INTRODUCTION

Network Layers (OSI model 7 layers; TCP/IP model 5 layers)

1. Physical layer
- The transmission of data on the physical medium of communication

2. Data link layer (frame)
- Ensures link-level reliability: data gets from one end of the link to the other
- Switches operate on this layer; MAC addresses used to uniquely identify nodes

3. Network layer (datagram/packet)
- Routing & forwarding: find a reliable set of links to connect the source and destination
- Fragmentation: to accommodate different mediums
- Routers operate on this layer; IP addresses used

4. Transport layer (segment)
- Ensures the end-to-end reliability: data gets from the source to the destination, and destination acknowledges
- Error checking, flow and congestion control

5. Application layer (/session, presentation)
- Connection control, translation of data formats, user's application interface, synchronisation, data integrity, error recovery

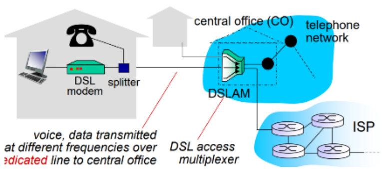
Encapsulation
- Additional information added as headers or trailers as data moves down the layers

Layer protocols
- Provide service interface used by higher level protocols
- Provide peer-to-peer interface used by different hosts at the same level

Cross layer design
- Violating the layered architecture: increased performance gains; but increases complexity when designing and at run-time, less reusability

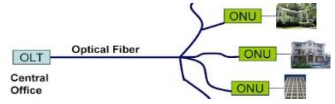
Access Networks

Wired
- DSL Modem
  - Dedicated access from home to CO
  - Uses twisted pair cables, existing telephone lines
  - ~2Mbps upstream, 24Mbps downstream, 5km



- Cable TV network
  - Broadcast network, different homes share the same access
  - Uses FDM for channels
  - Uses coaxial cables
  - ~2Mbps upstream, 24Mbps downstream, 5km

- Fiber network based (PON)
  - Optical Network Units: receives all data, accepts/rejects based on packet headers
  - Large bandwidth, low bit error rate (BER)
  - Uses optical fiber links
  - Up to 20km



Wireless
- LAN (local area networks)
  - 100ft, 54 Mbps (WiFi)
- WAN (wide area networks)
  - Cellular operators 10km, 1-10Mbps depending on 3G/4G/LTE/etc

Network Links

Broadcast links
- One node transmits, all nodes receive; bandwidth is shared by all hosts
  - Only one node can transmit at any one time, else collision
- Used in LAN (eg Ethernet)

Point-to-point links
- Each link connects one node to one node
- Used in switched networks

Another classification of links
- Full duplex: data can go both ways simultaneously
- Half duplex: data can go both ways, but not simultaneously
- Simplex: data can only go one way

Multiplexing: Multiple inputs sharing one output

Frequency Division Multiplexing (FDM)
- Every user given a different frequency band, every user transmits all the time

Time Division Multiplexing (TDM)
- Uses byte interleaving
- Fixed TDM
  - Data rate of output >= sum of rates of inputs
  - Each input link is given a fixed, dedicated bandwidth
  - Pros: Guaranteed service, no queuing delay, but cons: inefficient bandwidth U
- Statistical TDM
  - Data rate of output = sum of average rates of inputs
  - Pros: efficient bandwidth U, links do not remain idle when there is demand, but cons: no service guarantee, may have queuing delay or link congestion (buffer overflows)

Switching: Forwarding data from an input port to an output port

Circuit switching
- Forwards all data bits from input circuit to output circuit (a fixed bandwidth or slot)
- Used in telephone networks, SONET; uses FDM or TDM
- Phases: circuit setup (+overhead time), data transmission, circuit release
- Pros: dedicated resources, guaranteed performance
- Cons: if the data is not being transmitted, wasted bandwidth, low U

Packet switching
- Forwards packets based on header destination address; resources used as needed
- Store and forward
  - Packet is received in full, stored in buffer, and forwarded when free
  - Local forwarding table decides the output link
- Used in Ethernet switches, IP routers; uses STDM
- Pros: More users due to resource sharing, no call setup
- Cons: No guaranteed service, packet delay, possible buffer overflow (congestion)

