CS2105 Introduction to Computer Networks
AY25/26, Y3S1
Notes

Sim Ray En Ryan

December 3, 2025

Contents

1 Course Admin

1.1 Assessment

- Tutorial Attendance (5%)
- Assignments (20%)
- Midterms (25%)
- Finals (50%)

2 Introduction to Computer Networks

2.1 What is the Internet?

- **Hosts / End Systems** (1.01B in 2019)
- **Network Edge**: Hosts access the Internet via **access networks**.
- **Switch → Router → ISP**

2.1.1 Wireless Networks

- **Wireless LAN**: 802.11 b/g/n/ac (Wi-Fi)
- **Wide-area wireless access**: 3G, 4G, 5G (Cellular)

2.1.2 Physical Media

- **Guided**
  - Twisted Pair Cable
  - Fiber Optic Cable
- **Unguided**
  - Signals
  - Radio

2.2 Network Core

2.2.1 Packet Switching

- Breaks application message into **packets** of L bits.
- Transmits onto the link at rate R (link capacity, bandwidth).

- Delay = L/R
- **Store and forward**: Entire packets arrive at the router before going to the next link.
  - Used to check if the packet is correct or not.
- Resources used on demand, **congestion is possible**.

2.2.2 Circuit Switching

- **Call Setup Required**.
- **Guaranteed performance**.
- Idle if not used (Cannot be shared by multiple connections).
- Commonly used in old telephone networks.

2.2.3 Routing and Addressing

- **Routing Algorithms**
- **Addressing**: Packets carry source and destination information.

2.3 Delay, Loss, and Throughput in Networks

2.3.1 Packet Loss

- If the buffer is full because arrival rate > departure rate, then the packet will be dropped.

2.3.2 End-to-End Packet Delay

- **Processing Delay** (< 1 ms)
  - Router checks if it is the intended recipient.
  - Router decrypts/de-encapsulates the data.
  - Routers check for bit errors.
- **Queuing Delay**
  - Time waiting in queue for transmission if the link is already full.
  - Ingress Queue: Negligible for CS2105 (Network device is congested).
  - Egress Queue (Output link is congested).
- **Transmission Delay**

  - Delay from the max bandwidth a link can transmit.
  - Depends on Packet Length and Link Bandwidth.
- **Propagation Delay**
  - Delay from how long it takes for the data to transfer physically.
  - Depends on Length of link and Propagation speed in medium.

2.3.3 Throughput

- How many bits can be transmitted per unit of time (including delays).

2.4 Protocol Layers and Service Models

2.4.1 Protocol Definitions

A protocol defines:

- Format and order of messages exchanged.
- Actions taken after messages are sent or received.

2.4.2 Layer Model (from top to bottom)

- **Application**: FTP, SMTP, HTTP
- **Presentation**: Encryption, Compression
- **Session**: Synchronisation, checkpointing, recovery
- **Transport**: TCP, UDP
- **Network**: IP, routing protocols
- **Link**: Ethernet, 802.11, PPP
- **Physical**: Wires

3 Application Layer

3.1 Types of Network Applications

- Remote access (telnet, ssh)
- Email, FTP
- Web
- P2P File Sharing, Games, IM, Youtube, Facebook

3.1.1 Application Characteristics

Applications run on different hosts and communicate over the network. They can be:

- Client-server
- Peer-to-Peer

They require different network service characteristics:

- **Data Integrity**: FTP requires lossless; Streaming can tolerate data loss.
- **Timing**: Games need low latency.
- **Throughput**: Multimedia requires minimum amount of bandwidth to be effective.
- **Security**: Encryption, Authentication.

3.1.2 Architecture

- **Client-Server**
  - **Server**: Provides service to clients, usually in a data center.
  - **Client**: Initiates contact, requests service from server.
- **Peer-to-Peer (P2P)**
  - No always-on server.
  - End systems directly communicate.
  - Self scalability: New peers bring new capacity and demands.

