Presentation Layer

- Aware of syntax / semantics (at least partially) of service requests from application layer
  - Lower Layer handle bit, bytes, messages, even files, but are, in general, unaware of their semantics
  - Application layer processes use
    - Semantically meaningful data / info - diff. datatypes, both primitive ~~or~~ and structured
    - Diff representation codes - ASCII, EBCDIC, UNICODE, etc
    - Diff. implementations of above in diff hosts

- Pro

- Provides conversion between varied semanticall

- Provides common services / f^{ns} needed by app^{l} processes (data compr., encry., auth. etc) - avoids replication at app^{l} level
  - Actual services depend on app^{l} process requirements

Application layer

- Implements diff protocols or services that are widely used by almost all appl. prog.

  - TELNET - creates 'network virtual terminal' so that a peer process (app prog) on one host can log on to ~~another~~ a virtual terminal on another host (peer appl. process) in a transparent manner through the network
  - HTTP and HTTPS for transfer of hypertext documents between diff applications on diff hosts
  - SMTP - basic protocol for all kinds of email, bulletin boards, news services, social networking sites
  - FTP - for all kinds of file transfer between peer applications
  - DNS - For mapping host names onto their network adr & vice versa; an application in a particular host can use services provided by another application running in another host by just using its symbolic name which does not change even if the host is relocated / replaced by another one with diff network address

- A new network architecture was developed to address the problem - the TCP/IP model (named after its two popular protocols)

- Primary design goals for the TCP/IP
  - Must retrofit to the existing (D)ARPANET
  - Seamlessly interconnect all kinds of heterogenous n/w using diff comm. technologies

The TCP/IP reference model

→ Originally developed for the already operational WAN-(D)ARPANET to address certain problems

⇒ (D)ARPANET : initially connected several US universities, research labs and defence installations using leased lines & telephone lines

⇒ Advent of wireless / satellite / optic fibre networks created problems - existing (D)ARPANET protocols failed to inter-connect with these new technology networks

~~⇒ A new network architecture was developed to address this problem by another app, running in another host by just using its symbolic name which does not change even if the this host is relocated / replaced by another or with a diff network address~~

⇒ A new network architecture was developed to address this problem: the TCP/IP model (named after its two popular protocols)

⇒ Primary design goals for the TCP/IP

- Must retrofit to the existing (D)ARPANET
- Seamlessly interconnect all kinds of heterogenous networks using diff. comm. technologies
- Provide high reliability / robustness / fault tolerance (since US-defense is involved) - if source and dest hosts are alive, comm. should continue in-spite of major faults
- Have a very flexible architecture
  - → Must support all kinds of apps - ranging from simple file transfer between university students / faculty to real time audio/video/satellite imaging, tranfer of defense classified data

→ Must be naturally compatible with all kinds of n/w communication technology - existing, fully developed / emerging / partially developed or experimental / even anticipated future ones

→ A four layered model

Fig 1-21, Pg 43

→ The internet layer

- Design goals (reliable, robust, ~~fault~~ fault-tolerant) dictated a packet switched store and forward n/w (literally an inter-network) offering connection-less reliable (acknowledged) services
- Functionality closest to OSI network layer
- Primary responsibilities
  - Route IP packets from source to dest. host with very high reliability
  - Leave handling of out of sequence packet arrival at dest. to higher layer
  - Implement some congestion control mechanism

⇒ The transport layer

- Implements end-to-end connection between peer entities on source and dest hosts
- ~~Functionality~~ Functionally closest to OSI transport layer
- Two end-to-end protocols are defined for this layer
  - Transmission Control Protocol – TCP
    - A reliable connection oriented protocol for error free delivery of byte stream
    - At source host 'byte-stream' is broken into packets (IP packets possibly) and passed on to internet layer

→ The Host-to-Network layer

- TCP/IP model provides no details here except that a host must connect to the network using some protocol to transfer IP packets to or from internet layer
- No protocol defined for this; varies from host to host and n/w to n/w

fig 1-22

→ Comparison between OSI and TCP/IP models

→ Similarities - many

- Layered / hierarchical model
- Protocol stack
- Service primitives and layer interfaces

