What makes the Internet?

1) Hardware
- **Hosts** = end systems (Anything devices that runs network apps | pc, phone, server)
- **Communication Links** (Running network apps | radio satellites, wireless access point, cables)
- **Packet Switches** (forward chunks of data called 'packets' | switches, router)

2) Collection of services
- Search engine, games, etc

Modem -> modulator-demodulator
- last time: used to modulate telephone signals so digital info can be encoded in it
- Modulation: a process to add text/video/image/sound, any data into a carrier signal like EM wave/optical/electrical signals
- The modem 'mounts' digital data into EM waves/electrical signal (carrier signal for broadband modems) so the data can travel along the respective medium.
- new modems directly feed digital data to comp but are still needed to convert electrical signals required over long connection into some signal form that is understood by router & computers

Routers: route signals

Challenges of Internet:

1. Protocol:
- Defines set of rules: format & order on how to send messages through the internet (each layer has its own protocol)
- IETF develops these standards, these docs are called RFC

2. Network API (implements the protocol, provides services to user app using API)

3. Sharing

Internet usage is bursty

Packet switching: (cheap) share a connection with many users (24h)

Packets occupy link on demand. Better for bursty data. Store and forward transmission (cannot forward pkt b4 all 10kb data is received)

Scenario: 1Mbps link, shared among 10 users, users will use 100kb/s if active, but active only 10% of the time

$$P_{active=k} = \binom{N}{k} P_a^k \, 0.9^{N-k}$$

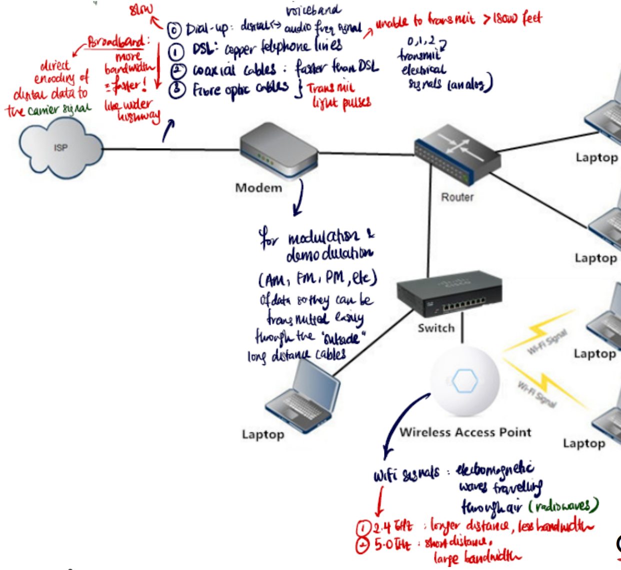
$$P_{active=10} = \binom{10}{10} 0.1^{10}\, 0.9^{10-10} = 0.1^{10}$$

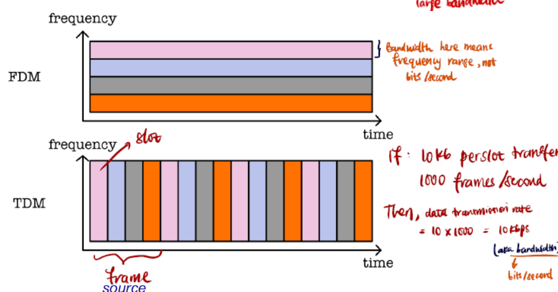
Circuit switching: (stable connectn, cont streamline usage) a fixed, dedicated fraction of the link for each user. (on-demand)

Methods: FDM (radio), TDM (tele) {data transmission rate (kbps) = kb/slot x frames/s}

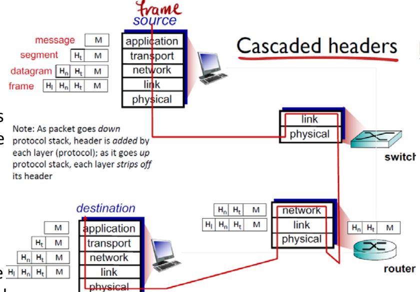
4. Complex Interaction -> 5 network layer model

Layers: each layer is like a module, it implements a service: – via its own internal-layer actions
– relying on services provided by layer below
– in turn provides services to layer above
• Layer i uses service of layer i-1 to provide service for i+1





Cascaded headers

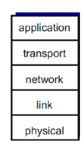


Note: As packet goes *down* protocol stack, header is *added* by each layer (protocol); as it goes *up* protocol stack, each layer *strips off* its header

NB: With layering, header vs. payload is now *relative* to the layer; e.g., transport layer's header is network layer's payload