Addressing
- IP Addresses
  - Unique within a network, changes each time host changes network (dynamic)
  - IPv4 (4 bytes, 32 bits) - 4 billion
  - IPv6 (16 bytes, 128 bits) - 3 x 10^{38}
  - IPv4 translated to IPv6 via tunnelling: IPv4 datagram carries IPv6 in payload (for backward compatibility)
- MAC Addresses (6 bytes)
  - Unique globally, hardwired at NIC (static)
- Address Resolution Protocol (ARP)
  - Every host on the same LAN knows IP+MAC addresses of other hosts
  - Every host & router keeps an ARP table <IP add, MAC add> for its network
  - When a source transmits to a destination
    - If source and dest have different network IDs, attach router MAC address
    - If source and dest have same network ID, attach dest host MAC address
  - A host copies a packet if it sees its MAC address in the packet header
- Address translation
  - Domain name to IP address: DNS
  - IP address to/from MAC address: ARP
- When a packet is being transferred, destination/source MAC address can change, but source/destination IP address stays the same (except in NAT)

Performance

Delay
- Message transfer time/latency (T_{total})
  - Transmission time ($T_t = \frac{message\ size\ in\ bits}{bandwidth\ in\ bps}$)
    - Time taken to put all the message bits onto the link
  - Propagation time ($T_p = \frac{link\ length}{propagation\ speed}$)
    - Time taken for a single bit to traverse from A to B
  - Queuing time (T_q)
    - Time waiting in queue before start of transmission; varies
- For one way, unacknowledged transfers
  - $T_{total} = T_t + T_p + T_q$
- For two way, acknowledged transfers
  - $T_{total} = T_t + 2T_p + T_q; RTT = 2T_p$
  - Assumes ACK packet size << message size

DB product
- Delay (Tp) x bandwidth = amount of data in the link

Throughput ($Throughput = \frac{message\ size}{message\ transfer\ time}$)
- Message size: normally in base 2 notation (1KB = 1024 bits)
- $Throughput\ rate = \frac{throughput}{bandwidth}$

DATA LINK LAYER (#2)

Channel capacity
- Shannon's theorem: gives an upper bound to the capacity of a link (max data rate)
- $C = B\ log_2(1 + \frac{S}{N})$
  - C: capacity (bps), B: bandwidth (Hz), S/N: signal-to noise ratio (NOT in dB)
  - Converting S/N in dB to NOT dB: $\frac{S}{N} = 10\text{\^{}}(\frac{\frac{S}{N} in\ dB}{10})$

Framing: breaking sequences of bits into frames

Sentinel-based framing
- Using a flag to delineate a frame (standard: 01111110)
- When flag appears in the payload, use
  - Bit stuffing in HDLC: sender inserts a 0 after 5 consecutive 1s, receiver always sees and removes the 0 unless it's a flag
  - Byte stuffing in PPP: sender changes "7E"→"7D 5E", "7D"→"7D 5D", flag remains "7E"

Counter-based framing
- Header includes the payload length in "count" field
- If count field is corrupted, can catch when the CRC fails, drop packet, wait for next SYN
  - If count field corrupts -> smaller, one frame is dropped, but if count field corrupts -> larger, corrupted frame and following frame is dropped

Clock-based framing
- Each frame is 125 microseconds long, special bit pattern is looked for every 125us
- STS-n carries n*(STS-1) frames, multiplexed using byte interleaving

Error detection using CRC
- Error correction is expensive and not feasible
- Let C(x) be a k-degree polynomial where $k \epsilon Z, k \ll n$
  - eg if k=3, C(x) has 4 terms ($x^3x^2x^1x^0$)

```
         11010111
      ____________
1101 | 10100110000
       1101
        1110
        1101
          1111
          1101
            1000
            1101
             1010
             1101
              1110
              1101
               011
```

k = 3
M(x) = 10100110 (message)
C(x) = 1101 (divisor)
T(x) = 10100110 000 (M shifted k bits)
R = 011 (remainder)
P(x) = 10100110 011
= (T-R) or (M append R)
= what sender transmits

If $\frac{received\ message}{C(x)} = 0$, receiver assumes no error

Selecting C(x)
- If x^k and x^0 have non-zero coefficients, can detect all single bit errors
- If C(x)%(x+1) = 0, all odd number errors can be detected
- All burst errors of length <= k can be detected

Flow Control: ensures sender does not overwhelm the receiver
- Congestion control is concerned with a link, flow control is node to node
- U: link utilisation, the ratio of time spent transmitting given data frames to the total time link is engaged until transfer is done
- Pf: frame error probability
- a = Tp/Tt = DB/message size = number of frames on a link at any point in time

