8bit = 1byte

Intro (lec 1)

Network principle
- → Packet switch
- → Circuit switch (phone)

Net Structure → Access Net (WAN, Lan, ...) Gbps
↓ Core Net (national, global, ...) Tbps/Pbps

Net Media
- → Acoustic (sound wave)
- → EM (light)

↓ Guided (wire)
- → Copper
- → Coaxial
- → Fiber

→ Broadcast (wireless)
- → Radio freq
- → Satellite
- → IR

QoS (Service): end → end delay (total)
Packet loss rate
Bandwidth / Throughput % / Achievable data rate

QoE (Experience): quality of speech / image, ...

Delay
- → Propagation delay $\frac{\text{distance}}{\text{medium speed}}$
- → Transmit $\text{bit length} / \text{bandwidth}$
- → Process: depend
- → Queue: random to be process

Pattern: Unicast / Broadcast / Multicast
Client-Server / Peer to peer (oposite)

Packet switch can send/receive packet parallel, VBR, Circuit switch can't

Socket program (lec 2)

Transport layer

UDP: Connectionless, Unack, Unrealiable, Unorder (datagram)

TCP: oposite, block not busy loop (stream)

Socket is post office, bound to a port and same buffer

TCP: Create socket (random port) → connect → read/write → close

TCP server: socket → bind (choose port) → listen → accept → ...

UDP: socket → connect → read/write → close

UDP server: socket → bind → read/write → close

Layers (lec 3)

- Application
- Representation (translate to packet)
- Session

} end to end

- Transport (TCP/UDP)
- Network (IP) } hop by hop
- Link (Ethernet ...)
- Physical (EM, IR, modulate, ...)

} single hop

Service
- → Confirm (after processed)
- → Unconfirm
- → Confirm delivery (received)

Multiplex: Combine fragments to 1 packet to save overhead

Split: break to smaller part (reduce bit err)
Need sequence number

Physical layer (lec 4)

Modulator: convert bit to analog, add extra bits to improve correct transmit. Source coding compress data before modulate

Attenuation: signal weaken after travel, harder to decode

$\eta = P_{tx}/P_{rx}$ $\quad \eta_{DB} = 10 \log_{10} \eta$ $\quad \eta = 10^{\eta_{DB}/10}$

rx: receiver

Passband transmit: encode data around f_c (center freq)

Amplitude Shift: A↗ or ↙ represent 1 or 0
$s_i(t) = A_i \cos(f_c \cdot t)$

Freq Shift: freq ↗ (closer) or ↙ (1 or 0)
$s_i(t) = A \cos[(f_c + f_i) \cdot t]$

Phase Shift: phase $-\pi$ represent change (0 or 1)
Starting phase 0 or π (0 or 1)
$s_i = A \cos(f_c \cdot t + \phi)$

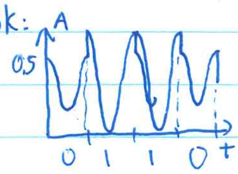
Baseband: NRZ & Manchester ($f_c = 0$)

volt / time

NRZ: high (+1) for 1 and low (0) (-1) for 0

Manchester: ⌊ for 1, ⌋ for 0 (in 1 unit time)

ASK: $f_c = 2\pi$, $A_0 = 0.5$, $A_1 = 1$



FSK: $f_0 = f_c + f_0$, $f_1 = f_c + f_1$

PSK: f_c const

MAC → Orthogonal MAC (OMAC)
MAC → RAP

O M A C / R A P

LAN (Lec 5) (Bus/Star topology)

(*) MAC is ~~a named~~ name of collection of protocol (no colision)

→ FDMA ($T_{total} = T_{transmit}$)
→ TDMA ($T_{total} = T_{Access}$ (wait) + $T_{transmit}$)

FDMA like phone, bandwith B (speed) is split evenly for N channel (N possible end transmiter)

TDMA give each transmitter B speed for a time frame, $T_{access} = \frac{T_{super\ frame}}{2}$, $T_{transmit} = \frac{1}{N} * frame$

(each transmit has $1/N$ s per frame)

=> Both FDMA, TDMA have reserve resource for each host, good for CBR, no colision

(*) Random Access Protocols (RAP)

Opposite of OMAC, accept colision, VBR, reusable resources

- ALOHA: receive → transmit imediately, ack timer start → receive ack from dest if success
  → if not → back off (random) then repeat for N time after N fails, drop frame

Packet queued, each packet send at random backoff

+ Advantage: $T_{access} = 0$, full bandwidth, low collision for small net, simple to implement
+ Dis: Vulnerable = 2 frame length, can't distinguish collision vs ~~dest~~ channel error

- CSMA: Carrier sense multiple access, listen before talk to sense busy/idle medium

When propagation (d) > transmit delay (L), sender might have completed sending when notice busy medium

+ Non-p: sense → idle → transmit
  → busy → random backoff (NORM dist)
  Collision: back off, start over
+ p-p: sense → busy → wait idle
  → idle → Yes/No for p chance start transmit, 1-p chance wait extra time slot
  Collision: start over
+ 1-p: sense → idle → send
  → busy → wait
  collision: collision resolution procedure
  → Abort transmit
  → Send jam
  → Start procedure (tree, backoff, ...)

