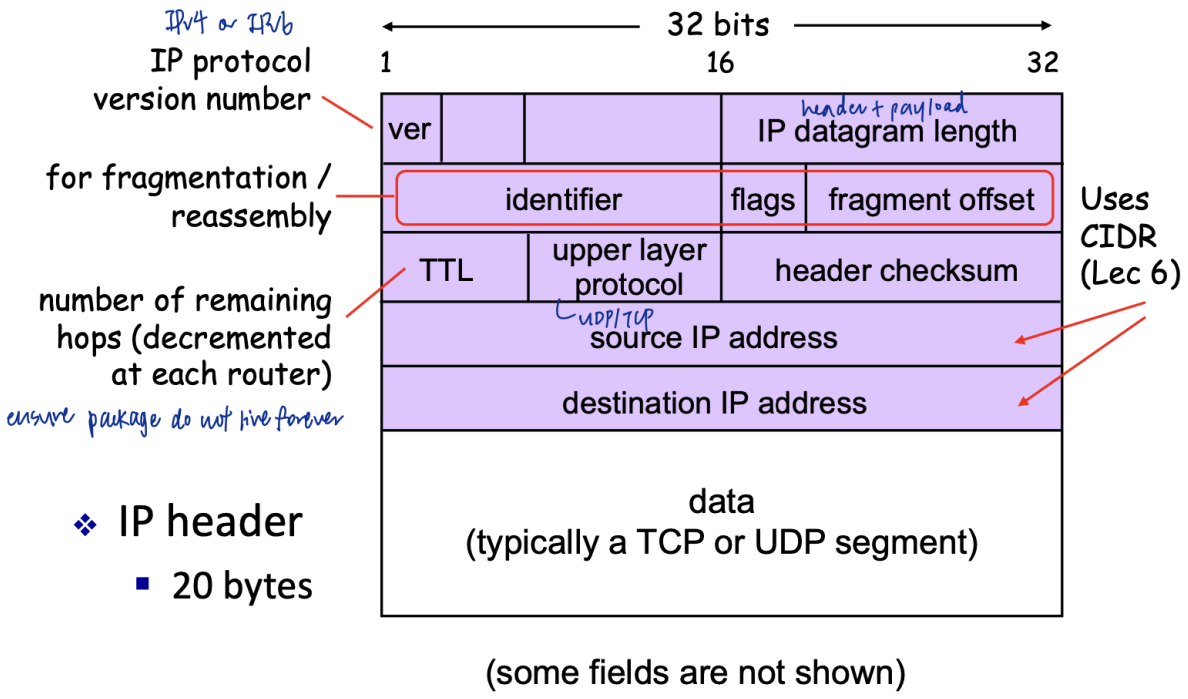
CS2105 Part 2

L7: Network Layer (datagram)

- provides communication between any two hosts
- an IP datagram may travel through multiple routers and links before it reaches destination
- routers run on
    - IP
    - RIP
    - ICMP
    - ARP

Internet Protocol (IPv4)

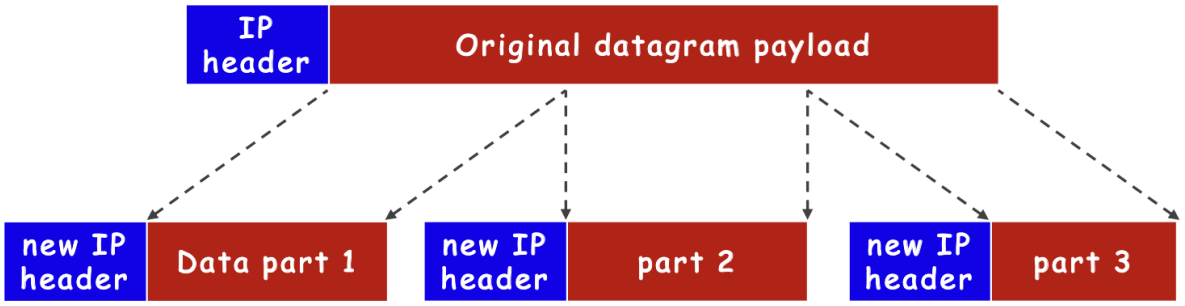
Datagram Format



- TTL: stores a counter that counts from certain value (say 120 seconds) down to zero. If the value reaches zero then the entry is deleted from the ARP table. This is to keep the table small and only remember IP/MAC pairs of nodes with which host A had recent communications.
  - only IP nodes have ARP tables
  - IP nodes are hosts or routers
  - TTL do not drop when it goes through a switch

