Introduction

Def Internet Service Provider (ISP): different tiers, may connect to 1+ high-tier ISPs.

Def Encapsulation: adding headers. **Message** $\xrightarrow{\text{transport}}$ **Segment** $\xrightarrow{\text{network}}$ **Datagram** $\xrightarrow{\text{link}}$ **Frame**.

Taxonomy of networks:

(through *physical media*) **guided** (fiber optics / coaxial cable) ; **unguided** (atmosphere).
(through *topology*) bus; star; ring; tree.
(through *scale*) personal to local to metropolitan to wide to the Internet.
(through *services*) connection-oriented/connectionless; reliable/unreliable (**Reliability**: multiple meanings including whether content is ensured correct and whether the order in which packages are tranported is correct).

Protocol: complex. **Layering** introduced to (1) reduce complexity, (2) maintain independence between each other. **Interfaces** introduced to use services provided.

Def Requests for Comments (RFC): the Internet standards developed to specify the content of protocols. Public protocols are specified in RFCs while proprietary ones are not.

Structure of networks: **Network Edge** (applications and hosts) + **Access Networks** (physical media, wired/wireless communication links) + **Network Core** (interconnected routers).

Network Edge: client/server model or peer-peer model. P2P model has minimal or no use of dedicated servers.

Access Networks: used to connect hosts to edge router, including residential, institutional, mobile access networks. Parameter: **Bandwidth** and **Data Rate**.

Def Bandwidth (in Hz): the range of frequencies transmitted without being strongly attenuated. "2.4GHz/5GHz" refers to the central frequency.

Def Data Rate (in bits/sec): the rate at which bits are transmitted.

Thm Shannon's Theorem: suppose the signal-to-noise ratio is S/N, maximal possible data rate is R, bandwidth is B, then $R = B\log_2(1 + S/N)$.

Residential Access: point to point access $\to$ cable modems. P2P access include Dialup via modem (only one surfer) and ADSL (asymmetric digital subscriber line, not effected by neighbour). Cable modems share bandwidth with neighbours.

Nework Core: circuit switching and packet-switching.

Circuit Switching: reserve bandwidth for communicating hosts. Establishing connection hard and wastes idle resources. Techniques include **Frequency Division Multiplexing** and **Time Division Multiplexing**. TDM is suitable for tasks like video streaming (fidelity > continuity) while FDM is suitable for tasks like phone calls (real-time transferring > fidelity).

Packet Switching: users share resources, while routers store packages and forward. Since sequences of all users' packets are mixed together, it is **statistical multiplexing**. Not suitable for real-time services because delays are unpredictable.

Delays: nodal processing delay, queuing delay, transmission delay, propagation delay, total nodal delay.

Processing Delay: time to examine header and decide direction.
Queuing Delay: time before earlier-arriving packets finish transmitting.
Transmission Delay: length of packet divided by transmission rate.
Propagation Delay: time of one bit's travelling. (e.g. γc_0 where $\gamma < 1$ for optic fiber)
Total Nodal Delay: $d_{\text{nodal}} = d_{\text{proc}} + d_{\text{queue}} + d_{\text{trans}} + d_{\text{prop}}$.

Def Traffic Intensity: Let packates L with bits arrive at a packates/sec, and R is transmission rate, then La/R is the traffic intensity. As traffic intensity increases to 1, the average queuing delay diverges to infinity.

There is package loss with limited buffer. And throuput is $\min\{R_1, R_2, ..., R_n\}$ given transmission rates $R_1 \sim R_n$.

Application Layer

Two architectures: **client-server** vs. **peer-to-peer** (*self-scalability*, peer geneates workload and also add capacity through distribution of data).

Unit of communication: **process**es. Within same host, processes communicate through OS-defined **inter-process** communication. The **client** the initiater of communication (both architectures!). Identified through `<IP>+<PortNum>`.

Def Socket / API: interface between application layer and transport layer.

Web and HTTP

The first Internet application. Web Browsers implement the client side of HTTP, while Web Servers implement the server side, and are addressable by URL.

Def Objects: the elements of a Web page (usually a base HTML-file accompanied by referenced objects). It's simply a file (.html/.jpg/...) addressed by a URL (host name + path name).

The HyperText Transfer Protocol defines how clients request Web pages and how servers transfer Web pages, running on TCP. It is *stateless* (maintaining no information about the client), *non-persistent* (connection shut down immediately, v1.0) or *persistent* (v1.1).

Def Round Trip Time: time forth and back. Non-persistent TCP connection requires $2 \times$ RTT per object while with pipelining, persistent TCP requires minimum $1 \times$ RTT for all objects.