$E = {1-p}/p$

$p(k) = p^{*}(1-p)^k$

$p(k) = p(1-p)^{k-1}$

$E = 1/p$

Ethernet Address (48 bit) 00:00:00:00:00:00

$2^{min(10, coll\_no)} - 1$: backoff slot

× slot time = actual backoff T

Hub: centralise repeater, amplify wave to broadcast to other host. Star topology, each host has 1 line for send/receive

Bridge: Connect LAN, have table of forward to know which host can be reached from port X.

Switch: full duplex, queue packet to same dest, forward frame on correct port. Hub is broadcast, switch have N parallel transmit

IPv4 (lec 6)

192.168.40.64/28 has $2^{32-28} - 2$ = 14 address

Network as a whole: 192.168.40.0010 0000 (host addr here)

Broadcast addr of net: 192.168.40.80 [0010 1111]

IP {}.{}.{}.{} = (addr & 0xFF000000) >> 24
(addr & 0x00FF0000) >> 16 ...

ARP: get MAC addr from IP (avoid full cache) → request/response accepted by station with match IP only

ICMP: allow host inform sender of unusual behavior

Attenuation calc:

0.2dB = 1.04 (RATIO)

After K km, $1.04 \times 1.04 \times \ldots \times 1.04 = 1.04^k$

$0.2 + 0.2 + 0.2 + \ldots + 0.2 = k \cdot 0.2$

$1.04^k = 0.2k$ (fact)
normal decibel

IP fragmentation can be done in intermediate, but IP reassembly can't (packet not always take same route, no router have all fragment)

At least 2 bit error: $1 - (1-p)^L - (L \times p \times (1-p)^{L-1})$

$\frac{2^c - 1}{2} + \ldots$ c: no collision

$P/(2-p)$ collision, $1-P/p$

Transport Layer

↳ ARQ → Rdt (Reliable data transfer) → Stop & wait

→ Go back N } Pipelining

→ Selective Repeat

rdt 1.0 = reliable channel

rdt 2.0: Use checksum/CRC, response ACK/NAK for channel with bit error

rdt 2.1: Alternate seq no, in case corrupt ACK/NAK

rdt 2.2: NAK free, use duplicate (ACK, seq) to replace NAK

rdt 3.0: Use for lossy channel with bit error

Solution: retransmit on time out

Cause: Data/ACK lost, Data/ACK delay

* still use alternate bit seq no.

This is stop-and-wait, not async (to send data parallel). Utilisation = $\frac{L/R}{RTT + L/R}$

Packet Size (L)   RTT (Round trip time)

Transmit Rate (R)

- GobackN: Use Seq no $0 \to 2^k - 1$, then repeat 0 on window size N (at most N packet sent and un ACK'd)

Use cumulative ACK: an ACK with seq # n means all packet upto & include nth have been correctly received

Receiver expect in order packet, will discard & ACK last in order packet, sender must resend all not-in order packet

- Selective Repeat/Reject: if seq # in accept window, buffer packet and send selective ACK.

When buffered packets in sequence (expected in order), deliver to upper layer, shift window.

Sender receive selective ACK, if ACKed packet in-order, shift window. if not, resend base packet only (if the base packet ACK'd, window has moved)

Window size ≤ 1/2 size of seq #, to avoid 2 duplicate seq# in a window

- TCP flow control: calc spare room in buffer → RcvWindow and tell sender → prevent sender send too fast, overflow at receiver

- TCP Congestion control: avoid overflow at router caused by congestion in the network.

Cause: Share link, finite output link buffer,

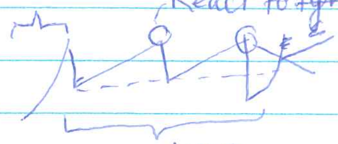
↳ Too many source sending too much data, too fast for network to handle (lost packet / long delay).

