**Networking Principles**

Networking principles revolve around core concepts such as **packet switching, circuit switching, error control, congestion control, quality of service (QoS), and security.** Advanced networking considers **scalability, fault tolerance, and efficiency** in handling massive traffic loads in distributed environments.

**Switching: Circuit vs. Packet Switching**

1. **Circuit Switching**: A dedicated communication path is established before data transfer. Used in traditional telephony but inefficient for bursty traffic.
2. **Packet Switching**: Data is divided into packets, each routed independently. It supports **store-and-forward mechanisms, statistical multiplexing, and adaptive routing** for dynamic networks.

Advanced networks use **MPLS (Multi-Protocol Label Switching)**, which combines circuit and packet switching to improve deterministic behavior in routing.

**Scheduling - Performance Bounds & Best-Effort Disciplines**

**Packet scheduling** determines how packets are selected for transmission from a queue. Advanced algorithms include:

* **WFQ (Weighted Fair Queuing):** Assigns bandwidth proportionally based on predefined weights.
* **EDF (Earliest Deadline First):** Used in real-time systems to ensure deadline adherence.
* **Deficit Round Robin (DRR):** Improves fairness over simple round-robin.

For **performance bounds**, queuing delay analysis using **Little’s Law (L = λW)** and **Network Calculus** helps in defining deterministic QoS.

**Best-Effort Discipline:**  
This represents a **no-QoS guarantee** model where packets are sent without prioritization (e.g., traditional IP routing). Techniques like **RED (Random Early Detection)** help manage congestion in best-effort networks.

**Naming and Addressing**

* **Hierarchical vs. Flat Addressing:** IP uses **hierarchical addressing** (e.g., CIDR), while MAC addresses are **flat** (unique identifiers).
* **DNS (Domain Name System):** Uses **recursive and iterative queries** with caching to resolve hostnames efficiently.
* **Name Resolution in Distributed Systems:** DHT-based approaches (like **Chord** in P2P networks) provide decentralized alternatives to traditional DNS.

**Protocol Stack (Layered Model vs. Cross-Layer Design)**

The **OSI model (7 layers)** and **TCP/IP model (4 layers)** guide protocol design.

* **Cross-layer optimization** challenges the strict layer separation to improve performance, such as dynamic power control in wireless networks.
* Protocol verification tools like **TLA+ and Spin** help validate protocol correctness in complex distributed systems.

**Internet Addressing**

* **IPv4 vs. IPv6:** IPv6 offers **128-bit addressing, hierarchical routing, and built-in security (IPsec)**.
* **Addressing Mechanisms:**
  + **Unicast, Multicast, Broadcast, Anycast** (for efficient content delivery).
  + **NAT (Network Address Translation):** Used for IPv4 address conservation.

**Routing**

Routing involves finding **optimal paths based on metrics like hop count, latency, and link stability.**

**Types of Routing:**

1. **Static Routing:** Manually configured routes (not scalable).
2. **Dynamic Routing:** Uses algorithms to determine paths dynamically (e.g., RIP, OSPF, BGP).

**Advanced Routing Protocols:**

* **OSPF (Open Shortest Path First):** Uses **Dijkstra’s Algorithm** to compute shortest paths.
* **BGP (Border Gateway Protocol):** The backbone of inter-domain routing, using **path vector mechanisms** to prevent routing loops.
* **SDN (Software-Defined Networking):** Decouples control and data planes, allowing centralized control via **OpenFlow**.

**Routing in High-Speed Networks**

* **Segment Routing (SR):** Enhances MPLS by encoding paths directly in packet headers.
* **QUIC (Quick UDP Internet Connections):** A transport-layer protocol designed to improve congestion control over unreliable networks.

**Point-to-Point Protocols and Links: Physical Layer, Error Detection, Comparative Performance**

**analysis of ARQ, Framing: Character Based, Bit Oriented, Length Field, Framing with Errors,**

**Maximum Frame Size.**

**Point-to-Point Protocols and Links**

Point-to-Point (P2P) protocols are essential for **direct communication between two network nodes**. These protocols are widely used in **serial communication, DSL connections, and VPN tunnels**.

**Key Aspects of P2P Communication:**

* **Data encapsulation** for transmission over physical links.
* **Error detection and correction mechanisms** to ensure data integrity.
* **Framing techniques** to structure transmitted data effectively.