→ Differences - many

- TCP/IP designed to retrofit already operational networks ~~software~~ and protocols
  - Excellent fit / match with existing protocols, networks
  - ~~Not~~ Not general / flexible enough for emerging future technologies / protocols
- OSI model developed to provide more general but precise standards for current, emerging & future technologies
- OSI model provides all kinds of protocols at the n/w layer (conn. less, conn. oriented, reliable, unreliable) but only connection oriented reliable protocol at the transport layer (where it actually counts)
- TCP/IP model provides only conn. less IP protocol at the internet layer, but provides for a choice of protocols at the transport layer (reliable connection oriented protocol TCP or an unreliable connectionless one - UDP)
  - This was apparently better for existing networks since application layer users could see the transport layer protocols available to them

Shortcomings of the OSI Model

many, but mostly for non technical reasons

→ Bad timing
→ Bad implementation
→ Bad technology
→ Bad politics

Shortcomings of the TCP/IP model

→ Specific service, interface and protocol as distinct concepts but does not precisely define the distinction - something that OSI does very carefully

→ Not general enough - not really suitable for describing protocols protocols other than TCP/IP
⇒ Virtually impossible to describe bluetooth / NFC protocols

→ The host-to-network layer is not a layer at all. It's an interface (between n/w & DL layer) - a good model must not loosely refer to a layer interface as a layer

→ Does not distinguish between or even mention physical or data link layers
⇒ Very imp. functions performed by both & at conceptually diff levels of abstraction
⇒ Must be included in separate layers, which OSI correctly does

→ IP & TCP well designed & implemented, but many other protocols were adhoc and badly implemented
⇒ free distribution caused widespread use, hence hard to replace
- TELNET designed for 10 cps mechanical teletype writers with no provision for GUI or interactive inputs but still widely used

Final Summary

→ ISO OSI
⇒ Model (less session and presentation layers) is exceptionally useful for formal discussions on and theoretical treatment of computer networks
⇒ OSI protocols however, have not become popular

→ TCP/IP
⇒ Model per se, is virtually non ~~existent~~ existent
⇒ Protocols, however, are immensely popular and extensively used

→ Working Model

⇒ Discussions and N/W description uses a hybrid 5-layer model

⇒ Protocols ~~primarily~~ actually used are the ones from TCP/IP model

fig 1-24

Medium Access Control (MAC) Sublayer

→ MAC sublayer is the lower one of two sublayers comprising the data-link layer

→ Performs arbitration, collision detection/avoidance and determines which host gets access to multiaccess channel

→ Important for LANs such as ethernet & for wireless satellite networks

→ Static Channel allocation

⇒ freq. division multiplexing (FDM)

- Simple and efficient for fixed no. of users N and uniform traffic (i.e. not bursty)
- Not feasible for large and varying N (random log-in & log out)
  - Some may be denied access for lack of bandwidth
  - Some may not use allocated bandwidth effectively
- Real life N/W traffic is bursty (1000:1 is quite common) FDM channels will remain underutilized most of the time

⇒ Time division multiplexing (TDM)

- Same disadvantages as FDM for varying no. of users and bursty N/W traffic

→ TDM or FDM not a good option – a simple queuing analysis:

⇒ Let mean time delay = T

channel capacity = C bits/sec

Arrival rate (Poisson's distribution) = λ frames/sec

(mean frame rate)
mean service rate (exponential probability density) = $1/\mu$ bits/sec

Then from queuing theory, for poisson arrival (λ frames/sec) and exponential service rate $\left[C/(1/\mu) \right] = \mu C$ frames/sec

$$T = \frac{1}{\mu C - \lambda}$$

⇒ Let channel be divided into N subchannels (using TDM or FDM)

Subchannel capacity = C/N
Mean arrival rate per channel = λ/N

Then T_{FDM} (or T_{TDM}) is given by

$$T_{FDM} = \frac{1}{\mu(C/N) - \lambda/N} = \frac{N}{\mu C - \lambda} = NT$$

⇒ Thus, delay for each FDM (or TDM) channel is N times the original channel, although mean arrival rate into each FDM (or TDM) is $1/N$ of the original arrival rate

⇒ Static channel allocation does not work well (even when N is assumed constant and traffic)
??