Internet protocol stack

- *application:* supporting network applications
  – e.g., FTP, SMTP (email), HTTP (web)
- *transport:* process-process data transfer
  – e.g., TCP (reliable), UDP (not reliable)
- *network:* routing of datagrams from source to destination
  – e.g., IP, routing protocols
- *link:* data transfer between neighboring network elements
  – Ethernet, 802.11 (WiFi), PPP
- *physical:* bits "on the wire"



Why is Layering Important?
- Cons of complex interactions between so many different modules (network components): **N^2** possible interactions, difficult to debug & might lead to emergent behaviour (eg. priority inversion)

Emergent behaviour: often undesirable, a condition that arises bcus of interaction among pans, but is not possessed by the individual pans.

possible interactions with N layers is **N-1**.

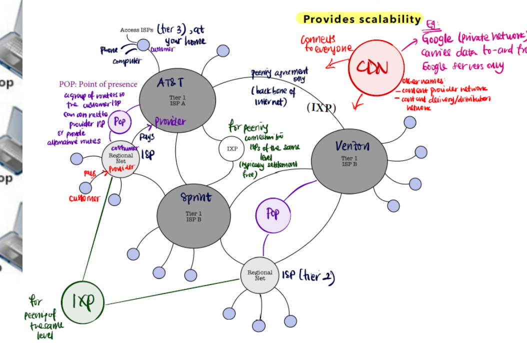
Challenge: 5. has to be scalable
- Tier 1, 2, 3 ISPs & CDNs 'layers' the connections, not O(n^2) connections.
- **Internet hierarchy**: hierarchical structure supports scalability.

Connect each access ISP to global transit ISP. Interconnect global ISPs via Internet Exchange Point (IXP).

Regional network: to connect access nets to ISPs

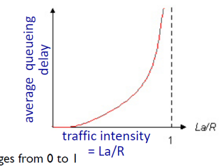
Content Provider Networks (eg Google) may run their own private network, to bring services, content close to end users (connects its data centers to internet, bypass tier1 & regional ISP)

4. INTERNET HIERARCHY



Queueing delay (revisited)

- *R:* link bandwidth (bps)
- *L:* packet length (bits)
- a: average packet arrival rate

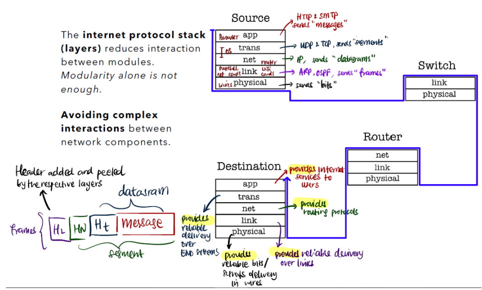


- *La/R* is also average link *utilization*, ranges from 0 to 1
- *La/R* ~ 0: avg. queueing delay small
- *La/R* -> 1: avg. queueing delay large
- *La/R* > 1: more "work" arriving than can be serviced, average delay infinite!

Note that queueing delay is *convex* increasing function of utilization

This assumes random bursty packet arrivals (natural for internet); traffic arrivals in special cases may have different results

You should not be too greedy and try to use every bit of the link capacity (e.g., 0.95 utilization) – why?



The **internet protocol stack (layers)** reduces interaction between modules. *Modularity alone is not enough.*

Avoiding complex interactions between network components.

From Packet Switching -> Delay/Congestion

Packets queue in router buffers (when pkt arrival rate to link (temporarily) > output link capacity.

1. Processing Delay (duration < ms)

Needs time to examine packet header for: 1) check for bit errors (checksum); 2) determine output link by dest IP addr (bounded) because packet size is fixed. (smaller the packet, better it is because if 1 bit dropped, delete only a smol pkt)

2. Queuing Delay (duration dep on traffic, unbounded)

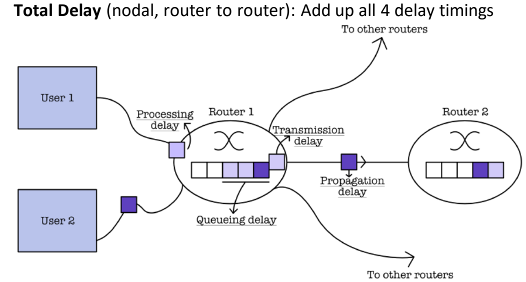
Packet needs to wait to get to the front of the queue to reach output link due to congestion (input rate>output rate)
- Traffic Intensity = La/R (L=pkt length(bits), a= avg pkt arr rate, R=link bandwidth (bits/s))