**Physical Layer in P2P Links**

At the **physical layer (Layer 1)**, P2P links can be **wired (e.g., serial links, fiber) or wireless (Wi-Fi, satellite)**. Transmission characteristics include:

* **Synchronous vs. Asynchronous Transmission**: Synchronous transmission (e.g., SONET) is more efficient but requires tight timing.
* **Bit Rate and Modulation Techniques**: Uses **NRZ (Non-Return-to-Zero)**, **Manchester encoding**, or **8b/10b encoding** in high-speed links.

**Error Detection in P2P Links**

Error detection ensures **data integrity** by identifying corrupted packets. Techniques include:

1. **Parity Bit** – Adds a simple **even/odd parity check** but is weak.
2. **Checksum** – A **modular arithmetic sum** of packet data (used in TCP, UDP).
3. **Cyclic Redundancy Check (CRC)** – More robust, uses **polynomial division** (used in Ethernet, PPP).
4. **Hamming Code** – Supports **error detection and correction** by adding redundant bits.

**Comparative Performance Analysis of ARQ (Automatic Repeat reQuest)**

ARQ is a **retransmission-based error control** mechanism used in data link and transport layers.

| **ARQ Type** | **Mechanism** | **Pros** | **Cons** |
| --- | --- | --- | --- |
| **Stop-and-Wait ARQ** | Sender waits for an ACK after sending each frame | Simple, easy to implement | Inefficient for high-latency links |
| **Go-Back-N ARQ** | Allows multiple frames to be sent before an ACK but retransmits all frames after an error | Higher efficiency than Stop-and-Wait | Retransmits unnecessary frames, wasting bandwidth |
| **Selective Repeat ARQ** | Retransmits only erroneous frames | More bandwidth-efficient | Requires complex buffering |

* **Go-Back-N vs. Selective Repeat**: **Selective Repeat ARQ is preferred in high-speed networks** where bandwidth is valuable.
* **Performance in High-Latency Networks**: **Go-Back-N and Selective Repeat outperform Stop-and-Wait** in long-distance links like satellite communications.

**Framing in P2P Communication**

Framing defines **how data is structured within packets** before transmission. Different techniques exist to **detect frame boundaries and handle errors efficiently**.

**Types of Framing Techniques:**

| **Framing Type** | **Characteristics** | **Issues** | **Examples** |
| --- | --- | --- | --- |
| **Character-Based Framing** | Uses special characters (e.g., STX, ETX) to mark frame boundaries | **Byte stuffing** needed to prevent conflicts | Used in early serial protocols (e.g., BISYNC) |
| **Bit-Oriented Framing** | Uses bit patterns (e.g., 01111110 in HDLC) instead of characters | **Bit stuffing** required to avoid accidental frame markers | Used in **HDLC, SDLC, PPP** |
| **Length Field Framing** | Specifies frame size in a header field | Susceptible to **header corruption** | Used in **Ethernet, IP packets** |
| **Framing with Error Control** | Uses additional **checksums, CRC** to detect framing errors | Overhead increases with error control complexity | Seen in **PPP, TCP/IP** |

**Framing Errors and Solutions:**

* **Bit/Byte Stuffing**: Prevents accidental marker sequences.
* **Synchronization Issues**: Handled using **preamble sequences** or **frame synchronization fields**.
* **Error Handling**: Frames are **discarded, retransmitted (ARQ), or corrected (FEC - Forward Error Correction)**.

**Maximum Frame Size Considerations**

* **MTU (Maximum Transmission Unit):** Defines the largest frame that can be sent without fragmentation.
* **PPP MTU:** 1500 bytes (standard) but can be negotiated.
* **Jumbo Frames:** Used in **high-speed networks (9000 bytes or more) to reduce overhead**.
* **Trade-off:** Large frames improve **efficiency**, but increase **latency and risk of retransmissions** due to errors.

**Summary:**

* P2P links operate at the **physical and data link layers**, handling **error detection, framing, and retransmissions**.
* **ARQ techniques** balance **efficiency and reliability** in different network conditions.
* **Framing methods** vary from **character-based (older) to bit-oriented (modern protocols like PPP, HDLC)**.
* **Maximum frame size** affects performance—**larger frames improve efficiency but risk higher retransmission costs.**

**Delay Models in Data Networks: Introduction, Markov Chains, Queuing Models: Little’s**

**Theorem, Single Server M/M/1, m-Server M/M/m, Infinite Server M/M/∞, m-Server Loss**

**M/M/m/m Queuing System. M/G/1 Queues with vacation.**

**Delay Models in Data Networks**

In **data networks**, delay analysis is critical for understanding **performance, congestion control, and Quality of Service (QoS).** Delays arise due to **propagation, transmission, queuing, and processing**. Advanced analysis uses **Markov Chains, Queuing Models, and Little’s Theorem** to model network performance.