Automatic Repeat Requests (ARQ)
- Stop and wait
  - TIMEOUT mechanism, alternates between ACK0 and ACK1
  - Simple implementation, but poor link utilisation, slow
  - Fragmentation can help: errors detected sooner, faster retransmission of error frames, host occupies medium for less time, but stop and wait for more frames
  - Performance
    - No errors: $U = \frac{T_t}{T_{total}} = \frac{1}{1+2a}$
    - With errors: $T_{total} = \frac{T_t+2T_p}{1-P_f}$, $U = \frac{1-P_f}{1+2a}$
- Sliding window
  - ACKn/RRn → receiver has received up to n-1 numbered frames
  - Receiver and sender have length W buffers, W frames can be sent without ACK
  - Frames are numbered 0 to 2^k
  - Sender: when frame is sent, W shrinks; when ACK is received, W expands; unacknowledged frames are kept in buffer
  - Receiver: when frame is received, W shrinks; when ACK is sent, W increases
  - Typical W for LAN: 7, for WAN: 127
  - Performance without errors
    - If (W >= 1+2a) $U = 1$, $\text{num of frames sent/s} = \frac{1}{T_t}$
    - If (W < 1+2a) $U = \frac{W}{1+2a}$, $\text{num of frames sent/s} = \frac{W}{T_t+2T_p}$
- Go back N
  - Max W = 2^k - 1
  - Receiver sending REJ-i discards all frames from i onward
  - Performance with errors
    - If (W >= 1+2a) $U = \frac{1-P_f}{1+2aP_f}$
    - If (W < 1+2a) $U = \frac{W(1-P_f)}{(1+2a)(1-P_f+WP_f)}$
- Selective reject
  - Max W = 2^{k-1}
  - Receiver sending REJ-i discards frame i only, subsequent frames buffered
  - Implementation is more complex, buffer must be large enough
  - Performance with errors
    - If (W >= 1+2a) $U = 1 - P_f$
    - If (W < 1+2a) $U = \frac{W(1-P_f)}{1+2a}$

Ethernet

- Maximum length 2500m using 5 segments
- Frame format
  - Minimum frame size is 64 bytes, required for collisions to be detectable
    - For 2500m Ethernet link, RTT = 51.2us; in a 10Mbps link, 51.2us = 512 bits; 512 bits = 64 bytes
    - If collision occurs on the opposite end of the cable, sender needs to be transmitting still when other sender transmits to detect the collision
  - Uses MAC addresses, CRC-32

Carrier sense multiple access - collision detection (CSMA-CD)

- Carrier sense: a host can sense whether the link is in use
  - If a host detects a link is idle, will send with a probability of 1 (1-persistent CSMA)
- Collision detection
  - A host listens while it is transmitting, can detect if there are any collisions

Exponential back-off algorithm when collision occurs

- Send a noise burst + preamble to inform other hosts of collision
- Wait for n slots, n ε [0, 2^i-1], randomly chosen, in the i-th consecutive collision
  - Interval increases to maximum of 2^{10}-1
  - Host gives up after 16 tries

LAN connections

- Bus (traditional)
  - Single collision domain, 10 Mbps shared by all hosts, only one can transmit at any time, a cable cut disconnects the entire network
- Hub/star configuration
  - Single collision domain, 10 Mbps shared by all hosts, jam signal generated by the hub, a cable cut does not disconnect the entire network
  - All frames are copied to all ports
- Switching hub/switch
  - Store and forward packet switching at the hub, collision domain is each port, more than one host can transmit simultaneously
  - Frames are forwarded to the correct port

LAN extensions

- Using bridges to connect multiple LANs
- Spanning tree/transparent bridges
  - Hosts do not need to know about the bridges
  - Backward learning: switches learn the port that a host can be reached by, maintains a forwarding table <host, port>; if unknown destination, will broadcast to all ports (but could lead to infinite loop if multiple bridges between 2 LANs)
- Spanning tree algorithm
  - To avoid loops within the tree: choose the shortest path of every node to a root node (bridge with smallest ID)
  - Message format: <Y, d, X>
    - Y: what bridge thinks is the root bridge
    - d: distance from root to itself
    - X: bridge's own ID
  - Initially, every bridge sets itself as root and broadcasts its message. Compares its own Y to other messages, updates and resends to neighbouring bridge.
  - Once the algorithm stabilises, only root will send messages periodically and other bridges will repeat it. If no messages are received for a period of time, algorithm resets to reselect the root bridge.