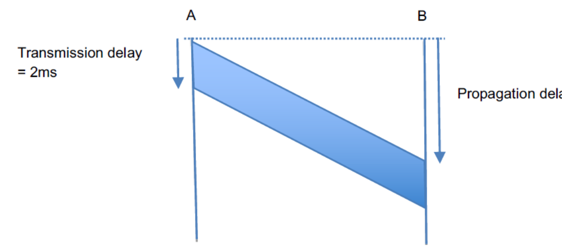
Avg Link Utilisation - [0,1], ~0: delay smol, -> 1: delay large, >1: delay infinite

3. Transmission Delay (dep on link bandwidth, bounded)
- delay = L/R
- need time to push whole packet bits from router end to the link (cable)

4. Propagation Delay (dep on link length, bounded)
- delay = d/S (d=link length (m), S=propagation speed of bits on wire (app 2x10^8m/s))
- Need time for bits to propagate from the output end of Router 1 to input end of Router 2

Total Delay (nodal, router to router): Add up all 4 delay timings





1. Data sent by network applications will eventually appear as link layer packets. In a bulk transfer (i.e., lots of application data sent quickly), a largest possible packet size should be used for efficiency. Give a reason why. **When the total packet size is larger, the percentage overhead of packet headers (whose sizes are fixed) is smaller.**
2. When network traffic is bursty, _packet_ switching can significantly increase the utilization of communication bandwidth compared with _circuit_ switching
3. Consider an end-to-end network path from a source to a destination. The path consists of two hops, and the packet loss rate of each hop is 10%.
(a) What is the packet loss rate of the end-to-end path? **19%**
(b) Assume that a lost packet is retransmitted until it is received by the destination. What is the probability that a packet will be retransmitted exactly *N* times before it is received? **0.19*N*×0.81**
4. 1000 bytes of data is sent as quickly as possible from A to B over a link of propagation delay 6ms and bandwidth 4 Mbits/s. Ignore nodal processing and queueing delays. (a) Draw a spacetime diagram of the whole transfer, from when the first bit is sent by A to when the last bit is received by B. (b) Calculate the transmission delay and indicate it in the diagram. (c) Indicate the propagation delay in the diagram.

① End host systems connected by:
↳ ① comm links
② packet switches

network (network layer) routers and switches (link layer) allow packets to travels through "routes" or paths

transmit packets: packages of info at different rates (bits/sec)

② ISP: a network of packet switches & comm. links

residential ISP provides broadband / fibre access through cable modem, wireless access, dial-up, etc.

③ for systems to communicate, we need protocols. End systems usually run: (TCP/IP)
① Internet Protocol (IP)
② Trans. control protocol (TCP)
③ HTTP (web)
④ SMTP (email)

specifies format of packets that are sent and rcv. among routers and end systems

④ IETF develops these standards, these documents are called RFC

The Internet P.1

(as hardware & software)

4) Internet as infrastructure that provides services to applications: P2P, web surfing, social networks (chat), games, filesharing, TV/streaming, remote login, etc.

possible by running Internet API on your system

↓

defines set of rules so that your code infrastructure can send data successfully throughout the internet

you can write any applications that can run on your end systems using Java or C or Python, etc, but we need a set of rules so that these hosts can communicate properly: use internet APIs that implements specific Protocols

↓

defines ORDER and FORMAT of packets sent and recieved

The Internet P2:

(as infrastructure providing services)

5) The network edge & core → ① packet switches

↳ break long messages into smaller chunks (packets)

↳ time taken: $\frac{L}{R}$ → packet length (b), → trans. rate (b/s)

↳ how? store & forward transmission

$d_{AB} = \frac{NL}{R}$, if N routers in between A & B

switches must receive the ENTIRE packet bits before forwarding to the next destination

causes queue delay

may have packet loss

edge: comps, phones, printers, TVs, consoles, etc

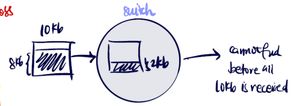
each router has this → ISP's job to solve

② Forwarding table & routing protocols

↳ each packet has IP addr

↳ router see IP, look at table & forward to adjacent router

↳ who sets the table: IP routing protocols used to automatically set forwarding table in routers



6) Circuit and Packet Switching

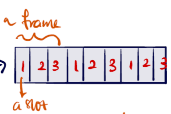
reserve all hardware (resources) b/w end to end path, e.g. telephone.
④ guaranteed transmission rate

The internet is this method, no guarantee over transmission rate. Resources shared among many hosts.

