1 Introduction to Computer Networks

A **Computer Network** is a collection of interconnected devices (computers, servers, routers, switches, etc.) that communicate to share resources, data, and applications. Networks enable everything from internet browsing to cloud computing, IoT, and global communication systems. They are foundational to modern computing, underpinning applications in business, entertainment, science, and more.

1.1 Key Concepts

- **Purpose**: Resource sharing (files, printers), communication (email, VoIP), scalability (cloud), and fault tolerance.
- **Components**: Nodes (devices), links (wired/wireless), protocols (rules for communication).
- **Types**: LAN (Local Area Network), WAN (Wide Area Network), MAN, PAN, etc.
- **Importance**: Drives the internet, distributed systems, and real-time applications.
- **As of 2025**: Trends include 6G research, AI-driven network management (e.g., intent-based networking), quantum networking (e.g., quantum key distribution), and sustainable, energy-efficient designs for edge computing and IoT.

1.2 History

- **1960s**: ARPANET (1969), precursor to the internet.
- **1970s**: TCP/IP developed (Cerf, Kahn). Ethernet (Metcalfe, 1973).
- **1980s–1990s**: Internet commercialization, WWW (Berners-Lee, 1989). LAN technologies (Wi-Fi, 1997).
- **2000s**: Broadband, mobile networks (3G, 4G). IPv6 adoption begins.
- **2010s**: SDN (Software-Defined Networking), NFV (Network Function Virtualization). 5G rollout (2019).
- **2020s**: 5G matures, 6G research accelerates. AI optimizes traffic, quantum networks emerge.

2 Network Types and Topologies

2.1 Types

- **LAN**: Small area (office, home). High speed, low latency (e.g., Ethernet, Wi-Fi).
- **WAN**: Large area (internet, ISP networks). Lower speed, higher latency (e.g., MPLS).
- **MAN**: City-scale (e.g., cable TV networks).
- **PAN**: Personal devices (e.g., Bluetooth).
- **VPN**: Secure overlay over public networks (e.g., OpenVPN).
- **SAN**: Storage Area Network for high-speed data access.

2.2 Topologies

- **Bus**: Single shared medium. Simple, but collision-prone.
- **Star**: Central hub (e.g., switch). Scalable, single point of failure.
- **Ring**: Circular connection. Reliable but slow for large networks.
- **Mesh**: Fully/partially connected. Robust, expensive.
- **Tree**: Hierarchical (e.g., enterprise networks). Balances scalability, cost.
- **Hybrid**: Combines topologies for flexibility.

3 OSI and TCP/IP Models

Frameworks to standardize network functions.

3.1 OSI Model (7 Layers)

Layer	Name	Function	Protocols/Tech
7	Application	User interface, services	HTTP, FTP, SMTP
6	Presentation	Data formatting, encryption	SSL/TLS, JPEG
5	Session	Session management	RPC, NetBIOS
4	Transport	End-to-end delivery, reliability	TCP, UDP
3	Network	Routing, logical addressing	IP, ICMP
2	Data Link	Node-to-node transfer, error detection	Ethernet, Wi-Fi, PPP
1	Physical	Hardware, signals	Cables, fiber, 802.11

Table 1: OSI Model Layers

3.2 TCP/IP Model (4 Layers)

- **Application**: OSI 5–7 (e.g., HTTP, DNS).
- **Transport**: TCP/UDP for reliability or speed.
- **Internet**: IP for routing (IPv4, IPv6).
- **Link**: OSI 1–2 (e.g., Ethernet).

3.3 Key Differences

- OSI: Theoretical, detailed. TCP/IP: Practical, internet-focused.
- Encapsulation: Data wrapped with headers at each layer (e.g., packet = IP header + TCP header + payload).

4 Physical Layer

Handles raw bit transmission.

4.1 Media

- **Wired**: Twisted pair (Cat5e, Cat6), coaxial, fiber optic (high bandwidth, low loss).
- **Wireless**: Radio (Wi-Fi, 5G), infrared, microwave.
- **Standards**: Ethernet (IEEE 802.3), Wi-Fi (802.11ax, Wi-Fi 6 in 2025).

4.2 Encoding

- **NRZ, Manchester**: Convert bits to signals.
- **Modulation**: AM, FM, QAM for wireless.

4.3 Devices

Hubs, repeaters, transceivers.

5 Data Link Layer

Ensures reliable node-to-node transfer.

5.1 Protocols

- **Ethernet**: Dominant LAN (CSMA/CD for collision detection, legacy).
- **Wi-Fi (802.11)**: CSMA/CA for wireless.
- **PPP**: Point-to-Point Protocol for direct links.

5.2 Functions

- **Framing**: Encapsulate data into frames.
- **MAC Addressing**: Unique 48-bit addresses (e.g., 00:1A:2B:3C:4D:5E).
- **Error Detection**: CRC (Cyclic Redundancy Check), parity.
- **Flow Control**: Sliding window protocols.

