spoofed, and they're visible in packet headers — attackers can use them for targeting or tracking.

MAC Address (Media Access Control Address):

A MAC address is a physical/hardware identifier burned into the network interface card (NIC) by the manufacturer. It operates at Layer 2 (Data Link Layer) of the OSI model.

- Format: 48-bit (6 bytes), usually written as six pairs of hexadecimal digits separated by colons or hyphens (e.g., 00:1A:2B:3C:4D:5E).
- Purpose: Used for communication within the same local network (LAN). Switches use MAC addresses to forward frames to the correct device on the same segment.
- Key point: MAC addresses are unique per device (globally assigned by IEEE), but can be spoofed/changed in software. They don't cross routers — once a packet leaves the local network, the MAC address is replaced.

How IP & MAC work together: When sending data locally, a device uses ARP (Address Resolution Protocol) to map an IP address to the corresponding MAC address on the same network.

TCP and UDP (Transport Layer Protocols):

TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are both Layer 4 protocols in the TCP/IP model.

TCP — Connection-oriented, reliable protocol.

- Features: Establishes a connection before data transfer, ensures reliable delivery (error checking, retransmission of lost packets, ordered delivery, flow control, congestion control).
- Overhead: Higher (due to acknowledgments, sequence numbers, etc.).
- Use cases: Web browsing (HTTP/HTTPS), email (SMTP), file transfers (FTP) — where data integrity is critical.

UDP — Connectionless, unreliable protocol.

- Features: No connection setup, no guaranteed delivery, no ordering, no retransmission — just "fire and forget".
- Overhead: Very low (fixed 8-byte header).
- Use cases: DNS queries, video streaming, online gaming, VoIP — where speed/low latency matters more than perfect reliability (some packet loss is tolerable).

TCP Three-Way Handshake:

The TCP three-way handshake is the process TCP uses to establish a reliable, full-duplex connection between client and server before data transfer begins. The steps involved are as follows:

- SYN (Synchronize) Client sends a SYN segment to the server → "I want to connect" (includes client's initial sequence number, SYN flag = 1).
- SYN-ACK (Synchronize-Acknowledge) Server responds with SYN-ACK → "I acknowledge your request and want to connect too" (server's sequence number + acknowledgment of client's sequence number +1, SYN and ACK flags = 1).
- ACK (Acknowledge) Client sends ACK → "Got it, connection established" (acknowledges server's sequence number +1, ACK flag = 1). → Connection is now in ESTABLISHED state on both sides — data transfer can begin.

The steps are very important because:

- Ensures both sides are ready and synchronized (sequence numbers for ordering/retransmission).
- Prevents half-open connections and old duplicate packets from interfering.

DNS (Domain Name System):

DNS is like the "phonebook of the internet" — it translates human-readable domain names (e.g., www.google.com) into machine-readable IP addresses (e.g., 142.250.190.174).

How DNS works (step-by-step resolution):

- User types www.example.com in browser → query sent to local resolver (usually ISP or public like 8.8.8.8).
- Resolver checks cache → if not found, queries Root Name Servers (13 global clusters).
- Root → refers to TLD Name Servers (.com, .org, etc.).
- TLD → refers to Authoritative Name Servers for the domain (holds actual records).
- Authoritative server returns the IP address → resolver caches it and sends to client.
- Browser connects to the IP.

Key records: A (IPv4), AAAA (IPv6), CNAME (alias), MX (mail), NS (name server). Usually uses UDP port 53 (fast), falls back to TCP if response is large.

Packet Sniffing:

Packet sniffing (or packet capturing) is the process of intercepting and analysing data packets as they travel across a network.

- How it works: Tools like Wireshark put a network interface into promiscuous mode → captures all packets (not just those addressed to the device).
- Legitimate uses: Network troubleshooting, performance monitoring, security analysis (detecting anomalies).
- Malicious uses: Eavesdropping on unencrypted traffic (passwords, cookies, personal data) → called sniffing attacks.
- Types: Passive (just listening) vs. Active (injecting/modifying packets).
- Protection: Use encryption (HTTPS/TLS), VPNs, switched networks (vs. hubs).

Task Procedure and Outcomes:

Filter Packets by Protocol (HTTP, DNS, TCP):

- For **HTTP** (web traffic, plain-text): typed http → showed GET/POST requests, Host headers, readable data.
- For **DNS** (domain name queries): typed dns or udp.port == 53 → showed queries like "www.google.com" and responses with IP addresses.
- For **TCP** (reliable connections): typed tcp → showed all TCP packets, including handshakes and data transfers.

These filters help focus on one protocol at a time instead of seeing thousands of mixed packets.

Observe Three-Way TCP Handshake:

The TCP three-way handshake is how two devices agree to start a reliable connection.

Use filter with tcp then looked for the start of a connection (e.g., when loading a website). This is the classic sequence of 3 packets:

- SYN – Client sends to server: "I want to connect" (Flags: SYN=1).
- SYN-ACK – Server replies: "OK, I want to connect too" (Flags: SYN=1, ACK=1).
- ACK – Client confirms: "Great, connection ready" (Flags: ACK=1).

Identify Plain-Text Traffic vs Encrypted Traffic:

Compare two types of traffic:

- **Plain-text (HTTP):** Filtered with http. Visit <http://neverssl.com>. In packet details, I could read everything clearly: GET request, Host header, even parts of the page content. Anyone sniffing could see usernames, passwords, or data.
- **Encrypted (HTTPS):** Filtered with tls or tcp.port == 443. Visit <https://www.google.com>. I saw the TLS handshake (Client Hello, Server Hello), where the "Application Data" was encrypted – just random bytes, no readable text. Only the domain name (via SNI) was visible.

This shows why HTTPS is much safer: sensitive info stays hidden even if traffic is captured.

Capture DNS Queries and Analyse Them:

DNS turns domain names (like google.com) into IP addresses.

Filter with dns or udp.port == 53 (DNS usually uses UDP port 53). While browsing, Wireshark captured many DNS queries. Example:

- Query: "Standard query 0x0001 A www.google.com" (my computer asking for the IP).
- Response: "Standard query response ... A 142.250.XX.XX" (server replying with the IP).

This helps understand how websites load quickly (DNS resolves names first).