~~Compul~~
Computer Networks

10/2/15

MAC sublayer

$$P_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}$$

↑ probability of n arrivals in time 't'

→ Dynamic channel allocation

- The model - 5 key assumptions
  - N independent stations / terminals
    - Each station generates frames for tx^m (one at a time)
    - Probability of generation of a frame on a time interval Δt is $\lambda \Delta t$ where λ is the frame arrival rate
    - Once frame is generated, station/process gets blocked and resumes only after successful tx^m of a frame
  - Single communication channel
    - All stations can transmit on it / receive from it
    - All stations considered equivalent
    - Stations may be assigned diff. priorities by a particular protocol
  - Collisions can, and do occur
    - Two frames tx^m simultaneously or tx^m of second started before end of first (tx^m overlap & garbled signal) causes a collision
    - All stations can detect collisions (directly / indirectly)
    - Collided frames must be fully retransmitted (full frame retx^m)
    - No errors other than those caused by collision considered
  - Any one of the following two holds true
    - Continuous time
      - Frame tx^m can start at an arbitrary point of time
      - No centralised division of time into discrete slots

- Slotted time
  - Time divided into discrete slots (all of same duration, normally 1 frame tx^{m} at a time)
  - Frame tx^{m} always begins at the start of a slot
  - A time slot may contain 0, 1 or more frames
    - corresponding to idle slot, successful tx^{m} or a collision

- Any one of the following two holds true
  - Carrier sense
    - Stations always checks if channel is in use before initiating frame tx^{m}
    - No st^{n} starts tx^{m} on an already busy channel
    - If channel is ~~by~~ busy, st^{n} waits for it to become free, checks and then initiates frame tx^{m} procedure
  - No carrier sense
    - Stations cannot sense channel condition before frame tx^{m}
    - Stations start tx^{m} a frame as and when ~~a~~ the frame becomes available
    - Stations can detect a collision after some delay in case tx^{m} is unsuccessful (indirect - no ack received)

→ Some observations on key assumptions

- Stations generate frames independently (of each other) and at ~~an~~ a const. avg. rate
  - One frame generating process per st^{n} assumed
  - More sophisticated models support multiprogrammed st^{n}s
  - A process (not entire st^{n}) blocks after frame tx^{m}, another process in same st^{n} can generate a frame now
- Absolutely single channel - no other comm. means, even for control or status information
- Not all LANs allow collision
  - Some use tokens to co-ordinate tx^{m} (Token ring / bus)

- Tokens are created, allocated, deallocated & destroyed on a dynamic basis

- Some networks use continuous time, while others use slotted time, but the same network cannot use both

- Carrier sense is easily implemented on Ethernet & wired LANs
  - May not be easy to implement on wireless and satellite networks (a stn may not be within range of all others)

- Wired carrier sense N/ws can & do abort frame tx^m immediately on sensing collision (saves precious bandwidth)

- Radio and satellite networks continue to transmit (at least for some time) even after start of collision

Multiple Access Protocols (w/o carrier sense)

→ ALOHA - Developed at univ. of Hawaii for ground based radio networks, but equally applicable to systems with multiple users competing for single shared channel

→ Pure ALOHA
- A contention protocol - no channel / carrier sense
- Continuous time - a st^n tx^{ms} as soon as frame is ready
- Fixed length frame
- Collision detected by
  - Feedback property of broadcasting
    - Sending st^n on wired LAN listens to channel while tx^m; feedback almost instantaneous
    - Sending station on radio / wireless n/w listens to transmission from other fellow stations within radio range

P.T.O

- Sending stations on sat. network listens to uplinks from other ground stations within range or waits for sat. downlink (270 ms approx) to hear from gnd. st^{n}s outside range