3.1.3 Application-Layer Protocols

App-layer protocols define:

- Type of messages exchanged
- Message syntax
- Meaning of messages
- Rules

They can be:

- **Open**: Allows for interoperability (HTTP, SMTP).
- **Proprietary**: Skype.

3.2 Transport Layer Protocols

- **TCP (Transmission Control Protocol)**
  - Reliable data transfer
  - Flow control
  - Congestion Control
  - Security
- **UDP (User Datagram Protocol)**
  - Unreliable data transfer (Datagrams)
  - May be out of order
  - Faster as it can tolerate data loss
  - Time sensitive

3.3 Web and HTTP

- **Web** content consists of a Base HTML File and Referenced objects (Other HTMLs, Pictures, JS).
- **Access** is by URL: hostname/pathname.
- **HTTP** uses **TCP**.

3.3.1 HTTP Connection Types

- **Non-persistent**: One object sent, then connection closed.
- **Persistent**: Multiple objects can be sent over a single TCP connection.

3.3.2 HTTP Response Time

The response time for non-persistent HTTP is:

Response Time = (RTT to establish TCP)+(RTT for HTTP Request)+(File Transmission Time)

(Slow, so often opens parallel TCP connections)

3.3.3 HTTP Request Methods

- **HTTP 1.0**
  - `GET`
  - `POST`: Web page includes input; input uploaded to server in entity body.

  - `HEAD`: Leave requested object out of response.
- **HTTP 1.1**
  - `PUT`: Uploads file to path in URL.
  - `DELETE`: Deletes file in URL.

3.3.4 HTTP Response Message

Structure: Status Line/r/n → Header Lines/r/n → /r/n → Data

- **Status Line**: Protocol (`HTTP/1.1`) and Status Codes.

Common Status Codes

- **Informational (1xx)**
  - `100 Continue`: Initial part of a request has been received; client can continue.
- **Successful (2xx)**
  - `200 OK`: The request was successful.
  - `204 No Content`: Successful but no content to send back.
- **Redirection (3xx)**
  - `301 Moved Permanently`: The resource has been moved to a new URL.
  - `304 Not Modified`: The resource has not been modified since the last request.
- **Client Error (4xx)**
  - `400 Bad Request`: Server could not understand the request.
  - `404 Not Found`: The requested resource could not be found.
- **Server Error (5xx)**
  - `500 Internal Server Error`: A generic error occurred on the server.
  - `503 Service Unavailable`: Server is currently unable to handle the request due to temporary overloading or maintenance.

3.3.5 Cookies

- HTTP is stateless; cookies fix that.
- Cookie file kept on user's host.
- Server does not respond if cached copy is up to date.

3.4 Domain Name System (DNS)

- DNS translates a **Hostname** (`[www.example.com](https://www.example.com)`) to an **IP Address** (`93.184.216.34`).
- DNS uses **UDP**.

3.4.1 Resource Records (RR)

Format: (name, value, type, TTL)

- **A (Address)**: name: hostname, value: IP address.
- **NS (Name Server)**: name: domain, value: hostname of authoritative name server for this domain.
- **CNAME (Canonical Name)**: name: alias name, value: canonical name.
- **MX (Mail Exchange)**: value: name of mail server associated with name.

3.4.2 DNS Structure

- **Root DNS Servers** (13)
- **TLD (Top-Level Domain)**: com, org, edu, sg, uk, jp
- **Authoritative Servers**: `yahoo.com, facebook.com`, etc.
- **Local DNS Server**
  - Default name server.
  - Not part of the hierarchy.
  - Host makes DNS query to local DNS server.
  - Tries to resolve from cache, else acts as proxy and forwards to hierarchy.
  - Caches the name-mapping; expires after some time (TTL).

3.4.3 DNS Caching

- If query can be resolved by non-authoritative server / local dns server, it will use it ($\sim$ 1s vs $\sim$ 1ms).
- Cached entries expire after a TTL.
- If host name changes IP, it may not be known until TTL expire.
- Query can be **iterative** or **recursive** (rarely used).