Web Caches (Proxy Server): (located in belonging ISP) connection with origin server only if the object requested is not in proxy server. Serves to (1) reduce response time, (2) reduce traffic, (3) help poor servers deal with requests.

The File Transfer Protocol: separates data connection and control connection. (*out-of-band* control, in contrast to *in-band* control)

The Simple Mail Transfer Protocol: *mail server* and *user agent*. Mail servers have mailbox and message queue. They are client when sending, server when receiving.

The Domain Name System: offering *host alising* and *load distribution* (get IP addresses). Two parts: (1) distributed database, (2) application-layer protocol. UDP protocol. *Distributed*. *Hierarchical* query: ask **root** server for **Top-Level Domain** server address, and then ask TLD for **Authoritative** server address, finally ask ADNS to get required IP. Either *recursive* (let server ask server) or *iterative* (client ask everyone) query. **Local Name Server** obtained by ISP acts as a proxy.

Transport Layer

Multiplexing and **Demultiplexing**: the process to generate segment (sender) and give to correct sockets (receiver). **Connectionless Demux** identifies by 2-tuple (only receiver) while **Connection-Oriented Demux** identifies by 4-tuple.

Reliable Data Transfer

Recovers *bit flipping* and *package loss*.

Stop-and-wait protocols: (too slow) *Bit flipping* $\to$ `ACK`s; garbled `ACK` $\to$ resend; `ACK` + 1-bit sequence number, then duplicate `ACK` = `NAK`. *Lossy channel* $\to$ timer.

Pipelining protocols: *Go-Back-N* and *Selective Repeat*. Sequence number is modulo k.

Go-Back-N: The sender sends as long as window length $< N$. The receiver always `ACK`s the last packet in order. In sending, the timer is never stopped unless all packets are `ACK`ed. After timeout, all un`ACK`ed packets retransmitted. **No buffer**.

Selective Repeat: (for large window size and larget bandwidth-delay product) timer for every packet. Buffer and wait until all arrived.

$*$ **Notice** that both requires $k \geq 2N$ to ensure packet N won't replace the packet 0 required to retransmit.

The Transport Control Protocol

Point-to-point, *reliable*, *pipelined*, *bi-directional data flow*, *connection-oriented* and *flow controlled*. Sender simultaneously send `seq, ack` where `ack` is the byte expected to receive next. Timeout value approximated by EstRTT $+ 4 \times$ DevRTT. **Fast Retransmission**: immediately resend after three duplicate `ACK`s rather than waiting for timeout. **Flow Control**: use receive window.

Handshake: `SYN=1` in `Cli->Ser` and `Ser->Cli`. In ending, `FIN=1` by `Cli->Ser`, the server `ACK`s the `FIN` packet. After server ends transmission, `FIN=1` by `Ser->Cli`. When client receives `FIN`, reply with `ACK` and wait a long time in case `ACK` is lost (then the server tries to retransmit `FIN`).

Congestion Control: define congestion window `cwnd=1*MSS` and `ssthresh=64KB`.

Slow Start: every acknowledged segment adds `1*MSS` to `cwnd`, growing exponentially. A timeout sets `cwnd=1` and `ssthresh=cwnd/2` and enter *slow start* again. Three duplicate `ACK`s leads to *fast transmit* and enter *fast recovery*. If `cwnd=ssthresh` then enter congestion avoidance.

Congestion Avoidance: `cwnd+=(MSS/cwnd)*MSS` for each acknowledged packet. Timeout $\to$ behave like *slow start*. Triple duplicate `ACK`s $\to$ `ssthresh=cwnd/2` and `cwnd/=2`, enter fast recovery.

Fast Recovery: `cwnd+=1*MSS` for each duplicate `ACK`. `ACK`ed missing segment $\to$ *congestion avoidance*. Timeout $\to$ act like *slow start*. (Tahoe doesn't have this state, only RENO)

Fairness: TCP is fair under same `RTT` (consider growth in *congestion avoidance*).

Network Layer

Performs **routing** (control plane) and **forwarding** (data plane). Network service model: *e.g.* in-order delivery, security services.

Routers: *longest prefix matching*. **Switching fabrics**: memory, bus and crossbar.

Scheduling: without preemption (can't interrupt), tail-drop, priority-based and randomly.

Round Robin: cyclically scan and send one from each. **Weighted Fair**: each class gets weighted amount of service each cycle.

Internet Protocol: 20 byte header, *fragmentation* to adapt to different MTU links using 16-bit identifier, flags and fragment offset. IP addresses for each interface instead of host.