- Non-receipt of ACK frame from receiving st^n
  - if acknowledgement is not received within timeout, collision is assumed; sending station waits for random time interval (to avoid lock step collisions) and retx^m frames)

- Collision rate, as expected is quite high

- <u>Efficiency of Pure ALOHA</u>

→ The model

- Stations in one of two possible modes
  - Tx^m a fixed length frame
  - Blocked, waiting for a successful tx^m or receipt of ack./reply frame
- Time to transmit a frame : frame time t $\left(\frac{\text{frame length}}{\text{channel b/w}}\right)$
- Infinite number of st^{n}s generate frames with Poisson distribution ≠ mean rate of N frames / frame time t
  - Infinite number ensure P. distribution in spite of st^{n}s getting blocked (wait mode)
  - $N > 1$ ⇒ more than 1 frame generated per frame time ⇒ collision
  - For reasonable throughput : $0 < N < 1$

- Collisions trigger frame tx^m
  - Let prob. of k tx^m attempts / frame time t (new & old combined) have Poisson distribution with mean rate of G frames / frame time
  - $G \geq N$: Light load → $G \approx N$, few or no coll./retx^m
    : heavy load → $G > N$, many " "

- Probability of successful frame tx^m is P_0 (no-collision)
- Throughput S - offered load (total gen frames / fr. time) × prob. of successful tx^m = GP_0

Pg 264, fig 2

- No collision for a frame if other frames not tx^{m} within one frame time of its ends (both before & after)
- Condition for collision free frame tx^{m}:
  - St^{n} initiates tx^{m} of frame F at $t = t_0 + t$
  - F suffers collision if
    - Another st^{n} has started tx^{m} frame F_0 after t_0, but well before $t_0 + t$ (F_0's end collides with F's start)
    - Another stn starts tx^{m} F_1 before $t_0 + 2t$ but well after $t_0 + t$ (F's end collided with F_1's start)
    - Vulnerable period of $2t$ - for frame F to be collision free no other st^{n} should tx^{m} during period t_0 to $t_0 + 2t$

- Probability that k frames are generated (by all st^{ns}) during a given frame time t is

$$P_k(t) = \frac{G^k}{k!} e^{-G}$$

Poisson's formula

$$P_n(t) = \frac{(\lambda t)^n}{n!} e^{-\lambda t}$$

λt is the no of frames generated in one frame time t here it's the total offered traffic G

- Prob. that k frames are generated in interval $2t$ (vulnerable period)

$$\frac{(2G)^k}{k!} e^{-2G}$$

- Probability P_0, that there is no collision - no other frame generated in "vulnerable period" is (k=0)

$$P_0 = \frac{(2G)^0}{0!} e^{-2G} = e^{-2G}$$

- Throughput is given by

$$S = GP_0 = Ge^{-2G}$$

- S is maximized at $G = 0.5$, i.e., $S_{max} = \frac{1}{2e} = 0.184$
- Max possible channel utilization is only about 18%.

→ Slotted ALOHA

- Time is divided into discrete intervals (not continuous as in pure ALOHA)
- Time interval / slot is equal to one frame time, i.e, t
- Centralized ~~sync~~ sync needed to broadcast signal at beginning of each time slot
- St^n can start tx^m frame only at beg. of a time slot. If frame is ready during a time slot, st^n has to wait till beg. of next slot

→ Vulnerable period reduced to t

→ Prob. P_0 that there is no coll. - no other frame gen. in "vulnerable period", i.e., k=0

$$P_0 = \frac{(G)^0}{0!} e^{-G} = e^{-G}$$

→ Throughput

$$S = GP_0 = Ge^{-G}$$

S is maximised at $G = 1.0$, so $S_{max} = \frac{1}{e} = 0.368$

→ Operated at $G = 1$, ie, 1 frame generated per 1 frame time

Max. channel utilization = 37%

- Performance of slotted ALOHA

  - $G = 1$ gives best throughput, operating with $G > 1$ decreases number of empty slots, but increases
  - Prob of tx^{m} frame avoiding collision is e^{-G}
  - Prob of coll. is $(1 - e^{-G})$
  - Prob of a frame requiring exactly k attempts to succeed (k-1 collisions followed by coll. free tx^{m}) is

$$P_k = (1 - e^{-G})^{k-1} e^{-G}$$

- Total expected number of tx^{m}. per time slot is

$$E = \sum_{k=1}^{\infty} k P_k = \sum_{k=1}^{\infty} k e^{-G} (1 - e^{-G})^{k-1} = e^{G}$$