4 Socket Programming

4.1 Addressing Processes

- A process is a program running within a host.
- Within the same host, processes communicate using inter-process communication (OS).
- On different hosts, they communicate by exchanging messages.
- Processes are addressed by **IP Address** (32 Bit integer) and **Port** (16 Bit integer).

4.2 Sockets

- Software interface between application layer and layer 4 (Transport Layer).
- Used by processes to send and receive messages.
- Can be **TCP** or **UDP**.
- **TCP (`SOCK_STREAM`)**
  - Server TCP creates a new socket for each process to communicate to client.
  - Has a welcome socket.
- **UDP (`SOCK_DGRAM`)**

5 UDP

5.1 Transport Services

- Provides logical communication between processes on different hosts.
- Protocol Data Unit: **Segments**.
- **Routers** do not run transport layer protocols; they only move segments. Reassembly/breakup is done by sender and receiver.

5.2 UDP Segment Services

UDP adds:

- **Connectionless multiplexing/demultiplexing**
  - **Demultiplexing**: Segments with the same destination port will be sent to the same socket.

  - **Multiplexing**: Gathers data from processes, forms packets, passes to IP layer.
- **Checksum**

5.3 UDP Characteristics

- **Unreliable**
- Does not establish connection (lower delay).
- Used by loss tolerant and rate-sensitive services.
- Application implements error detection and recover mechanisms if needed.

5.4 UDP Segment Format

All fields are 16 Bits:

- Source Port
- Destination Port
- Length (in bytes, including header)
- Checksum
- Data

5.4.1 Checksum Calculation

- Treat UDP Segment as a sequence of 16 Bit Integers.
- Checksum is 0 initially.
- Checksum is 1's complement of the sum of all 16-bit words (including the checksum field itself, which is initially 0).

5.5 Reliable Data Transfer (RDT) Protocols

5.5.1 Data Transfer Issues

Things that can happen during data transfer:

- Corrupted packets (Bit Flips)
- Dropped packets
- Reordered Packets
- Packets delivered after long delays

Protocols should ideally guarantee packets are: Delivered, Correct, and in the correct order.

5.5.2 RDT Interfaces

- rdt_send()/rdt_rcv(): Reliable interfaces.
- udt_send()/deliver_data(): Unreliable/delivery interfaces.

5.5.3 RDT 1.0 (Underlying channel is perfectly reliable)

- rdt_send(data) → udt_send(make_pkt(data))
- rdt_rcv(packet) → deliver_data(extract(packet))

5.5.4 RDT 2.0 (Channel has bit errors)

- Uses **ACK** (Receiver tells sender packets received OK) and **NAK** (Receiver tells sender packets received had errors, detected via checksum).
- **Sender retransmits packet** on NAK.
- **Stop and Wait Protocol**: Sender sends one packet at a time, waiting for ACK or NAK.

5.5.5 RDT 2.1 (Handling corrupted ACKs/NAKs)

- Sender adds a **sequence number** to each packet (0 or 1 is enough in Stop and Wait).
- Receiver discards duplicate packets when ACK is corrupted.

5.5.6 RDT 3.0 (Packets can be lost)

- Uses a **timeout**: If ACK for a packet is not received by the timeout, the sender retransmits the packet.
- If receiver receives a duplicate, it sends the ACK again.
- Has very bad utilization.

5.6 Pipelining Protocols

Pipelining allows sending **multiple packets at once** (without waiting for ACK for each one), leading to better utilization.

5.6.1 Go Back N (GBN)

- **Packet Loss**: If a packet is lost, the receiver sends a cumulative ACK for the last known/in-order packet.
- **Receiver**:
  - Sends **cumulative ACK** for the most recent correctly received in-order packet.
  - **Discards out of order packets** (no receiver buffering).
- **Sender**:
  - Has a timer for the **oldest unACKed packet**.
  - When the timer expires, retransmits **all** packets starting from the expired one.
  - Maintains a sender window of up to M unACKed packets.
- **Advantages of Cumulative ACK**: Simpler to implement, good performance if low error rate. If an ACK is lost, a later cumulative ACK may cover it.
- **Disadvantage**: A lot of retransmission and discarding of packets.

