The **transport layer services** include logical communication between application processes running on different hosts. The sender breaks messages into segments and the receiver recompiles these segments. TCP or UDP is used.

**Multiplexing** is a way to handle multiple sockets. It adds a transport header on the sender side that is later **demultiplexed** by the receiver so the segment can be delivered to the correct socket.

**UDP** is a bare bones transport protocol. It provides best-effort service that is vulnerable to loss, and out-of-order delivery. It is connectionless (no handshake required) and each segment is handled independently. Advantages include speed and simplicity.

A **Checksum** is used in UDP to detect errors. Checksum is the 1's complement sum of the segments content. This value is sent along with the packet. If the receiver computes the same checksum on arrival the packets contents are good.

**RDT** stands for **reliable data transfer** and were the precursors to TCP

- **RDT 1.0**: Assumes underlying channel is perfectly reliable. Uses separate FSM's (finite state machines) for both sender and receiver. Both send data to an underlying channel
- **RDT 2.0**: Assumes underlying channel is unreliable. Uses a checksum to check for errors. Uses Stop & Wait (sends 1 packet then waits for acknowledgement). Uses ACK's and NAK's to confirm successful receive or error. Problem, what if ACK is corrupted? What if duplicates are sent?
- **RDT 2.1**: Still uses ACK's and NAK's but has sequence number to remember what packet was last sent to avoid duplicates. Problem, receiver is unaware if last ACK/NAK was received by sender.
- **RDT 2.2**: Replaces NAK's with ACK's. If a packet is received bad, the receiver sends an ACK for the last OK packet. Duplicate ACK's result in retransmission. TCP uses the NAK free approach.
- **RDT 3.0**: Assumes the underlying channel can have complete data lost. Uses timeouts to determine if a packet should be resent. Duplicates still handled by sequence numbers. Receiver specifies sequence number be ACK'd.

**Pipelined protocols** allow multiple packets to be "in-flight" at the same time. Unlike where Stop & Wait only allows one

- **Go-Back-N:** Uses a sender window of N consecutive un-ACK'd packets. Each with individual sequence numbers and timeouts. Receiver always sends Ack for correctly received packet so far with highest in-order seq#. If an error occurs, starting from the bad packet, all segments retransmitted.
- **Selective Repeat:** Receiver individually acknowledges correctly received packets (out of order packets buffered for eventual deliver to lower layer in order). Only bad packets retransmitted. Errors can occur if the window is not half of the number of sequence numbers.

**TCP** is a transport layer protocol with many features. It uses **cumulative sequence numbers** (counting bytes of data) and **cumulative ACK's** (seq# of next expected byte).

- **RTT**: TCP uses estimated RTT and estimated deviation.
    - estRTT = (1- alpha) * estRTT + alpha * sampleRTT
    - estDvt = (1- beta) * devRTT + beta * |estRTT – sampleRTT|
- **Retransmission**: TCP uses a timeout period calculated with:
    - estRTT + (4 * devRTT)
- **Flow Control**: TCP specifies the number of bytes a receiver is willing to accept in RWND field in header. Typically, the free buffer space. Sender limits un-ACK'd data to this space.
- **3-way Handshake**: Sender sends SYN to server. Server responds with SYNACK. Client sends ACK for that with possible data.
- **TCP AIMD** stands for Additive increase (increase sending rate by 1 every RTT until loss is detected), multiplicative decrease (cut sending rate in half at each loss event). AIMD optimizes congested flow for networks worldwide and has desirable stability properties.
- **TCP Reno**, every time a loss occurs cut sending rate by half and increase from that point.
- **TCP Tahoe**, cut to 1 max segment size when a timeout is detected.

**Forwarding** (moving packets from a router's input link to the appropriate router output link). **Routing** (determine route taken by packets from source to destination).

**Data plane** is a local per router function, it determines how an arriving datagram is forwarded to router output port.

**Control plane** is network wide logic that determines how a datagram is routed amount routers along an end-to-end path from src to dst. Either uses traditional routing algorithms or SW defined networking (SDN)

**Longest Prefix Matching** uses the longest address prefix that matches the dst address when looking for a given link interface.

**Switching fabrics** transfer a packet from the input link to the appropriate output link. Three main types:

- **Memory**: Packets copied to system memory; speed limited by memory bandwidth
- **Bus**: Uses a shared bus, speed limited by bus bandwidth
- **Interconnection Network**: Exploits parallelism, datagrams use multiple paths.

**IP Datagram** consists of a 32bit identifier associated with each host or network interface.

- **Subnet:** Device interfaces that share common high order bits, typically 24bits of high order denote a subnet.
- **DHCP:** Host dynamically obtains IP from network server when it joins the network. Can also return address of first hop router, name & IP of DNS server, and network mask (network vs host portion).
- **NAT:** Network address translation, all devices on a local network share a single IPV4 address.
- **IPv6:** 128bit address that has no checksum, fragmentation/reassembly (compared to 32bit address of IPv4.
- **Tunneling:** The process of carrying an IPv6 datagram as payload in an IPv4 datagram among IPv4 routers (packet in a packet)
- **ICMP**: A error reporting protocol used to generate errors to source IP when network problems prevent delivery of IP packets.