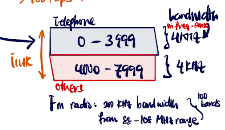
2 flavours of circuit switching:
① TDM: reserve over a "time" period
② FDM: reserve over a "frequency" domain

↓

telephone: 4kHz bandwidth, ~300 – 3300 Hz
radio: 88MHz – 108MHz



trans rate: x frames/s
bandwidth in data transfer sense
b bits/slot
eg: 10 bits/slot & 1000 frames/s
↳ 100 kbps transmission rate



FM radio: 20 kHz bandwidth, 100 bands from 88 – 108 MHz range

Circuit switching vs packet switching (current trend for Internet usage)

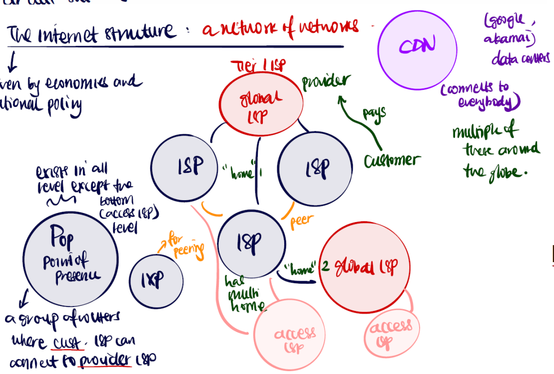
✓ good for real time services: video call, telephone, to avoid unexpected delays
✗ limited number of users

✓ more efficient, don't have to "reserve" the connection, eg: chat
✓ idle resources can be used by other people
✓ suitable for Internet because our usage pattern is BURSTY
✓ unlimited number of users
✗ delays can go to ∞ (like traffic jam)

BTW, Netflix is fast because of CDNs, not circuit switching!

7) The internet structure: a network of networks

driven by economics and national policy



a group of routers where cust. ISP can connect to provider ISP

Who is Google? → a content delivery network → delivers contents (copied) directly, no need to ask server

↳ data centers interconnected over private TCP/IP network
↳ separate from the public internet
↳ only carries traffic to and from Google servers
↳ "bypass" the upper tier internet, can connect to access ISP directly. (free)
↳ also connect to tier 1, 2 ISPs (paid)

basically, Google is its own network: fast

Netflix used CDNs, now they created their own
① Limelight ↳ openconnect optimized for large files.
② Akamai
③ Cloud Front (Amazon)
④ Cloud Fare
∴ many businesses use CDNs.

8) Layered Network Architecture allows encapsulation

↳ complex system, hence needs this architecture
④ analogous to plane boarding protocol → a complicated process with layered checkpoints

↳ reduce complexity into O(n) since each layer only provides services for the layer ABOVE it.

↳ each network protocol belongs to only ONE layer

sends "message": bits of info software in end systems

protocol stack

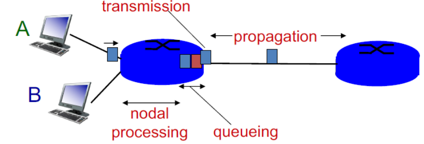
Application:	SMTP & HTTP, FTP
Transport:	TCP & UDP (reliable delivery over end systems) → sends "segment"
Network	IP, routing protocol → sends "datagram"
link:	delivers to next node along route → sends "frames"
physical:	cables → sends "bits"

in OS

mix of software & hardware

network interface cards: WiFi/ethernet card

Protocol: reliable delivery over LINKS
ARP, OSPF

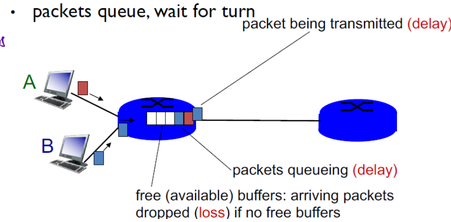


Single "hop" nodal delay (i.e., delay from one node to *immediate* next one):

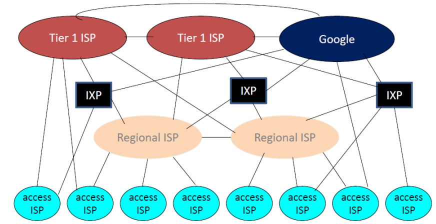
$$d_{nodal} = d_{proc} + d_{queue} + d_{trans} + d_{prop}$$

packets *queue* in router buffers

- packet arrival rate to link (temporarily) exceeds output link capacity
- packets queue, wait for turn



Internet structure: network of networks



- at center: small # of well-connected large networks
  - "tier-1" commercial ISPs (e.g., Level 3, Sprint, AT&T, NTT), national & international coverage (tier 1 represents the *largest* global ISPs)
  - content provider network (e.g. Google): private network that connects it data centers to Internet, often bypassing tier-1, regional ISPs

Tier-1 ISP: e.g., Sprint