- ~~Even~~ Even a very small increase in channel load G can appreciably degrade its performance (exponential dependence)

Pg 265, fig 3

o p- pe

o Cou

- p-persistent CSMA-CD

  - Used only with slotted channel (time-slot = frame tx^m time)
  - St^n continuously senses channel, at start of time slots
  - On determining free channel
    - St^n may start tx^m with probability p
    - St^n may defer to next slot with prob. $(1-p) = q$
    - If ~~chan~~ channel is free in deferred slot, st^n may tx^m it with prob. p or defer again with prob. q
    - St^n either successfully tx^m its in some deferred slot or, if channel / slot becomes ~~buy~~ busy (tx^m from other st^n), starts afresh by continuously sensing for free channel slot once again
    - Protocol similar to slotted ALOHA "with added ~~carrier~~ carrier sense"

- Collision detection - some H/W details

  - Inherently ~~a H/W~~ an analog process
  - Special H/W listens to channel while tx^m
  - Collision is detected when received signal is of diff power level / pulse width from the one tx^m
  - Special encoding used - must allow collisions of 2 (or more) 0's or two (or more) 1's to be detected
  - Encoding must allow st^n to synchronise using local 'clock' - global clock not feasible because of clock skew over long cable lengths
  - Ethernet uses Manchester encoding, with high & low represented by +0.85 and -0.85 volts respectively; 0 volts indicate idle channel
  - Diff manchester encoding is used in some LANs : eg, token ring

- Single channel CSMA/CD is inherently half duplex - cannot send and receive frames simultaneously
  - tx^m st^n uses receiving H/w to detect collision

fig.

→ <u>Collision detection - the model</u>

fig.

→ Model used in CSMA/CD and many other LAN protocols:

- A st^n finishes successful frame tx^m at t_0
- Ready st^ns start tx^m (as channel is sensed free/idle)
- If 2 or more st^ns transmit, a collision results (contention period)
- Collision continues for a finite time, i.e., for a contention slot
- st^ns detect collision and abort frame tx^m
  - Contention slot width = time needed by st^n to detect collision and stop tx^m
- Stations wait for arbitrary time and attempt frame tx^m again
- Further collisions result in one or more contention slots else a st^n successfully tx^mits its frame
- Idle st^n period - no st^n ready with frame to tx^mit

→ <u>Collision detection time - contention slot width</u>

- Dependent on signal propagation time

fig

- Propagation delay between farthest stations (A & B) = τ
- St^n A tests channel free and starts tx^m frame to B at t_0
- Frame should normally (collision free) reach B at $t_0+\tau$
- St^n B tests channel at $t_0+\tau-\varepsilon$, where $\varepsilon \to 0$, finds it free and starts ~~a~~ tx^m a frame that travels towards A
- At $t_0+(\tau-\varepsilon)+\varepsilon/2 = t_0+\tau-\varepsilon/2$, the two frames collide
- Collision signal (garbled frame) propagates towards both A & B
- Signal gets back to B at $(t_0+\tau-\varepsilon/2)+\varepsilon/2 = t_0+\tau$, i.e., just ~~of~~ after a delay of ε, and B stops tx^m itting
- Sending st^n must continue to listen to channel for atleast 2τ after start of tx^m to ensure there is no collision

CSMA protocols

→ Contention Free protocols (CSMA/CA)

- Collision during contention periods degrade performance
  - Channel utilization decreases and frame delay increases
- Situation further aggravated by
  - Large propagation delay τ (longer cables)
  - Short frame times (higher b/w cables, eg: optic fibre)
- Contention free protocols are quasidynamic and do not suffer collisions even during so called "contention periods"
- Assumption: exactly N stations wired into multi-access channel