**Markov Chains in Network Delay Analysis**

Markov Chains are used to **model packet arrival and service processes** in queuing systems. They assume:

* **Memoryless Property**: The system's future state depends only on the present state.
* **Poisson Arrivals (λ)**: Common in network traffic modeling.
* **Exponential Service Times (μ)**: Used in many queuing models.

A **continuous-time Markov chain (CTMC)** is used for modeling queuing systems where:

* **States represent the number of packets in the system** (e.g., 0, 1, 2, …, n).
* **Transitions depend on packet arrival rates (λ) and service rates (μ).**

**Queuing Models in Networks**

Queuing models analyze **packet delays, system congestion, and resource utilization**.

**Little’s Theorem**

A fundamental result in queueing theory states:

L=λWL = \lambda WL=λW

where:

* LLL = Average number of packets in the system.
* λ\lambdaλ = Arrival rate of packets.
* WWW = Average time a packet spends in the system.

**Implication**: This theorem applies to **all queuing models, regardless of service time distribution**.

**Single Server Queue (M/M/1)**

The **M/M/1 model** represents a **single-server system** where:

* **Arrivals follow a Poisson process** (Markovian, rate λ).
* **Service times are exponentially distributed** (rate μ).
* **There is a single server** (1 service point).
* **FIFO discipline** (First In, First Out).

**Performance Metrics**

* Probability of **n** packets in the system: Pn=(1−ρ)ρn,where ρ=λμP\_n = (1 - \rho) \rho^n, \quad \text{where } \rho = \frac{\lambda}{\mu}Pn​=(1−ρ)ρn,where ρ=μλ​
* Average queue length: L=ρ1−ρL = \frac{\rho}{1 - \rho}L=1−ρρ​
* Average waiting time: W=1μ−λW = \frac{1}{\mu - \lambda}W=μ−λ1​

**Key insight**: As **ρ → 1 (λ ≈ μ), the system becomes congested**, causing delay to grow exponentially.

**Multi-Server Queues (M/M/m)**

The **M/M/m model** extends M/M/1 to **multiple servers (m)** handling packet arrivals in parallel.

**Performance Metrics**

* **Traffic intensity per server**:

ρ=λmμ\rho = \frac{\lambda}{m\mu}ρ=mμλ​

* **Probability of an empty system** (Erlang-B formula for loss systems):

P0=(∑k=0m(λ/μ)kk!)−1P\_0 = \left( \sum\_{k=0}^{m} \frac{(\lambda / \mu)^k}{k!} \right)^{-1}P0​=(k=0∑m​k!(λ/μ)k​)−1

* **Average number of packets in the system**:

L=λμ+PQ1−PQ×λmμL = \frac{\lambda}{\mu} + \frac{P\_Q}{1 - P\_Q} \times \frac{\lambda}{m\mu}L=μλ​+1−PQ​PQ​​×mμλ​

where PQP\_QPQ​ is the probability that an arrival has to wait.

**Use case**: **Data centers, cloud computing** where multiple servers handle network traffic.

**Infinite Server Model (M/M/∞)**

In an **M/M/∞** queue:

* **Each arrival gets an instant server (no waiting)**.
* **No congestion since the number of servers is unlimited**.
* **Poisson departure rate = Poisson arrival rate** (λ).
* **Number of active servers follows a Poisson distribution**:

Pn=e−ρρnn!,ρ=λμP\_n = \frac{e^{-\rho} \rho^n}{n!}, \quad \rho = \frac{\lambda}{\mu}Pn​=n!e−ρρn​,ρ=μλ​

**Use case**: **Modeling large-scale cloud networks** where servers auto-scale to match demand.

**Loss Model (M/M/m/m) – Erlang-B Queues**

A **pure loss model** (M/M/m/m) has **m servers and no queue**:

* If all servers are busy, new arrivals are **dropped**.
* This models **circuit-switched networks (e.g., telephony, VoIP)**.
* **Blocking Probability (Erlang-B Formula)**:

Ploss=(λ/μ)mm!∑k=0m(λ/μ)kk!P\_{loss} = \frac{\frac{(\lambda/\mu)^m}{m!}}{\sum\_{k=0}^{m} \frac{(\lambda/\mu)^k}{k!}}Ploss​=∑k=0m​k!(λ/μ)k​m!(λ/μ)m​​

