1 Basic Concepts

- **Circuit Switching** (connection-oriented) requires expensive setup phase but little processing required after setup. Resources are reserved at setup time guaranteeing Quality of Service, but limits resource sharing
- **Packet Switching** (connectionless) has no setup cost but incurs processing cost for forwarding and space overhead as every packet is self contained. Designed to share links achieving better network utilisation

Definition 1.1 (Quantifying Data Transfer)

Bandwidth $R = \frac{L}{t_2 - t_1}$ *or* $\frac{transferred}{duration}$ *amount that can get transferred in time unit*

Throughput *amount that actually gets transferred in time unit*

Latency *(propagation delay)* $d = t_1 - t_0$ *or* $\frac{distance}{propagationspeed}$*, time it takes for one bit to go through connection*

Packetization *(transmission delay/store-and-forward delay)* $\frac{L}{R}$

Transfer Time $\triangle = d + \frac{L}{R}$ *= propagation delay + transmission delay*

Router Delay *= processing delay* d_{prop} *(check bit error, determine output) + queuing delay* d_q *(wait at output)*

Traffic Intensity $= \frac{La}{R} = \frac{packet\ length \times\ avg\ packet\ arrival\ rate}{link\ bandwidth}$ *If* $\frac{La}{R} \to 1$*,* d_q *large. If* $\frac{La}{R} > 1$*,* d_q *infinite.*

2 Application Layer

2.1 Hyper Text Transfer Protocol (HTTP) TCP

- **HTTP/1**: (1996) 1 TCP connection per object leading to inefficent use of network/OS
- **HTTP/1.1**: (1997) Introduced persistent connections where same TCP connection used for multiple requests with multiple replies (can be pipelined). "Connection:close" by client/server to indicate otherwise
- **HTTP/2**: (2015) Binary content and no longer needs to be ordered. **HTTP/3**: Exchanges in UDP/QUIC.
- **Request**: GET /a.html HTTP/2 HEADERS \n **Response**: HTTP/2 200 OK HEADERS \n OBJECT
- **Methods**: GET, POST, HEAD, PUT, DELETE, OPTIONS (not cacheable)
- **Status**: 1XX Informational, 2XX Success, 3XX Redirection, 4XX Client Error, 5XX Server Error
- **Cache** control by client/server using Cache-Control: no-chache/max-age=20;must-revalidate / Expires:
- **Stateless** protocol but use Set-Cookie: (from Server) and Cookie: (from Client) header for stateful sessions

2.2 Domain Name System (DNS)

- **A** IP Address **NS** Authoritative Name Server **CNAME** Alias Domain Name **MX** Mail Server Domain Name
- **Round Robin** DNS for load balancing: Short TTL, Order of IP addresses returned changes
- **Non-authoritative** means reply was extracted from cache

2.3 Content Delivery Network (CDN)

- **Enter Deep**: Push servers deep into many networks close to users
- **Bring Home**: Smaller number of larger clusters in Points of Presence near access networks

2.4 Simple Mail Transfer Protocol (SMTP)

- **Sends emails**: Set up TCP/IP, client requests server to accept messages, server responds and client sends
- HELO \n MAIL FROM: .. \n RCPT TO: .. \n DATA \n other headers \n \n content \n . \n QUIT
- Oblivious to message content, but every receiving SMTP server must add a Received: header
- Plain text unless ESMTP(SMTPS) used (EHLO greeting).

2.5 Post Office Protocol (POP3)/Internet Message Access Protocol (IMAP)

- telnet pop3.a.com 110: OK, USER .., OK, PASS .., OK, LIST, 1 2505 \n ., RETR 1 + DELE 1 + QUIT
- POP3 iimplicitly assumes mail is deleted at server, IMAP solves this problem

2.6 File Transfer Protocol (FTP)

- **Active Mode**: Client "PORT x" to Server:21. Server ACK, connect to client from 20 to x. Client ACK.
- **Passive Mode**: Client "PASV" to Server:21. Server "PORT x". Client connects to X.