5.6.2 Selective Repeat (SR)

- **Receiver**:
  - Sends an **individual ACK** for each correctly received packet.
  - **Maintains a sliding window to buffer out of order packets**.
- **Sender**:
  - Maintains a **timer for each un-ACKed packet**.
  - Only retransmits the specific packet whose timer expired.
- **Performance**: Better than GBN when there's high error rates.
- **Complexity**: More complex, higher memory usage.

6 TCP (Transmission Control Protocol)

6.1 TCP Overview

- **Point to Point** connection.
- **Handshaking** required.
- **Full Duplex** communication.

- **Reliable in-order byte stream**.
- **Pipelined**: Uses a sender window, similar to Go-Back-N (GBN) with some Selective Repeat elements.

6.2 TCP Header

Header size: 20 Bytes (160 Bits) minimum. Key fields:

- **Source Port** (16 Bits)
- **Dest Port** (16 Bits)
- **Sequence Number** (32 Bits): "Byte Number" of the first byte of data in the segment. Initial SEQ is chosen randomly.
- **Ack Number** (32 Bits): Sequence Number of the **next byte expected** from the other side (cumulative ACK, like GBN).
- **Receive Window (rwnd)** (16 Bits): Number of bytes the receiver is willing to accept (**Flow Control**).
- **Flags** (1 Bit each):
  - `ACK`
  - `SYN` (Synchronization - connection setup)
  - `FIN` (Final - connection teardown)
  - `RST` (Reset)
  - `PSH` (Push Data)
  - `URG` (Urgent Data)
- **Checksum** (16 Bits)
- **Application Data**

6.3 TCP Reliability and Flow Control

6.3.1 TCP Flow Control

- The sender limits the amount of data sent based on the **Receive Window (rwnd)** advertised by the receiver. This prevents the sender from overwhelming the receiver's buffer.

6.3.2 Delayed ACK

- Receiver waits up to 500ms for the next segment before sending an ACK.

6.3.3 Out-of-Order Segments

- Receiver sends a **duplicate ACK** but buffers the out-of-order segment.
- Once the missing segment is received, it skips over the buffered data (cumulative).

6.4 Timeout and Retransmission

6.4.1 Estimated RTT

- Estimated RTT $= (1 - \alpha) \cdot$ Estimated RTT $+ \alpha \cdot$ SampleRTT
- Timeout = Estimated RTT $+ 4 \cdot$ DevRTT (Safety margin).
- Timeout must be longer than RTT.

6.4.2 Fast Retransmission (RFC2001)

- If the sender receives **4 ACKs for the same segment** (3 duplicate ACKs), it assumes the segment is lost and retransmits it immediately, even if the timeout has not been reached.

6.5 Connection Management (TCP Handshake)

6.5.1 Connection Establishment (3-way Handshake)

1. **Client → Server**: `SYN` (With SYN Flag, Seq#x)
2. **Server → Client**: `SYN-ACK` (With SYN/ACK Flags, Seq#y, ACK#$x + 1$)
3. **Client → Server**: `ACK` (With ACK Flag, ACK#$y + 1$)

6.5.2 Closing a Connection

1. **Client → Server**: `FIN`
2. **Server → Client**: `ACK` (Server can still send data)
3. **Server → Client**: `FIN`
4. **Client → Server**: `ACK`

7 The Network Layer

7.1 Network Layer Functions

- **Forwarding (Data Plane)**: Local, per-router function. Moves the packets.

- **Routing (Control Plane)**: Network-wide logic. Determines where the packets should go.
  - Traditional routing algorithm (in routers).
  - Software-defined networking (in remote servers).