LastByteSent - lastByteACK ≤ min{CongWin, RcvWin}

① Control: AIMD (additive ↗, multiplicative ↙)

② → Slow start

React to timeout / 3 Dup ACK



Reaction to timeout

AIMD

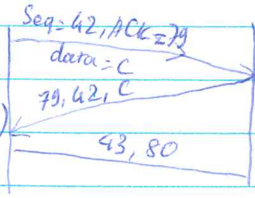
Time out is more severe than 3 Dup ACK, CongWin drop to 1 instead half, rise exponential instead linear, until half, then linear as 3 Dup ACK

App	App layer Proto	Transport Layer pro.
Email	SMTP	TCP
Web	HTTP	TCP
Routing	BGP	TCP
Routing	RIP	UDP
	DNS	UDP

- UDP is connectionless, simple, small header, no congestion control
  - can be used to implement rdt.
  - can send as fast as posible

- TCP: One receiver, one sender, reliable, in-order data stream full duplex, pipelined, handshaking, flow control

Use 3 way handshake, ACK = seq# of next byte expected, cumulative



- Net App need: rdt, bandwidth, timing (Client-server or P2P architect)

HTTP non persistent has a longer delay for each object, waste buffer & variable

HTTP stateless since it work without knowing past request

Cookie: L16 p23

Webpage Cache: L16 p26-27

Quiz 7-8: Ipv4 Networking

OSI layer

- App: How to share data A → B
- Transport: Reliable / Easy transport
- Network: Find route A → B (IP

Message: App
↳ Segment: Transport
  ↳ Datagram: Network
    ↳ Frame: Link

- Network layer: include IP protocol, ICMP (error), routing protocol (Bellman-Ford, Dijsktra ...), forwarding table ...
- Bellman-Ford algo:
  - Decentralise, no node has full network info
  - Distance Vector (DV), store and share direction to neighbor only, good for update big net.
- Dijsktra algo:
  - Centralise, each node know full network
  - use link state (value: distance, traffic ...), good for small net as flooding is fast

Autonomous System (AS)

- Stub AS: only have 1 connection to other AS
- Multihome: connect to multiple AS, but traffic not pass
- Transit: multihome, allow network pass through

- Routing decide path to take, populate forwarding table. Foward send the packet, router choose which link to send per packet depend on forwarding table.

Routing Algo ≠ Routing Protocol

- A protocol use an algo into networking context (unideal environment)
  - Bellman-Ford: RIP, BGP
  - Dijsktra: OSFP
- Routing class
  - Static / Dynamic
  - Global / Decentralise
  - Load sensitive (change cost on congestion) or insensitive (RIP, BGP, OSPF)

Network layer, build on IP

- OSPF (Intra-inside a workgroup): Link state, has hierachical in large domain, multiple same cost path
- RIP (Intra): Max 15 hops, application layer, build on UDP (for advertisement), exchange advertise every 30sec
- BGP (inter): customer don't advertise B to C, B only advertise to customer (neighbors)
  ↳ kept silent / no free riding

(different ip)

- NAT: Translate LAN ip to WAN ip (share same ip of router, different port). Router has NAT translation table

a.b.c.d, 5001	10.10.10.1, ~~3345~~ xxxx
a.b.c.d, 5003	10.10.10.3, yyyy

- Port no. at Layer 4, so router with net attached need to process upto Lv4.

IPv6:

Ver (4)	Traffic (8)	flow lable (20)	
Len		next hdr	hop limit
Source Addr (128 bit)			
Dest. Addr (128 bit			
Data			

Transition Ipv4 → Ipv6
- Dual stack (with convert router)
  ↳ lost some data
- Tunnelling: put v6 packet inside v4

Hierachical Routing: group routers into AS, each run by a company, run same protocol, ...
Adopted by the Internet (scale issue fix)

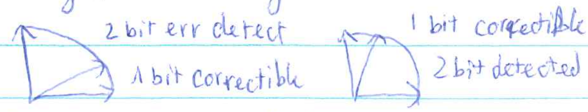
- Parity check: add 1 parity bit to make data has odd / even number of 0 or 1
- Check sum: Sum all bit (number) and write, receiver check writen sum vs sum all data received
  - Partial sum = Sum + 1 (if carry over)
  - Check sum = NOT (Partial sum)

Internet check sum is weaker than CRC.

- Needed at IP, UDP, TCP b/c some link-layer don't do err-detection, bit-error happen everywhere, end-to-end check is always needed.
- CRC: $\underbrace{\overset{d}{\boxed{D}}\overset{r}{\boxed{FSC}}}_{T = (d+r)\ bit}$ $F = D \times 2^r / G$ (1 bit less than G) $D = T/G$
- FEC: Error correction has limit (min-Haming distance) of which it can't fix

if FEC has 5 bit to encode 4 data word, there are $2^5 - 4$ invalid code work.

Haming dist: How many bit error between 2 code work



The machine assume the case with fewest error (1 bit) and fix it, eventhough user could have sent the other