3 Transport Layer

- 0-1023 reserved, 1024-49151 registered user applications, 49152-65535 dynamic
- FTP TCP/20-21, SSH TCP/22, HTTP TCP/80, HTTPS TCP/443, DNS UDP/53, SMTP TCP/25, POP3 TCP/110, IMAP TCP/143, DHCP UDP/67-68
- max output rate = $\frac{W}{RTT}$, $timeout = \overline{RTT} + 4\sigma^2_{RTT}$, $U = \frac{\text{actually used network}}{\text{could have used network}} = \frac{d_{trans}}{RTT+d_{trans}}$

3.1 Transmission Control Protocol (TCP)

- Full-duplex service so both endpoints can send and receive simultaneously
- Server: socket, bind, listen, accept, read + write, close. Client: socket, connect, write + read, close
- **Maximum Segment Size (MSS)**: max data transmitted in a single segment
- **Maximum Transission Unit (MTU)**: largest link-layer frame avaliable to sender host
- **Sequence Number**: Position of first byte in segment, random initial sequence number (ISN) on set up
- **Acknowledgement Number**: First sequence number not seen by receiver (usually every other)
- **Congestion Window**: LastByteSent - LastByteAcked $\leq$ W = min(CongestionWindow, ReceiverWindow)
- **Headers**: 20+ bytes: source & dest ports (2x16), seq no (32), ack no (32), offset - header length in 32bits/Bbytes (4), reserved (6), URG **ACK** PSH RST **SYN FIN** (6), receive window (16), checksum (4), urgent (16)
- **Three Way Handshake**: SYN+ClientISN, SYN+ACK(ClientISN)+ServerISN, ACK(ServerISN)+ClientSeq
- **Client**: (CLOSED) open/SYN (SYN_SENT) SYN,ACK/ACK (ESTABLISHED) close/FIN (FIN_WAIT_1) ACK/ (FIN_WAIT_ FIN/ACK (TIME_WAIT - this side closed it) wait x seconds/ (CLOSED)
- **Server**: (CLOSED) open/ (LISTEN) SYN/SYN,ACK (SYN_RCVD) ACK/ (ESTABLISHED) FIN/ACK (CLOSE_WAIT - other side closed it) send FIN/ (LAST_ACK) ACK/ (CLOSED)
- **Disconnect**: Client FIN, Server ACK, Server FIN, Client ACK
- **Stop and Wait**: Send 1 packet, requires receiving ACK before sending next
- **Acks**: in order + prev acked: wait x ms then ACK, in order + prev not acked: cummulative ACK, out of order (gap): duplicate ACK (NACK), segment that fills gap: ACK if segment at lower end of gap
- **Slow Start (SS)**: Initial W = MSS, doubled every ACK until ssthresh (initially receive window/very high)
- **Congestion Avoidance (CA)**: $W = W + MSS \times \frac{MSS}{W} \approx W + MSS$ until congestion detected
- **Additive-Increase/Multiplicative-Decrease (AIMD)**: at packet loss, half W
- **Fast Recovery**: Set ssthresh to W/2. Timeout: W = MSS then SS. NACK: W = W / 2 then CA.
- **Congestion control** aims not to overflow network, **Flow control** aims not to overflow receiver

3.2 User Datagram Protocol (UDP)

- Max = 20B IP header + 8B UDP header + 65,507B data = 65,535B
- **Header**: 8 bytes: source & dest ports (2x16), length (16), checksum (16)
- Finer Application Layer control over what data sent when, no connection establishment/state, smaller header - used for apps that require speed/do not care if data dropped
- Used to support QUIC (2012).