5.3 Devices

Switches (Layer 2, MAC-based forwarding), bridges.

5.4 Sub-Layers

- **LLC (Logical Link Control)**: Flow control, error handling.
- **MAC (Media Access Control)**: Hardware addressing.

6 Network Layer

Handles routing and logical addressing.

6.1 Protocols

- **IP (Internet Protocol)**:
  - IPv4: 32-bit addresses (4.3B unique).
  - IPv6: 128-bit addresses (340 undecillion, adopted widely by 2025).
- **ICMP**: Ping, error messages.
- **Routing Protocols**:
  - RIP (distance-vector), OSPF (link-state), BGP (internet backbone).

6.2 Functions

- **Routing**: Path selection (e.g., shortest path via Dijkstra's algorithm).
- **Addressing**: IP addresses, subnetting (e.g., 192.168.1.0/24).
- **Fragmentation**: Break packets for MTU (Maximum Transmission Unit).

6.3 Devices

Routers (forward based on IP), gateways (protocol translation).

7 Transport Layer

Provides end-to-end communication.

7.1 Protocols

- **TCP**: Connection-oriented, reliable (3-way handshake, retransmission, flow control). For HTTP, FTP.
- **UDP**: Connectionless, fast, no reliability. For DNS, streaming.
- **SCTP**: Combines TCP/UDP features, multi-streaming.

7.2 Functions

- **Segmentation**: Break data into segments.
- **Port Addressing**: Identify applications (e.g., 80 for HTTP).
- **Congestion Control**: TCP uses AIMD (Additive Increase, Multiplicative Decrease).
- **Multiplexing**: Multiple apps share network.

8 Application Layer

Interfaces with user applications.

8.1 Protocols

- **HTTP/HTTPS**: Web (REST APIs, HTTPS dominant with TLS 1.3 in 2025).
- **DNS**: Resolves names to IPs (e.g., x.com $\rightarrow$ 104.244.42.1).
- **SMTP/IMAP/POP3**: Email.
- **FTP/SFTP**: File transfer.
- **DHCP**: Assigns IPs dynamically.

8.2 Services

Web servers (Apache, Nginx), email servers, VoIP (SIP, WebRTC).

9 Network Security

Critical for protecting data and systems.

9.1 Concepts

- **CIA Triad**: Confidentiality, Integrity, Availability.
- **Threats**: Packet sniffing, DDoS, man-in-the-middle.
- **Encryption**:
  - Symmetric: AES (fast, shared key).
  - Asymmetric: RSA, ECC (public/private keys).
  - TLS/SSL: Secure communication (HTTPS, VPNs).
- **Authentication**: Passwords, OAuth, biometrics.
- **Firewalls**: Filter traffic (stateful, stateless).
- **IDS/IPS**: Intrusion Detection/Prevention Systems.

9.2 Protocols

- **IPSec**: Secure IP communication.
- **VPN**: Tunneling (e.g., WireGuard, OpenVPN).
- **Quantum Cryptography**: QKD (Quantum Key Distribution) emerging in 2025.

10 Advanced Topics

- **Software-Defined Networking (SDN)**: Centralized control plane (e.g., OpenFlow). Flexible, programmable networks.
- **Network Function Virtualization (NFV)**: Virtualized firewalls, routers.
- **5G and 6G**:
  - 5G: Low latency (1ms), high bandwidth (10 Gbps).
  - 6G (research in 2025): Tbps speeds, AI-native, holographic communication.

- **Quantum Networking**: Entanglement-based communication, quantum repeaters.
- **AI-Driven Networks**: ML for traffic prediction, anomaly detection.
- **Edge Computing**: Process data near source (IoT, autonomous vehicles).
- **Intent-Based Networking**: Declare goals, AI implements (e.g., Cisco DNA).

11 Performance and Optimization

- **Metrics**: Latency, throughput, jitter, packet loss.
- **Techniques**:
  - Caching (CDNs like Akamai).
  - Load Balancing: Distribute traffic (e.g., HAProxy).
  - QoS (Quality of Service): Prioritize traffic (e.g., VoIP over HTTP).
- **Tools**: Wireshark (packet analysis), iperf (bandwidth testing).

12 Practical Implementation

Example (Python socket programming, TCP client):

```python
import socket
client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client.connect(('127.0.0.1', 12345))  # Server IP, port
client.send(b"Hello,␣Server!")
data = client.recv(1024)
print(data.decode())  # Receive response
client.close()
```

13 Best Practices

- **Security**: Encrypt data (TLS), use strong authentication.
- **Scalability**: Design for growth (e.g., subnetting, SDN).
- **Monitoring**: Use tools like Nagios, Prometheus.
- **Redundancy**: Avoid single points of failure (e.g., BGP for routing).