7.2 IP Addressing

- Each Interface has its own **IP Address** (32 Bit Identifier).
- **Subnets**: Hosts are in the same subnet if they have the same network prefix, and can reach each other without a router.

7.2.1 Classless Inter-Domain Routing (CIDR)

- IP Address/x: x bits for the **network prefix**; $32 - x$ bits for the **host ID**.
- **Subnet Mask**: Set all prefix bits to 1 and host ID bits to 0. AND with IP Address to find the network address.
- **Longest Prefix Matching**: Forward to the address that matches the longest prefix.

7.2.2 Special IP Addresses

- 0.0.0.0/8 (Source address when no address is assigned)
- 127.0.0.0/8 (Loopback Address)
- Local Networks: 10.0.0.0/8, 172.16.0.0/12, 192.168.0.0/16
- 255.255.255.255/32 (Broadcast Address)

7.3 Dynamic Host Configuration Protocol (DHCP)

- Allows hosts to dynamically obtain an IP address (Plug and Play).
- DHCP Server Port: 67; DHCP Client Port: 68.

7.3.1 DHCP Timeline (Simplified)

1. Client sends **DHCP Discover** (Src: 0.0.0.0 : 68, Dst: 255.255.255.255 : 67)
2. DHCP Server returns **DHCP Offer** (Src: $x.x.x.x$: 67, Dst: 255.255.255.255 : 68, addr : $y.y.y.y$)
3. Client sends **DHCP Request** (requests the offered IP $y.y.y.y$)
4. DHCP Server sends **DHCP Ack** (confirms the IP $y.y.y.y$ is assigned)

7.4 IPv4 Datagram Format

Header: 20 Bytes minimum. Key fields:

- **IP Version** (4 Bit - should be 4)
- **IP Datagram Length** (16 Bits)
- **Identifier** (16 Bits), **Flags** (3 Bits), **Fragment Offset** (13 Bits) - Used for fragmentation and reassembly at the receiving host, based on link MTU (Max Transfer Units).
- **TTL (Time To Live)** (8 Bits - not 7): Number of remaining hops, decremented at each router.
- **Upper Layer Protocol** (8 Bits - not 9)
- **Header Checksum** (16 Bits)
- **Src IP Addr** (32 Bits)
- **Dst IP Addr** (32 Bits)
- **Data**

7.5 Network Address Translation (NAT)

- Maps multiple private IP addresses in a LAN to a single public IP address (WAN).
- Replaces source IP address and ports to NAT IP address and new ports.
- Remembers this mapping in the **NAT Translation table**.

7.5.1 Motivations

- Saves public IP addresses (One public IP needed per router).
- Hosts inside the local network are not explicitly addressable and visible by the outside world (security).

7.5.2 Challenges

- Breaks original network design (prevents direct communication).
- Increases switching delays.
- Limits number of concurrent connections through one public IP address.

7.6 Routing Algorithms

7.6.1 Intra-AS (Within an Autonomous System)

- Examples: RIP, OSPF.
- Single policy, focused on **Performance**.
- Each link (edge) has a cost (weight) related to bandwidth/congestion/hop count.
- **Link State Algorithms**: All routers have complete knowledge of network topology and link cost (e.g., Dijkstra, non-examinable).
- **Distance Vector Algorithms**: Routers only know link costs to neighbors, exchange local views, and update (e.g., Bellman-Ford).
- **Routing Information Protocol (RIP)**: Uses **Hop Count** as cost metric. Exchanges routing table every 30s using UDP Port 520.

7.6.2 Inter-AS (Between Autonomous Systems)

- Example: BGP.
- Admin wants control over how traffic is routed, so **Policy** may dominate over performance.

7.7 Internet Control Message Protocol (ICMP)

- Used to report errors and diagnostic information.
- **Ping**: Echo Request (Type 8, Code 0) and Echo Reply (Type 0, Code 0).
- **Traceroute**: Sends a small series of packets with incremental TTLs.

8 Link Layer

8.1 Protocol Data Unit (PDU)

- The PDU at the Link Layer is a **Frame** (which adds a Header and Trailer).