4 Network Layer

- Area Networks: Personal Local Metropolitan Wide
- **Dynamic Host Configuration Protocol**: Discover, Offer, Request, Ack
- **Internet Control Message Protocol (ICMP)**: echo reply 0, echo 8, destination unreachable 3, TLE 11

4.1 Internet Protocol (IP)

- **Header**: 20+ bytes: total length (16), identification (16), fragment offset - in 64bits/8Bytes (13), **M**ore Fragments Follow (1), header checksum (16), source & dest address (2x32)
- **Fragmentation** if input datagram exceeds MTU of output link, only reassembled at destination
- **Class**: A (1.0.0.0 - 127.255.255.255/8), B (128.0.0.0 - 191.255.255.255/16), C (192.0.0.0 - 223.255.255.255/24)
- **Private**: 10.0.0.0 - 10.255.255.255/8, 172.16.0.0 - 172.31.255.255/12, 192.168.0.0 - 192.168.255.255/16
- Loopback: 127.0.0.0/8. Link Local: 169.254.0.0/16 (error acquiring IP address)
- First subnet address is network address, last subnet address is broadcast address.
- Routers match the longest prefix possible. If not in forwarding table, default port/depends on algo.
- Network Address Translation violates machine identifiable by IP and make Internet connection-oriented
- IPv6: uses 128bit/16byte addresses

4.2 Routing

- Routers collaborate to build sink tree of optimal routes
- **Shortest Path Routing** Dijkstra's Algorithm
- **Flood Routing**: Forward to every output (selectively/once) except original, hop counter to avoid drowning
- **Distance Vector Routing**: Bellman-Ford. Take direct neighbours' advertised cost to dest, advertise min (neighbours' cost + cost to neighbour) to neighbours. Count-to-infinity problem if dest goes down (RIP)
- **Link State Routing**: Discover direct neighbours, constructs Link State Advertisement by calculating cost to them, collect/flood to all routers, run Dijkstra locally. HELLO to discover, ECHO to measure. (OSPF)
- AS may be divided into areas, area border routers route traffic to backbone area
- **Reverse-path Forwarding broadcast** every router forwards to all neighbours except to original, routers only accept broadcast packet originating in A if on a unicast path between themselves and A
- **Core-based trees multicast**: single spanning tree per group with a root near middle, multicast sent to root, which performs multicast along tree
- **Hierarchical Routing**: Gateway routers discover inter-AS info + intra-AS info propagated within AS
- **Path-Vector Protocol**: Router advertises routes (AS-PATH sequence of AS ad sent through + NEXT-HOP interface/address to forward packets to) to others, AS choose to accept ad (politics/cost), AS forward ad. Routes ranked by preference value, shortest AS-PATH, closest NEXT-HOP. Count-to-infinity solved by withdrawal updates.

5 Data Link Layer

5.1 Ethernet

- Introduced in 1980 with 2.94Mbps coaxial (10BASE5), standard in 1983 (IEEE 802.3)
- Unshielded Twister Pair, Shielded/Screened TP, Foiled TP, Shielded & Foiled TP to protect against EMI
- 100Base-TX Fast Ethernet Cat5 UTP 100m 100Mbps. 100Base-T Gigabit Ethernet Cat6 100m 1000Mbps.
- Straight-through (different layers, Media Dependent Interface), Crossover (MDI with Crossover, same layer), Rollover (directly into device - troubleshoot). Modern - fake swap with Auto-MDI/MDIX
- **Ethernet II Header**: 22+4bytes: preamble (8), dest & src address (2x6), type (2), data (46-1500), fcs (4)
- **MAC Address**: 6bytes, Organisationally Unique Identifier (7-Universally / Locally administered, 8-Individual / Group address) that is manufacturer specific and unique, Network Interface Controller Specific

5.2 Switch

- Use Forwarding Information Base to remember which port is which MAC
- Store-and-Forward (receive whole frame, check errors, forward), Cut-through (forward when enough info)
- Switching Loop results in broadcast flooding. Use (Rapid) Spanning Tree Protocol so switches maintain a updated spanning tree using Bridge Protocol Data Units