Get IP Address: hard-coded by system or **Dynamic Host Configuration Protocol**. *Discover* (broadcast) $\to$ *respond* $\to$ *request* $\to$ *ACK*. Maybe 1+ servers answering.

Subnets: physically reach without intervening routers. **Classless Inter-Domain Routing**: `a.b.c.d/x` (classful: `x=8/16/24` as A,B,C networks). ISPs get address blocks from Internet Corporation for Assigned Names and Numbers.

IPv6: 40 byte header. Remove checksum and fragmentation. *Tunneling* (wrapping IPv4 header outside IPv6 header) to adapt to IPv4 routers.

Routing Algorithms: *global information* (LS) vs. *decentralized information* (DV); *static* vs. *dynamic*.

Link-State: Use Dijkstra. Know all net topologies and link costs. $O(nE)$ messages (n nodes and E links). $O(n^2)$ time. Problem is oscillation.

- initiate $D(v) = c(u, v)$ if $v \in N(u)$ else ∞ and $S = \emptyset$
- while $\exists w \notin S$: add $w_0 \notin S$ with minimum $D(w_0)$ to S and update $D(v) = \min\{D(v), D(w_0) + c(w_0, v)\}$ for all v s.t. $v \notin S$ and $v \in N(w_0)$.

Distance-Vector: Use Bellman-Ford. Each node store DV of itself and neighbours.

- Each node send distance vector $\boldsymbol{D}_v = \{D_v(y) | y \in N\}$ to all neighbours.
- Each node x update according to its own information: for each $y \in N$, $D_x(y) \leftarrow \min_{v \in N(x)} \{c(x, v) + D_v(y)\}$.

$*$ Comparison:

(message complexity) LS requires $O(nE)$ messages sent if there are n nodes and E links. But DV has messages online between neighbours.

(speed of convergence) LS costs $O(n^2)$, while DV convergence time varies (*e.g.* routing loops and count-to-infinity).

(robustness) When router malfunctions, LS allows nodes to advertise incorrect costs, but DV would let errors propagate through network.

AS Routing: hierarchy because network too large and ISPs want administrative autonomy.
Def Autonomous System (domains): the networks where routing algorithms are applied.
Def Gateway Routers: The router connected to routers in other ASes.
Intra-AS Routing (Interior Gateway Protocols) - Open Shortest Path First: link state, carried over IP, reliability ensured by itself. **Two-Level Hierarchy**: *local area* and *backbones* both run OSPF only in themselves. *Local area* routers only remember direction towards backbone.
Inter-AS Routing - Border Gateway Protocol: provide *eBGP* (among neighbouring ASes) and *iBGP* (inside AS), allowing subnets to advertise itself; determining the best route to a subnet according to policy (here policy dominates performance).

Attributes: **AS-PATH** (the already passed ASes) and **NEXT-HOP** (the IP address of the router interface that begins the current AS-PATH). Prevent loops through rejecting when a router sees its own AS in AS-PATH.

Hot Ptato Routing: Minimize the cost within AS.

BGP Routing: (1) (highest priority) local preference (completely a policy thing), (2) shortest AS-PATH (that is, counting the minimal AS hops instead of router hops), (3) hot potato routing. Therefore BGP isn't selfish.

$*$ If a router A doesn't want to take traffic between B and C, it simply not advertise ABx to C. (**Dual-homed** customers of ISPs)

Software Defined Networking: logically centered control plane. Three planes: (1) *SDN-controlled switches* (data plane), (2) *SDN controller*, (3) *network-control applications* (routing/access control/...).
SDN-Controlled Switches require protocol for communicating with controller and API for table-based switch control (*e.g.* OpenFlow). Only does data-plane forwarding.
SDN-Controller maintains network state information. Uses **northbound API** to interact with control applications and **southbound API** to interact with network switches.
Network-Control Apps uses API provided by controller to implement control functions. It is unbundled so can be provided by 3rd party.

Internet Control Message Protocol: Used to communicate network-layer information, e.g. report errors, and echo requests/replies. Its messages are carried in IP datagrams.
Message format: `<Type> + <Code> +` First 8 bytes of IP datagram, *e.g.* 0+0 is ping reply while 8+0 is ping request.
Network Management: Each agent maintain a **management information base** containing statistics, while the managing server can access those data or set them.

Simple Network Management Procotol (SNMP): application layer. Works in two ways: (1) managing entity requests data from agents, (2) agent send a **trap message** to manager, informing it of a change of MIB item.