8.2 Link Layer Functions

- **Link Access Control**: Who can send what over the link.
- **Reliable Delivery**: Used on error-prone links (like Wireless links).
- **Error Detection/Correction**

8.3 Error Detection and Correction

Errors are caused by Signal Attenuation and Noise.

- **Parity Checking**
  - **Single Bit Parity**: Can detect single bit errors.
  - **Two-dimensional bit parity**: Can detect and correct single bit errors, and detect two bit errors.
- **Cyclic Redundancy Check (CRC)**
  - Sender: Has Data (D), appends R (CRC of r bits). Sends (D, R).
  - G: Generator polynomial of $r + 1$ bits, agreed upon by sender/receiver.
  - Receiver divides (D, R) by G. Non-zero remainder $\rightarrow$ Error.

8.4 Types of Links and Multiple Access Protocol

- **Point to Point Link**: No need for multiple access control (e.g., RJ45).
- **Broadcast Link**: Multiple nodes connected to a shared broadcast channel (e.g., Wifi, Satellite).
  - Simultaneous transmission $\rightarrow$ Frames collide $\rightarrow$ None would be read.
  - **Multiple Access Protocol** coordinates channel usage.

8.4.1 Multiple Access Protocol Categories

- **Channel Partitioning** (Divide channel into pieces, allocate exclusively)
  - **Time Division Multiple Access (TDMA)**: Each node gets a fixed-length time slot in each round. Unused slots go idle.
  - **Frequency Division Multiple Access (FDMA)**: Each node is assigned a fixed frequency band. Unused time in a band goes idle.
- **Taking Turns**
  - **Polling**: Master node invites slave nodes to transmit. Overhead, Single point of failure.
  - **Token Passing**: Control token is passed from one node to the next. Overhead, Single point of failure.
- **Random Access** (Focus on recovering from collisions)
  - **Slotted ALOHA**: Nodes only transmit at the beginning of a slot. If collision occurs, retransmit with probability P until success.

  - **CSMA/CD (Carrier Sense Multiple Access / Collision Detection)**: Sense channel before transmission. If collision detected, abort and retransmit after a random time. Used in Ethernet.
  - **CSMA/CA (Carrier Sense Multiple Access / Collision Avoidance)**: Used in WiFi; receiver returns ACK if a frame is received OK.

8.5 MAC Addresses and ARP

8.5.1 MAC Addresses

- Every Network Interface Card (NIC) has a **MAC Address** (48 Bits, 12 Hex Digits).
- First three bytes are the **OUI** (Organizational Universal Identifier).

8.5.2 Address Resolution Protocol (ARP)

- Maps **IP Address** to **MAC Address**.
- Each node has an **ARP Table** (IP Addr, MAC Addr, TTL).
- If destination MAC not known:
  1. **A sends ARP Query** (Dst MAC: `FF:FF:FF:FF:FF:FF`).
  2. **B sends ARP Reply** (Src MAC: `BB:BB...`, Dst MAC: `AA:AA...`).
  3. A updates its ARP table.
- If the destination is in another subnet, the Dst MAC changes to the **Router's (Default Gateway's) MAC**.

8.6 Ethernet and Switching

8.6.1 Ethernet Frame Structure

- **Preamble** (8 Bytes): For synchronization and clock rate.
- **Dst MAC** (6 Bytes), **Src MAC** (6 Bytes)
- **Type** (2 Bytes): Usually IP.
- **Data**
- **CRC** (4 Bytes): In trailer.

8.6.2 Switches

- Link Layer Device to store and forward Ethernet Frames.
- Has **no IP Address**; hosts don't know a switch is there.
- Maintains a **Switch Table** (MAC Address, Interface, TTL) to intelligently forward frames.