5.3 Topology

- **Bus**: Main coaxial cable to connect all hosts. BNC T connector for new host, BNC terminator at each end.
- **Ring**: Each host uses 2 NICs and data flowed one way. If link cut, network died (unless designed to adjust).
- **Token Ring**: MultiStation Access Unit to connect hosts in (logical) ring. If host dies, it stops getting token. Listen - keeps copy of frame addressed to it and forwards rest of stream. Transmit (host with token) - read & transmit from memory, removes frames it sent then releases token, early release mode means token can be released before frames it sent is back. Frame Status - A for dest host working, C for correctly read.
  Priority - only claim token if data priority $\geq$ token priority. Reservation - host can increase reservation priority (encoded in sent packets). Active Monitor station (by election) to generate tokens and drain orphaned frames.
- Fiber Distributed Data Interface: Class A 2 rings, B 1. B failure affects 1 ring, A failure creates short circuit.
- **Star**: Single Point of Failure

5.4 Wi-Fi

- Wi-Fi 0 (1997). Wi-Fi 5 802.11ac (2014) 5GHz 433-6933Mbps. Wi-Fi 6 802.11ax (2019) 2.4/5/6GHz 574-9608Mbps.

5.5 Medium Access Control

- Static: Time Division Multiplexing or Frequency Division Multiplexing.
- ALOHA (1971): stations wait random time after collisions, slotted transmissions to reduce vulnerable period
- Carrier Sensing Multiple Access: check channel idle before transmission, collisions due to transmission delay
- Collision Detection: jamming signal if detected, min frame length 2 x end-to-end transmission delay (512bits)
- Back-off when busy channel: 1-persistent - keep checking (Ethernet: exponential - rand$(0, 2^c - 1)$ * min frame len), Non-persistent - wait random time, p-persistent: keep checking, if free transmit with probability p.

6 Physical Layer

- Patch Panel: Socket Panel (cable ends up), Network Switch (LAN), Private Branch Exchange (phone systems)
- Coaxial Cable: considerable shielding, wider range of frequencies, higher cost per meter (still used by TV)
- Microwave 4-11Ghz, Satellite .5-10GHz, UHF .3-3GHz TV, VHF 30-300MHz, HF 3-30MHz, MF .3-3MHz
- Baud rate - symbols per second. Digital in analogue using modem, analogue in digital using codec.
- **Modulation**: aka Shift Keying. Amplitude (high=1,low=0), Frequency (increase=1,base=0), Phase (change=1)
- Digital Subscriber Line: Remove phone line limit of 3000Hz. Asymmetric DSL: split into channels, more downstream than upstream. ADSL modem, then DSL Access Multiplexer recovers bit signal.
- Copper or fiber slows speeds to 2/3 of speed of light.

7 Security

- **H**ackers **P**hreakers **V**irii **A**narchists **C**rackers
- **Firewall** can be Application-level gateway (runs on host, protects host), Proxy server (runs on network, protect LAN), Circuit-level gateway (takes over host's communication) or Packet Filtering (Stateful/Stateless)
- **Proxies** can be normal (client aware), transparent (network level) or reverse (CDN server)
- **Bastion Host** expects attacks so logging/auditing, runs secure minimal OS and dedicated terminal
- **DMZ**: Neutral zone, external only can speak to hosts in DMZ, use NAT to forward to protected internal
- **Port Forwarding**: Certain ports forwarded directly to internal host/port. Useful for hiding internal
- Get around firewall by tunneling with ssh, spoof IP/MAC address, using VPN to hide activity
- Public Key encryption is slower but more secure as owner does not have to disclose key
- **Diffe-Helman Key Exchange**: A and B agree on g and p, with secret values a and b. The public value is $x = g^b \mod p$ and $y = g^a \mod p$. The secret key is $y^b \mod p = x^a \mod p = g^{ab} \mod p$ or **Kerberos**
- **Wireshark**: Promiscuous (wired/wireless) keep everything. Monitor (wireless) listen on all networks
- **NMap**: -sn scan network w/o checking port, -p X-Y scan ports (range optional), -Pn scan w/o discovery