NETWORK LAYER (#3)

Header includes: IP version, header length, length, frag identifier, flags, frag offset, TTL (changes at each hop), checksum, source IP, dest IP

Routing, forwarding and fragmentation

Virtual Circuits, Datagram Networks

Virtual circuits

- Connection oriented: buffers, CPU, bandwidth etc reserved for a data transfer session
  - Path for first data packet is followed by all other packets
  - More reliable, but more costly
- Common packet header for all packets
- Dumb end systems, complexity inside network
- Used in ATM for telephones

Datagram networks

- Connectionless service: no resources reserved, dynamic paths, each packet has its own header, less reliable, but cheaper, no set-up cost
- Smart end systems, complexity at network edges
- Used by IP networks (no strict timing requirements)

Internet Protocol

Fragmentation

- Large datagrams divided if datagram exceeds maximum transfer unit size, reassembled at destination's Network Layer
- Fragmented datagrams have the same "ID", "fragflag" = 1, "offset" determining order

IP addresses

- IPv4 (4 bytes, 32 bits), IPv6 (16 bytes, 128 bits)
- IP address can be hard-coded by router, or Dynamic Host Configuration Protocol (DHCP) dynamically assigns addresses

Subnets

- Each interface from a router forms an isolated network = subnet
- Classless Interdomain Routing (CIDR) notation: /24
  - eg 192.168.45.0/21
    - From 21, subnet mask: 255.255.248.0 (248 = 11111000)
    - From 192.168.45.0, network ID: 192.168.40.0 (45 = **00101**101)
    - Broadcast network: 192.168.47.255 (192.168.**00101**111.11111111)
    - # of valid hosts: 2^{32-21} - 2 = 2^{11} - 2 = 2046
    - If more subnets were created, for example using 5 bits, then # of subnets = 2^5 = 32, # of valid hosts in each: 2^{11-5} - 2 = 62
  - # of valid hosts removes 2: broadcast ID (all 1s), unassigned network ID (all 0s)
  - Better than class-based: allows for finer add resolution, more efficient allocation
- Addresses managed by ISP → managed by ICANN

Network address translation (NAT)

- Router uses its IP address for all datagrams leaving its local network (router IP, port2)
  - Within the local network, can have its own addressing scheme
  - Remote hosts respond to <router IP, port2>
  - NAT translation table to track <source IP, port1> map to <router IP, port2>
- Allows multiple devices to use the same IP address
  - Increase in number of Internet accessing devices
  - Separation of local and global network
  - Security: cannot access local devices outside of local network
- However, accesses layer 4, violates e2e
  - Address shortage should be solved with IPv6 instead

NAT traversal problem

- When remote clients want to connect to local client (how to specify who?)
- Solution 1: static NAT configuration
- Solution 2: universal plug and play (UPnP) internet gateway device protocol
  - Local host allowed to configure its own static NAT port
- Solution 3: relaying (eg Skype)
  - External client acts as relay (eg signing in to an application so that they get your IP address); connection initiated by clients, relay bridges packets

Routing

Router architecture

- Input port → switching fabric → output port (slowest one introduces queuing delay)
- Input port
  - Given a destination, use forwarding table to lookup output port
- Switching fabric
  - Memory, bus or crossbar (in increasing order of switching speed)
- Output port
  - Scheduling: choosing which to transmit among queued datagrams

Routing algorithms

- Link state/centralised (Dijkstra)
  - All routers should have complete topology & cost info
  - Start off with source node in "visited" set, find least cost edge from any node in "visited" set, add that node (x) to "visited"; set that as shortest path to x.
  - Will always terminate after n-1 iterations
  - Might be incorrect answer for negative weights, cannot work for negative cycles
- Distance vector/decentralised (Bellman-Ford)
  - Routers only need to know physically connected neighbours
  - Each node maintains a table of its cost to every other node, sends its table to its neighbours, every node compares and updates its tables based on its neighbours'
  - At any change, the node that changed recomputes and broadcasts
  - Count to infinity
    - If node X travels through Y to get to Z, edge Y to Z updates, Y will try to get to Z through X, which is currently getting to Z through Y (will take many iterations to settle)
    - Count to infinity if the link Y to Z breaks: unreachable
  - Poisoned reverse
    - If Y tries to get to Z through X, X tells Y its distance is infinity

Routing protocols

- Intra-AS (aka Interior Gateway Protocols): within an autonomous system
  - Routing Information Protocol (RIP): distance vector
  - Open Shortest Path First (OSPF): link state
  - Interior Gateway Routing Protocol (Cisco proprietary)
- Inter-AS: between different autonomous systems
  - Border Gateway Protocol (BGP): standard protocol for the Internet