**Routing Algorithms** can be global (all routers have complete topology, link cost info) or decentralized (iterative process, info exchanged with neighbors, initially only link costs of neighbors known).

- **Link State Routing**: A.K.A Dijkstra's, calculates the least cost path from 1 node to all others. All nodes have same info, N^2, all nodes need to be considered for each node.
- **Distance Vector:** Uses the Bellman Ford equation. Neighbors broadcast their estimates (distance vectors), updates occur to each distance Vector based on BF equation, if BF changes, notify your neighbors.
    - Dx(y) = minv {Cx, v + Dv(y) }

**Intra-AS Routing**: Uses RIP (distance vector exchanged every 30sec) or EIGRP (distance vector) or OSPF (link state, flood advertisements over IP to all routers in autonomous system (AS), use Dijkstra to calculate topology).

**Inter-AS Routing**: BGP (Boarder Gateway Protocol) allows subnets to advertise existence and the destination it can reach. eBGP obtains the reachability info of neighboring AS's and iBGP contains reachability info to all internal routers. Gateway routers must perform eBGP and iBGP.

**Error Detection (CRC)**: A powerful error detection technique that uses a bit pattern generator of r+1 bits. If <data, R> are exactly divisible by G, no errors. If a remainder exists, error. Can detect all burst errors less than r+1 bits.

**NIC**: Network interface controller that connects a computer to the internet (generally ethernet).

**Multiple Access protocols:** Idea, shared broadcast channel. Two or more simultaneous transmission causes interference, when 1 node wants to transmit it can send at the full rate, when M nodes want to transmit, they send at an avg R/M rate. Fully decentralized requiring no special transmission.

- **TDMA**: Time division multiplexing, access is given in rounds. Each node gets a fixed times slot to broadcast, unused slots go idle.
- **FDMA**: Frequency division multiplexing,  channel is divided into spectrum bands, each node is assigned a band, unused transmit time in band goes idle.
- **Slotted ALOHA**: When a node obtains a frame, transmit it in next time slot. If a collision occurs, retransmit frame in each subsequent frame with probability p until success.
    - Pros: Highly centralized, simple, single active node can transmit at full rate.
    - Cons: Collisions waste time slots, idle slots, clock sync
- **Pure ALOHA**: same as above, but no time slots, transmit immediately. Collision probability increases.
- **CSMA/CD**: Channel sensed idle, transmit frame, busy? Defer transmission. Collisions are detected quickly, those colliding are aborted reducing channel waste. Easy for wired, tough for wireless.

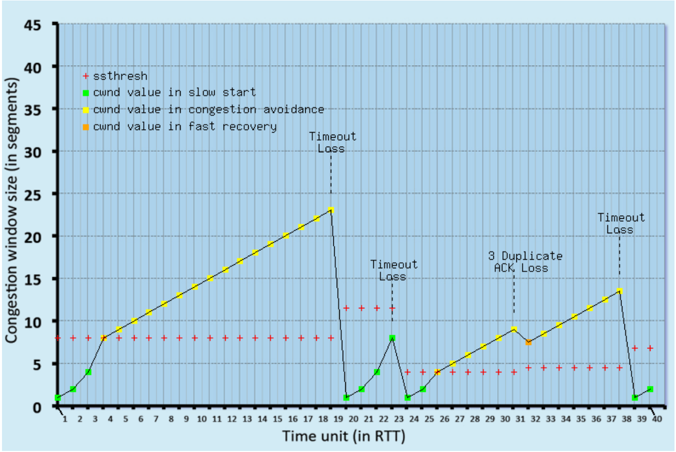
**MAC Addresses**: 48bit address that is portable (linked to HW generally). Unique for each device.

**ARP**: Address resolution protocol used to determine an interfaces MAC address based on its IP address. If a MAC address for the IP is not held within its local table, an ARP message is used to send a network wide request in which the target IP address responds with its MAC address which is then stored by the originator.

**Ethernet Switch**: Switches consider link layer & physical layer. It is a link layer device. It stores and forwards an ethernet frame and examines incoming frames MAC address and selectively forwards using CSMA/CD to access segment. Each switch has a switch table containing the MAC address of a host along with the interface to reach. These are self-learned, if not known flood network, target switch responds, result stored for later use.

**Putting it all Together:**

- DHCP is used when a device connects to a network to get IP address, Address of first hop, and DNS
- DNS is used to translate requests into IP addresses, if MAC address is unknown ARP is used to get MAC
- Once IP is known client opens TCP socket with webserver (TCP handshake) and sends HTTP request
- Webserver responds with content and webpage is displayed



Congestion window size (in segments)
45
40
35
30
25
20
15
10
5
0
+ ssthresh
🟩 cwnd value in slow start
🟨 cwnd value in congestion avoidance
🟧 cwnd value in fast recovery
Timeout Loss
Timeout Loss
3 Duplicate ACK Loss
Timeout Loss
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40
Time unit (in RTT)