**Use case**: **Cellular networks, VoIP**, where exceeding capacity leads to call drops.

**M/G/1 Queues with Vacation**

**M/G/1 Model:**

* Poisson arrivals (**M**), **general (arbitrary) service time distribution (G)**, and **one server (1)**.
* More general than **M/M/1** since service times **don’t have to be exponential**.
* **Waiting time formula (Pollaczek-Khinchine equation)**:

W=λE[S2]2(1−ρ)W = \frac{\lambda E[S^2]}{2(1 - \rho)}W=2(1−ρ)λE[S2]​

where **E[S2]E[S^2]E[S2]** is the second moment of the service time.

**Vacation Queues (M/G/1 with Vacation)**

* The server **takes breaks (vacations) when idle** to save resources.
* Applications: **Energy-saving mechanisms in wireless networks**, where access points go into low-power mode.
* **Impact on delay**:
  + Increases waiting time if the server is on vacation when a packet arrives.
  + Reduces energy consumption in power-constrained systems.

**Summary**

* **Markov Chains** model **Poisson arrivals and exponential service** processes.
* **Little’s Theorem** applies to **all queues** and links **queue length with delay**.
* **M/M/1** queues model basic **single-server systems** with exponential delays.
* **M/M/m** handles **multiple servers** and is used in **data centers, cloud computing**.
* **M/M/∞** models **infinitely scalable systems** like cloud networks.
* **M/M/m/m (Erlang-B)** models **call-blocking systems** like VoIP.
* **M/G/1 queues with vacation** model **power-saving mechanisms** in wireless networks.

**Multi-access Communication: Introduction, Slotted Multi-access and the ALOHA system,**

**Splitting Algorithms, Carrier Sensing, Multi-access Reservation.**

**Multi-Access Communication**

**Introduction**

Multi-access communication is a fundamental concept in **shared medium networks**, where multiple users attempt to send data over a common channel. Efficient **medium access control (MAC) protocols** are required to minimize collisions, improve throughput, and manage fairness.

* **Applications**: Wireless LANs (Wi-Fi), satellite networks, cellular networks, and Ethernet.
* **Challenges**: Collision avoidance, delay minimization, and efficient bandwidth utilization.

**Slotted Multi-Access and the ALOHA System**

**ALOHA Protocol (Developed at the University of Hawaii)**

**ALOHA is one of the earliest multi-access protocols**, used in wireless and satellite networks. It operates under the **pure ALOHA** and **slotted ALOHA** variants.

**Pure ALOHA**

* A station transmits **whenever it has data**, without checking the medium.
* If a collision occurs, the station waits a **random backoff time** before retransmitting.

**Throughput Analysis of Pure ALOHA**

* Vulnerable period = **2 frame times (T)**
* Probability of success: Psuccess=Ge−2GP\_{success} = G e^{-2G}Psuccess​=Ge−2G
* **Maximum throughput** occurs when G=0.5G = 0.5G=0.5, giving **18.4%** efficiency.

**Slotted ALOHA**

* **Time is divided into slots**, and transmission only starts at the beginning of a slot.
* **Collisions only occur if two packets start in the same slot**, improving efficiency.

**Throughput Analysis of Slotted ALOHA**

* Vulnerable period = **1 frame time (T)**
* Probability of success: Psuccess=Ge−GP\_{success} = G e^{-G}Psuccess​=Ge−G
* **Maximum throughput** occurs at G=1G = 1G=1, giving **36.8%** efficiency.

**Comparison**: Slotted ALOHA **doubles the efficiency** of Pure ALOHA by reducing the vulnerable period.

**Splitting Algorithms for Multi-Access**

Splitting algorithms help resolve **collisions** by dividing users into smaller groups iteratively.

**Tree Splitting Algorithm**

* **Used in ALOHA-based networks** to handle multiple collisions.
* If a collision occurs, the group is **split into two subsets**, and retransmissions happen separately.
* Recursively applies until all packets are successfully transmitted.
* **Used in RFID and wireless sensor networks**.

**Binary Exponential Backoff (BEB)**

* **Used in Ethernet (CSMA/CD) and Wi-Fi (CSMA/CA)**.
* After a collision, a node waits for a **random time within a backoff window [0,2k−1][0, 2^k-1][0,2k−1]** slots.
* **The window size doubles after each collision**, reducing repeated collisions.