9 Network Security

9.1 Security Problems and CIA Triad

- **Eavesdroppers** → **Confidentiality** (Only sender/receiver understand message).
- **Deletion/Modification/Hijacking** → **Integrity** (Messages are not altered without detection).
- **Impersonation/Repudiation** (Denying a transaction) → **Authentication** (Sender/receiver confirm identity).
- **Denial of Service (DoS)** → **Availability** (Resource accessible to authorized users).

Security is a tradeoff with cost, usability, convenience, etc.

9.2 Cryptography (Crypto)

- Disguise data and recover it. $c = K_a(m)$, $m = K_b(c)$.

9.2.1 Symmetric and Asymmetric Encryption

- **Symmetric**: Same key for encryption (K_a) and decryption (K_b). Faster.
- **Asymmetric (Public Key Cryptography, PKC)**: Public key (K_B^+) for encryption, Private key (K_B^-) for decryption. Only one party (Alice) has the private key.
- Asymmetric is often used to securely share a symmetric **session key**.

9.2.2 Ciphers (Symmetric)

- **Caeser Cipher** (Shift number).
- **Monoalpha** (Weak to statistical analysis).
- **Block Cipher** (e.g., DES, AES).
  - **DES**: 56 Bit Symmetric Key, 64 Bit Blocks.

  - **AES**: 128, 192, 256 Bit Symmetric Key, 128 Bit Blocks.

9.2.3 RSA (Public Key Cryptography)

- Based on the difficulty of factoring large numbers.
- Public Key: (n, e), Private Key: (n, d).
- Encryption: $c = m^e \bmod n$, Decryption: $m = c^d \bmod n$.

9.3 Message Integrity

9.3.1 Message Authentication Code (MAC)

- Sender sends: $m + h(m + s)$, where s is an **authentication key** shared with the receiver.
- Attacker cannot change m to m' because they cannot recompute $h(m' + s)$ without s.
- **Repudiable**: Either Alice or Bob (or anyone with s) can sign it.

9.3.2 Hashing

- Much faster than encryption.
- Used for MAC, Block Chain, Proof of Work.

9.4 Digital Signatures

- Provides **Verifiability** and **Unforgability**.
- Uses the RSA reversible property: Bob encrypts message/hash with his **private key** ($K_B^-(m)$).
- Anyone can use Bob's **public key** (K_B^+) to verify the message came from Bob.
- Signature = $K_B^-(h(m))$

9.5 Public Key Infrastructure (PKI)

- Public Key needs to be on a secure broadcast channel to prevent man-in-the-middle attacks.
- **Certificate Authority (CA)**:
  - Issues and signs digital **Certificates** (Identity, Public Key, Time Window, Signature, etc.).

  - Binds a public key to an entity.
  - OS comes preinstalled with Trusted Root CAs (self-signed).
- Issues: CA is a bottleneck, and a single point of failure if malicious or breached.

9.6 Firewalls

- Isolate an organization's internal network from the larger Internet.
- Prevents DoS by dropping fake connections.

9.6.1 Types

- **Stateless Packet Filter**: Filters packet by packet based on Src/Dst IP, Src/Dst Port #, ICMP Type, TCP Syn/Ack bits (Uses Access Control Lists, ACLs).
- **Stateful Packet Filter**
- **Application Gateways**

9.6.2 Limitations

- IP can be spoofed.
- Can bottleneck actual traffic.
- Does not prevent all attacks.

9.7 Security at Different Layers

- **Application**: HTTPS (HTTP over SSL/TLS)
- **Transport**: SSL, TLS
- **Network**: IPsec and VPN
- **Link**: WiFi 802.11 (WPA2, WPA3)
- **Physical**: Port Access

10 Multimedia Networking

10.1 Applications

- **Streaming stored video**: Client side buffering; client plays earlier parts while server sends later parts.
- **VoIP (Voice over IP)**: Needs low latency.

10.2 Protocols for Real-Time Applications

- **Real Time Protocol (RTP)** (RFC 3550)
- **Session Initiation Protocol (SIP)** (RFC 3261)
- **Dynamic Adaptive Streaming over HTTP (DASH)**