Link Layer

Framing and *encapsulating* of IP datagrams, offer *error detection* and *correction*. **Point-to-point** links or **multiple access** links, implemented in **adaptor (network interface card)**.
Error Detection: *two-dimensional bit parity* and *cycle redundancy check* (sender generates $R = \text{rem}\frac{D\times 2^r}{G}$ and receiver checks whether $\langle D, R\rangle = 0 \mod G$).

Multiple Access Protocol

Handle collision and distributed. Ideally no synchronization is needed. **Taxonomy**: channel partitioning, random access and taking turns.
Time (Frequency) Division Multiple Access: access in turn, problem: waste of idle slots.
Slotted ALOHA: assuming all frames have the same size, sending time synchronized, all nodes detect collision. When collision, node retransmits frame in each subsequent slot *w.p.* p until success. Max efficiency $\lim_{N\to\infty} Np(1-p)^{N-1} = 1/e$. Unslotted case $\Pr[\text{success}] = p(1-p)^{2(N-1)}$ with maximum $1/2e$.
Carrier Sense Multiple Access: not interrupting others. But collisions still happen due to propagation delay.
CSMA/Collision Detection: when collision detected, abort the transmission. Easy for wired LANs by measuring signal strengths, but difficult for wireless LANs because received signal strength is overwhelmed by local transmission.

Binary Exponential Backoff: After the m-th collision, NIC waits $K \times 512$ bit times where K is selected at random from $\{0, 1, ..., 2^m - 1\}$. *Efficiency*: suppose T_{prop} is the max prop delay between 2 nodes in LAN and T_{trans} is the time to transmit max-size frame, then eff $= \frac{1}{1+5T_{\text{prop}}/T_{\text{trans}}}$.

For 802.11, CD impossible, so it uses **CSMA/Collision Avoidance** instead. Consider hidden terminal problem, ACKs are needed. $*$ For sender, the frame is not transmitted until **DIFS** or **timer expiration**. If channel is busy, then start timer whose countdown time is determined by binary exponential backoff. The timer only counts down when channel is idle. If no ACK received, then backoff interval is increased and sending process is performed again. $*$ For receiver, ACK after **SIFS**.

Other ways include: **(1)** Use **request-to-send packets** (sent to routers) and **clear-to-send(client)** (sent from router to all clients) responses. **(2) Taking-Turns**: **Polling** where master node invites slave nodes to transmit in turn (but suffers from latency and fear of failure of master, used by Bluetooth), or **token passing** allowing only the token holder to send.

Medium Access Control: 48-bit address.

Address Resolution Protocol: figure out MAC address (which is used in link layer) by IP address. Each IP node maintains an ARP table, with each item recording `IPaddr; MACaddr; TTL`. TTL refers to the time after which address mapping will be forgotten (often 20min).

Ethernet: Topology: bus or star. Format is `preamble, dest.addr, source.addr, type, data, CRC` where preamble has 8 fixed bytes, used to synchronize clock rates. Type indicate network layer protocol.

Features: *connectionless*, *unreliable* (data dropped if CRC failed), uses CSMA/CD with binary backoff.

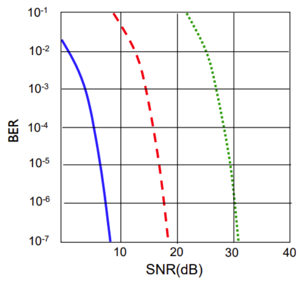
Virtual Local Area Network: switch ports grouped, each group isolated from any other, while the group each port belongs to is managed by software, not hardware. Forwarding between VLANS are completed using an extra router.
Problems arise when we need N groups and we want to connect using multiple VLANs. Solution is VLAN trunking. **Trunk Port**: used to carry frames between VLANs for all groups. To separate those frames, 802.1Q is defined. 4 extra bytes are appended to the Ethernet header.

Wireless Network

Challenges: *wireless* link and *mobility* (changing the point of attachment to network) (wireless doesn't mean mobile).
Elements: wireless hosts, base station, wireless link (to connect the former two). In *infrastructure* mode, base station connects mobiles into wired network while in *ad hoc* mode, nodes transmit to other nodes within link coverage, orgainzing there own networks.

Wireless Link Characteristics: decreased signal strength, interference from other sources, multipath propagation.

Tradeoff: SNR and BER (bit error rate): Given physical layer (fixed noise) we *increase power*; Given SNR we choose physical layer. *Rate adaption*: switch to slower physical layer to obtain lower BER.



802.11: *access point* (=base station). Offers *power management*.
Cellular Netowrk: *base station* analogous to AP. TDMA+FDMA.