**Carrier Sensing in Multi-Access Networks**

Carrier sensing techniques help **avoid collisions before transmission**. Used in **Carrier Sense Multiple Access (CSMA) protocols**.

**Types of CSMA:**

1. **CSMA (1-Persistent)**
   * A station senses the channel.
   * If **idle**, it **transmits immediately**.
   * If **busy**, it **waits until idle**, increasing **collision probability**.
   * **Used in early Ethernet**.
2. **CSMA (Non-Persistent)**
   * If the channel is **busy**, the station waits a **random time** before rechecking.
   * Reduces collision probability but increases delay.
3. **CSMA (P-Persistent)**
   * Used in **slotted channels (e.g., Wi-Fi)**.
   * If the channel is **idle**, transmission occurs with **probability p** or waits for the next slot.
   * Balances efficiency and fairness.
4. **CSMA/CD (Collision Detection)**
   * **Used in wired Ethernet**.
   * Listens while transmitting. If a collision is detected, **immediately stops and backs off**.
5. **CSMA/CA (Collision Avoidance)**
   * **Used in Wi-Fi (802.11)** since **collision detection is not possible in wireless networks**.
   * Uses **RTS (Request-to-Send) / CTS (Clear-to-Send)** handshake to avoid collisions.
   * Uses **exponential backoff** to avoid repeated collisions.

**Multi-Access Reservation Techniques**

Reservation techniques **schedule transmission slots** to avoid collisions.

**Time Division Multiple Access (TDMA)**

* **Time is divided into slots** assigned to different users.
* **Collision-free but requires synchronization**.
* **Used in GSM cellular networks**.

**Frequency Division Multiple Access (FDMA)**

* The frequency spectrum is **divided into separate bands** for different users.
* Used in **analog cellular systems** (1G networks).

**Code Division Multiple Access (CDMA)**

* Users **share the same frequency and time** but use **unique codes** for differentiation.
* Used in **3G and 4G mobile networks**.

**Polling and Token Passing**

* A **central controller (polling)** or **token-based system** (token passing) regulates access.
* Used in **Token Ring and Bluetooth networks**.

**Comparison of Multi-Access Protocols**

| **Protocol** | **Access Mechanism** | **Collision Handling** | **Efficiency** | **Used in** |
| --- | --- | --- | --- | --- |
| **Pure ALOHA** | Random transmission | Retransmit on collision | **18.4% max** | Satellite networks |
| **Slotted ALOHA** | Slot-based transmission | Retransmit on collision | **36.8% max** | Wireless sensor networks |
| **CSMA** | Senses before transmission | Avoids some collisions | Medium | Ethernet, Wi-Fi |
| **CSMA/CD** | Detects and aborts collision | Stops and retries | **High** | Wired Ethernet |
| **CSMA/CA** | Collision avoidance via RTS/CTS | Uses backoff | **High** | Wi-Fi (802.11) |
| **TDMA** | Time slots assigned | No collisions | **Very high** | GSM, 4G LTE |
| **FDMA** | Frequency bands assigned | No collisions | **Very high** | 1G cellular |
| **CDMA** | Unique code per user | No collisions | **Very high** | 3G, 4G LTE |

**Summary & Key Takeaways**

* **ALOHA-based protocols** allow random access but have **low efficiency** due to collisions.
* **CSMA improves performance** by sensing the channel before transmission.
* **CSMA/CD is used in Ethernet**, whereas **CSMA/CA is used in Wi-Fi**.
* **TDMA, FDMA, and CDMA eliminate collisions** but require complex coordination.
* **Polling and token-based methods** ensure fairness but introduce delay.

**Conclusion:** The choice of multi-access protocol depends on **network requirements (latency, efficiency, fairness, scalability).**

**Routing and Flow Control: WAN Routing, Shortest Path Routing, Optimization of routing data**

**structures. Flow Control Objectives, Window Flow Control, Rate Control Schemes, Rate**

**Adjustment Algorithms: Traffic and Congestion control.**

**Routing and Flow Control in Data Networks**

Routing and flow control are **critical mechanisms** for ensuring efficient and congestion-free data transmission in large-scale networks.

**1. WAN Routing**

Wide Area Networks (WANs) require **robust and scalable routing algorithms** due to:

* **Large-scale topology** (multiple interconnected routers).
* **Dynamic link conditions** (latency, bandwidth variations).
* **Fault tolerance and congestion control requirements**.