→ Basic bitmap protocol

fig

- Contention interval divided into exactly N-1 bit wide contention slots
- Time measured in 'contention slot' unit, i.e.; 1 bit time
- Frame length assumed to be d-bits and frame tx^{n} time as d units
- During 'contention interval', each st^{n} waits for its 'contention slot'
  - ~~A~~ ready st^{ns} ~~writes~~ write a '1' in their own slots
  - st^{ns} not ready to tx^{m} write a '0' in their own slot
- All st^{ns} continuously listen to channel
- After 'contention interval', all st^{ns} know which st^{ns} are ready to tx^{m} it
- st^{ns} tx^{m} frames in order of their st^{n} no. There is no collision since all st^{ns} know whose turn it is to tx^{m} it. next
- After all ready st^{ns} complete frame tx^{m}, the cycle is repeated

Approx Analysis

fig.

- Light traffic load - channel mostly idle; occasional ready st^{n}
  - When a low numbered st^{n} (0 or 1 etc) becomes ready, 'current' contention slot will on an avg., be the middle one, i.e.; $\approx N/2$

- Delay before frame tx^m $= N/2 + N$ (a st^n writes a 1 in it's own slot)
  $= 1.5N$

- For a ready high numbered st^n (N, N-1, etc), avg. delay before frame tx^m $= N/2$

- Mean wait for any st^n $= \frac{(1.5+0.5)N}{2} = N$

- Channel utilization $= \frac{d}{N+d}$ *

→ <u>Modified bit-map protocol</u>

- A ready st^n ~~writes~~ writes a '1' into it's contention slot and immediately begins tx^m its frame
- After frame tx^m, contention period does not start with slot '0' again. Rather, it starts with slot number for st^n following the one that just tx^m ited successfully
- 'Permission to send'
- Each contention interval contains a run of '0' slots followed by exactly 1 slot

→ <u>Approx Analysis</u>

- Light load - st^ns wait for N/2 slots on an average before frame tx^m (better than basic)
- Heavy load - performance similar to basic protocol, since major delay is from frame tx^m, not contention slots

* <u>Approx analysis for basic bitmap protocol</u>

Heavy ~~load~~ traffic : Almost all st^{ns} ready to tx^mit

- N-bit contention interval prorated over N st^{ns} give avg. delay before tx^m ≈ 1 bit time
- channel utilization $= \frac{d}{d+1}$

→ Binary countdown protocols

- Tx^{m} delay is assumed ~~neglible~~ negligible, i.e., all st^{ns} read 'set' bits at the same time
- Bidding cycle
  - St^{n} broadcasting a '0' in a contention slot gives up for a current cycle if it reads back a 1
  - Process continues until a single st^{n} is left as a winner which then tx^{m}its its frame
- New bidding cycle begins if there are ready st^{ns}

→ Binary countdown (analysis)

- Higher no. st^{n} has higher priority; may be good or bad depending on context
- Channel utilization $= \dfrac{d}{d + \log_2 N}$
- Frame payload normally contains sender's address
- St^{n} label overhead ($\log_2 N$) can be avoided if frame format is designed cleverly and protocol allowed to use sender's adr. portion of frame payload

→ Modified Binary countdown - Mok & Ward protocol

- Works properly only for high traffic
- Uses parallel instead of serial interface to perform binary bidding

- St^{ns} are assigned virtual numbers from 0 to N and these are used as st^n labels for bidding process
- Virtual st^n numbers permuted after each successful tx^n
- St^n with nos. less than that of the successful station are reassigned numbers, incremented by 1
- St^n which ~~originally~~ successfully tx^m is assigned st^n no. 0
- Thus results in rotating priority among all st^{ns}

Example: St^{ns} A, B, C, D, E, F, G, H are assigned virtual st^n numbers 7, 6, 5, 4, 3, 2, 1, 0

After successful tx^m by st^n D, new virtual nos. are

A	B	C	D	E	F	G	H
7	6	5	0	4	3	2	1

Adaptive B. search
root is level 0