Broadcast routing

- Uses in-network duplication to avoid source duplication
- Uses spanning tree to ensure no flooding

Multicast routing

- Find a tree connecting routers having the multicast group members
  - Using Dijkstra from a specific source to all the receivers
  - Using Steiner Tree to find minimum cost tree (no specific source)
    - This is not used: an NP-complete problem

TRANSPORT LAYER (#4)

Ensuring end-to-end reliability, error checking, flow and congestion control

Protocols

Header includes: source port, dest port, seq number, ack, header len, flags, window, checksum

- UDP: User Datagram Protocol
  - Minimal, no setup required: no overhead and unnecessary delays
  - But not reliable delivery, optional error check, no retransmissions, not in-order
  - Used for applications like multimedia streaming, DNS queries
- TCP: Transmission Control Protocol
  - Connection set-up, discard/retransmits corrupt packets, packet acknowledgement, in-order delivery, flow control, congestion control
  - Multiplexing based on port numbers

TCP Connection

Checksum

Sequence numbers

- Initial sequence number (ISN) do not start from 0, to prevent mixing up with old packets or new connections
- ISN set from a 32-bit clock that ticks every 4us, hosts exchange ISNs

Handshake

- Three-way handshake: A sends SYN to B, B returns SYN ACK, A sends ACK before starting to send data
- ISN included in the handshake
- Initial TIMEOUT value estimated: 3-6 seconds

Closing a connection

- A sends FIN to close & get remaining bytes/RST to not get remaining, B sends FIN ACK

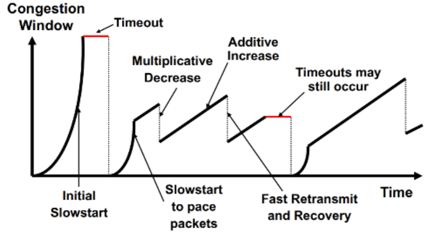
TCP Reliable Delivery

Retransmissions: when packet is lost, ACK is lost, or early timeout

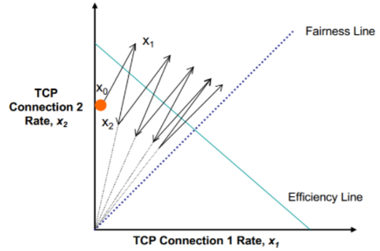
- Determining timeout
  - Timeout set as a function of RTT plus fudge factor to account for queuing
  - Estimated RTT = a*Estimated + (1-a)*Running Avg ; Timeout = 2*Estimated
  - Only add the RTTs of segments sent a single time
    - May be duplicate packets for lost ACKs or early timeouts, RTTs unreliable
- Triple duplicate ACKs (4^{th} ACK) for fast retransmissions
  - Timeout-based retransmissions are slow
  - With sliding window, triple duplicate ACK-n indicates receiver is still awaiting nth packet - hint that packet is lost
  - Works best for long data transfers, large W, low burstiness of packet loss

Sliding window

- Flow control window size (FW)
  - Receiver initially advertises this window to sender; receiver needs to be able to store this amount of data without overflow
- Congestion control window size (CW)
  - Amount that the network is able to handle without loss; set by sender
  - Additive Increase/Multiplicative Decrease (AIMD): congestion control & avoidance
    - Probes the network congestion limits
    - CW increases by 1 for every CW's worth of packets acknowledged
    - CW cuts to 1 for timeouts, cuts to half for triple duplicate ACKs (TCP Reno), or cuts to 1 for triple duplicate ACKs (TCP Tahoe)
  - Slow start
    - CW = 1 initially; increase CW by 1 (CW x2 each time) for each ACK returned
    - Until reaching the value of ssthresh (ssthresh = the previous timeout CW/2)
    - After that, additively increase CW



- Actual window size W
  - W = min(CW, FW)
- TCP fairness using AIMD
  - TCP allocates on average the same resources to different flows (equal distribution)
  - Other notions of fairness: proportional fairness (proportional to requested amount), max-min fairness (maximise the minimum resources first)



TCP throughput: proportional to window size

- Throughput = W/RTT (packets per second)

End-to-end principle

- Network functions are implemented at the end hosts of the communication session (keeping in the in-betweens of the network as dumb as possible)
- Rethinking e2e: flow recognition (handling different flows differently, no longer just looking at packets)
  - Shared bandwidth queuing (no hogging), explicit loss notification, flow security