**Types of WAN Routing**

**1.1 Static vs. Dynamic Routing**

* **Static Routing**:
  + Predefined paths are manually set by network administrators.
  + Suitable for **small, stable networks** but **not scalable**.
* **Dynamic Routing**:
  + Uses **real-time network information** to update routes.
  + Adapts to **failures and congestion dynamically**.

**1.2 Distance Vector Routing (DVR)**

* Each router maintains a **routing table** with distances to other nodes.
* Uses the **Bellman-Ford algorithm**:

D(i,j)=min⁡k{D(i,k)+D(k,j)}D(i, j) = \min\_k \{ D(i, k) + D(k, j) \}D(i,j)=kmin​{D(i,k)+D(k,j)}

where:

* + D(i,j)D(i, j)D(i,j) = Distance from router iii to jjj.
  + D(i,k)D(i, k)D(i,k) = Known distance from iii to kkk.
  + D(k,j)D(k, j)D(k,j) = Distance from kkk to jjj.
* **Issues**:
  + **Slow convergence** (count-to-infinity problem).
  + **Not suitable for large-scale WANs**.
* **Examples**: Routing Information Protocol (RIP).

**1.3 Link-State Routing (LSR)**

* Each router **floods link-state advertisements (LSAs)** to the entire network.
* Uses **Dijkstra’s Shortest Path Algorithm**:

**Algorithm Steps:**

* 1. **Initialize** distance to all nodes as ∞, except source (0).
  2. Select the **minimum-distance node** and update distances of its neighbors.
  3. Repeat until all nodes are processed.
* **Advantages**:
  1. **Fast convergence**.
  2. **More accurate routing**.
* **Examples**: Open Shortest Path First (OSPF), Intermediate System to Intermediate System (IS-IS).

**1.4 Hierarchical Routing**

* WANs are divided into **smaller autonomous systems (AS)** for **scalability**.
* Uses **Interior Gateway Protocols (IGPs)** like **OSPF** inside AS and **Exterior Gateway Protocols (EGPs)** like **BGP** between ASes.

**1.5 Policy-Based Routing (PBR)**

* Used in **BGP (Border Gateway Protocol)**.
* Routes traffic based on **business and security policies** instead of shortest paths.

**2. Shortest Path Routing**

**Optimization Techniques for Routing**

Routing optimizations focus on reducing **latency, congestion, and computational complexity**.

1. **Dijkstra’s Algorithm** – Used in **OSPF** for **fast and reliable shortest path computation**.
2. **Floyd-Warshall Algorithm** – Computes all-pairs shortest paths, useful for **dense networks**.
3. **Bellman-Ford Algorithm** – Used in **Distance Vector Routing** (e.g., RIP).
4. *A Algorithm*\* – Enhances Dijkstra’s algorithm using a **heuristic function**, used in **AI-driven networking**.

**Optimizing Routing Data Structures**

* **Trie-based Routing Tables**: Used in hardware routers for **fast lookup** (e.g., Patricia Tries, Radix Tries).
* **FIB (Forwarding Information Base)**: Stores optimized **next-hop** decisions for fast packet forwarding.
* **TCAM (Ternary Content Addressable Memory)**: Hardware-accelerated lookup for IP routing.

**3. Flow Control in Networks**

Flow control regulates **packet transmission rates** to **prevent buffer overflow and congestion**.

**3.1 Objectives of Flow Control**

* **Maximize throughput** while avoiding **buffer overflow**.
* **Minimize packet loss** and retransmissions.
* **Achieve fairness** between competing flows.

**3.2 Window-Based Flow Control**

**Stop-and-Wait Protocol**

* Sender sends **one packet at a time** and waits for an **ACK** before sending the next.
* **Inefficient for high-latency networks** (e.g., WANs).

**Sliding Window Protocol**

* Sender **maintains a window** of packets that can be sent before receiving ACKs.
* **Types:**
  + **Go-Back-N (GBN)**: If a packet is lost, retransmit **all unacknowledged packets**.
  + **Selective Repeat (SR)**: Retransmit **only the lost packet** (more efficient).
* **Window Size WWW**:
  + **Optimal window size**: W=Bandwidth×RTTPacketSizeW = \frac{Bandwidth \times RTT}{PacketSize}W=PacketSizeBandwidth×RTT​
  + **Used in TCP (Transmission Control Protocol)**.

**3.3 Rate Control Schemes**

**Leaky Bucket Algorithm**

* Enforces a constant output **data rate** by controlling bursty traffic.
* **Used in traffic shaping (QoS enforcement)**.

**Token Bucket Algorithm**

* Allows **burst transmission** while maintaining an average rate.
* Used in **ATM networks, Wi-Fi, and cloud networks**.

**4. Rate Adjustment Algorithms**

**4.1 TCP Congestion Control**

TCP dynamically adjusts its **transmission rate** based on **network congestion** using:

**Slow Start**

* **Exponentially increases window size** until first loss/congestion occurs.
* **Formula**: cwnd=cwnd×2cwnd = cwnd \times 2cwnd=cwnd×2

**Congestion Avoidance**

* After slow start, **window grows linearly** to avoid congestion.
* **Formula**: cwnd=cwnd+1cwnd = cwnd + 1cwnd=cwnd+1

**Fast Retransmit & Fast Recovery**

* If **triple duplicate ACKs** are received, assume packet loss and **retransmit early** without waiting for timeout.

**4.2 Traffic and Congestion Control Techniques**

**Active Queue Management (AQM)**

* **Random Early Detection (RED)**: Proactively drops packets before the queue is full to signal congestion.
* **Explicit Congestion Notification (ECN)**: Marks packets instead of dropping them to signal congestion.

**Load Balancing**

* Spreads traffic **across multiple paths** (e.g., ECMP - Equal Cost Multipath).
* **Used in SDN (Software-Defined Networking) for dynamic traffic engineering**.

**Summary & Takeaways**

**Routing**

✅ **Dijkstra’s Algorithm (OSPF)**: Shortest Path Routing for WANs.  
✅ **BGP**: Policy-based WAN routing between ISPs.  
✅ **Trie-based Routing Tables**: Used for **fast IP lookup**.

**Flow Control**

✅ **Sliding Window (TCP)**: Efficient **flow control mechanism**.  
✅ **Rate Control (Leaky & Token Bucket)**: Used in **traffic shaping**.  
✅ **TCP Congestion Control (Slow Start, RED, ECN)**: Key for **internet stability**.

**TCP/IP Protocol Suite:**

**IP addresses, Delivery and forwarding of IP packets, Mobile IP.**

**Unicast routing protocols: UDP, TCP, DHCP, DNS, Remote login-Telnet, SSH; FTP, e-mail- SMTP,**

**POP; SNMP.**

**TCP/IP Protocol Suite: An Advanced Overview**

The **TCP/IP Protocol Suite** forms the backbone of the **Internet and enterprise networks**, handling everything from addressing and routing to application-layer services. It follows a **layered architecture** for efficient communication.

**1. IP Addresses & Packet Delivery**

**1.1 IP Addressing**

An **IP address** uniquely identifies a device on a network. It consists of:

* **IPv4 (32-bit address, e.g., 192.168.1.1)**
* **IPv6 (128-bit address, e.g., 2001:db8::ff00:42:8329)**

**1.1.1 IPv4 Addressing**

IPv4 addresses are classified into:

* **Class A**: Large networks (0.0.0.0 – 127.255.255.255)
* **Class B**: Medium networks (128.0.0.0 – 191.255.255.255)
* **Class C**: Small networks (192.0.0.0 – 223.255.255.255)
* **Class D**: Multicasting (224.0.0.0 – 239.255.255.255)
* **Class E**: Reserved (240.0.0.0 – 255.255.255.255)

However, **Class-based addressing** is outdated, and **CIDR (Classless Inter-Domain Routing)** is used for better efficiency.

**Subnetting**

Subnet masks divide a network into **subnetworks** for better management and security.  
Example:

* **IP Address**: 192.168.1.100
* **Subnet Mask**: 255.255.255.0
* **Network ID**: 192.168.1.0
* **Host Range**: 192.168.1.1 – 192.168.1.254

**Supernetting**

* Combines multiple **smaller networks** into a **larger network**.
* Used by **ISPs for efficient routing**.

**1.2 Packet Delivery and Forwarding**

**1.2.1 Direct vs. Indirect Delivery**

* **Direct Delivery**: When source and destination are in the **same subnet**.
* **Indirect Delivery**: When packets **cross routers** to reach another subnet.

**1.2.2 Forwarding Methods**

1. **Next-Hop Forwarding** – Uses only the **next-hop router** for packet forwarding.
2. **Longest Prefix Matching (LPM)** – Finds the **best-matching route** based on **CIDR**.
3. **Load Balancing Forwarding** – Uses multiple paths (ECMP).

**1.2.3 Routing Table Optimization**

* Uses **Trie-based lookup (Patricia Trie, Radix Trie)** for fast searches.
* **Forwarding Information Base (FIB)** helps routers make **quick routing decisions**.

**2. Mobile IP**

**Mobile IP (MIP)** enables devices to **maintain IP connectivity** while moving across different networks.

**2.1 Components of Mobile IP**

* **Home Agent (HA)**: Maintains the mobile node’s permanent IP.
* **Foreign Agent (FA)**: Provides a temporary care-of-address (CoA) when the node roams.
* **Mobile Node (MN)**: The device that moves between networks.

**2.2 Tunneling in Mobile IP**

When a **Mobile Node (MN)** moves to another network:

* **Packets sent to the MN’s home address** are **tunneled** (encapsulated) and forwarded via the **Foreign Agent (FA)**.
* Uses **GRE (Generic Routing Encapsulation)** or **IP-in-IP encapsulation**.

**3. Unicast Routing Protocols**

Unicast routing protocols determine the best path for **one-to-one communication**.

**3.1 UDP (User Datagram Protocol)**

* Connectionless, **best-effort** protocol.
* Used for **low-latency applications** like **VoIP, DNS, and video streaming**.
* No congestion control, making it faster than TCP.

**3.2 TCP (Transmission Control Protocol)**

* **Connection-oriented**, reliable delivery.
* Uses **flow control (Sliding Window Protocol)** and **congestion control (AIMD, TCP Reno, TCP CUBIC, etc.)**.

**3.3 DHCP (Dynamic Host Configuration Protocol)**

* Assigns **dynamic IP addresses** to clients in a network.
* Uses a **4-step process**:
  1. **Discover** (Client requests IP).
  2. **Offer** (Server responds with an available IP).
  3. **Request** (Client requests an offered IP).
  4. **Acknowledge** (Server assigns the IP).

**3.4 DNS (Domain Name System)**

* Resolves **domain names (e.g., google.com)** to **IP addresses**.
* Uses a hierarchical **distributed database** structure.
* **Types of DNS Queries**:
  + **Recursive**: Server finds the answer.
  + **Iterative**: Server refers the client to another DNS server.

**4. Remote Login Protocols**

**4.1 Telnet (Insecure Remote Login)**

* Provides **command-line access** to remote systems.
* **Transmits credentials in plaintext**, making it **insecure**.

**4.2 SSH (Secure Shell)**

* **Encrypted version** of Telnet using **asymmetric cryptography** (RSA, ECC).
* Uses **port 22** and supports **key-based authentication**.
* Provides secure file transfers via **SCP and SFTP**.

**5. File Transfer and Email Protocols**

**5.1 FTP (File Transfer Protocol)**

* Transfers files between client and server over **TCP (port 21)**.
* **Active Mode vs. Passive Mode**:
  + **Active Mode**: Server initiates data connection.
  + **Passive Mode**: Client requests a data port to avoid firewall issues.
* **FTPS (FTP Secure)** encrypts connections using **SSL/TLS**.

**5.2 Email Protocols**

* **SMTP (Simple Mail Transfer Protocol)**:
  + Sends emails from client to mail server (port 25 or 587).
* **POP3 (Post Office Protocol v3)**:
  + Downloads emails to local device and **deletes them from the server** (port 110).
* **IMAP (Internet Message Access Protocol)**:
  + Stores emails on the server for **multi-device access** (port 143 or 993 for IMAPS).

**6. SNMP (Simple Network Management Protocol)**

SNMP is used for **network monitoring and management**.

**6.1 SNMP Components**

* **SNMP Manager**: Collects network data.
* **SNMP Agent**: Runs on network devices (routers, switches) and **responds to queries**.
* **MIB (Management Information Base)**: Database storing information about network devices.

**6.2 SNMP Versions**

* **SNMPv1**: Basic network monitoring (insecure).
* **SNMPv2**: Introduced **bulk data retrieval**.
* **SNMPv3**: Secure encryption and authentication.

**Conclusion & Key Takeaways**

✅ **IP Addressing**: CIDR and Subnetting improve network efficiency.  
✅ **Routing**: TCP/IP relies on **dynamic routing algorithms (OSPF, BGP)**.  
✅ **TCP vs. UDP**: TCP ensures reliability, while UDP is best for real-time applications.  
✅ **Remote Login**: SSH is a **secure alternative to Telnet**.  
✅ **FTP & Email Protocols**: **FTP (data transfer), SMTP (email sending), IMAP (email access)**.  
✅ **SNMP**: Essential for **network monitoring**.