IP Fragmentation

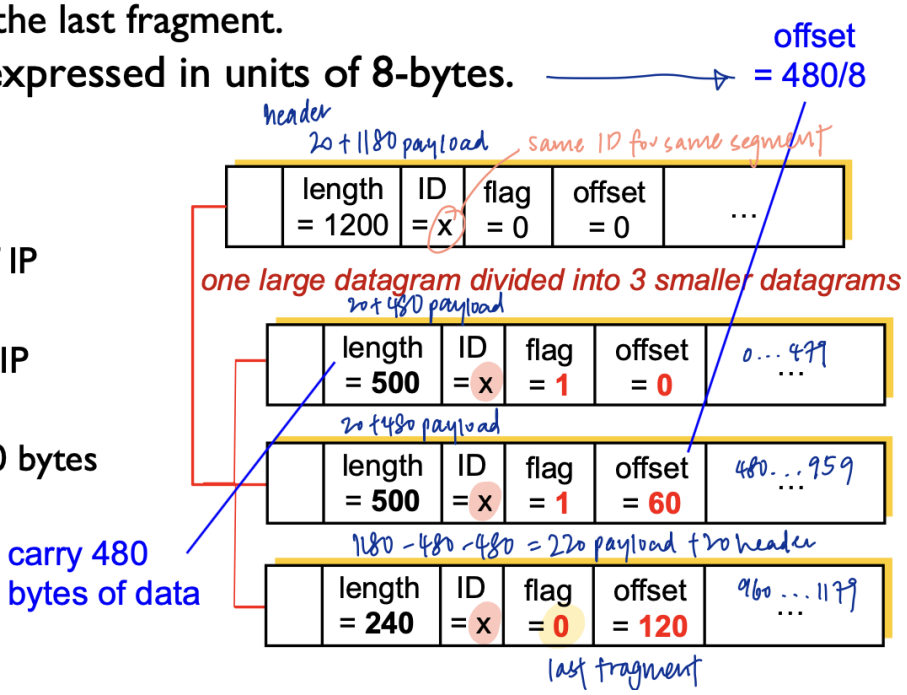
- different links may have different Max Transfer Unit (MTU) - the max amount of data a link-level frame can carry
  - includes IP and TCP header (usually up to 40 bytes)
- payload size > MTU ⇒ fragmentation
- destination host will reassemble the packet
- header fields are used to identify fragments and their relative order



- Flag: 0 for last segment, 1 otherwise

- Flag (frag flag) is set to
  - 1 if there is next fragment from the same segment.
  - 0 if this is the last fragment.
- Offset is in expressed in units of 8-bytes.

- Example
  - 20 bytes of IP header
  - 1,200 byte IP datagram
  - MTU = 500 bytes



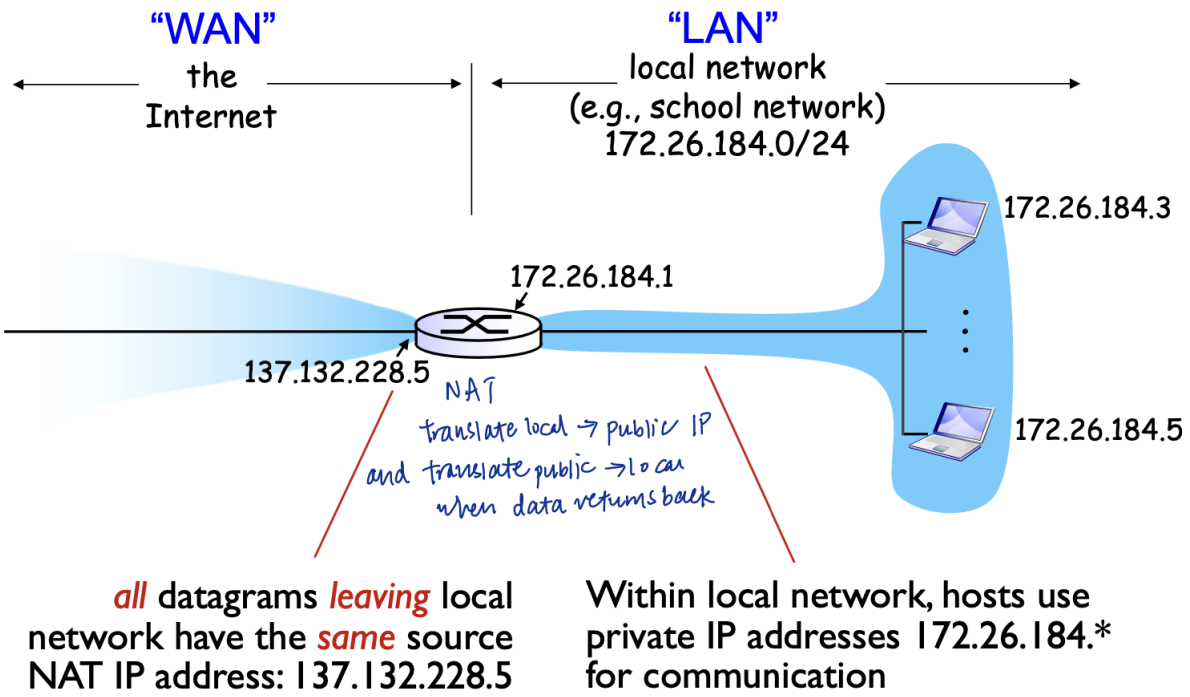
data length = IP datagram - IP header = 1200 - 20 = 1180 bytes
max size of data in each fragment = 500 - 20 = 480 bytes

Network Address Translation (NAT)

2 types of IP

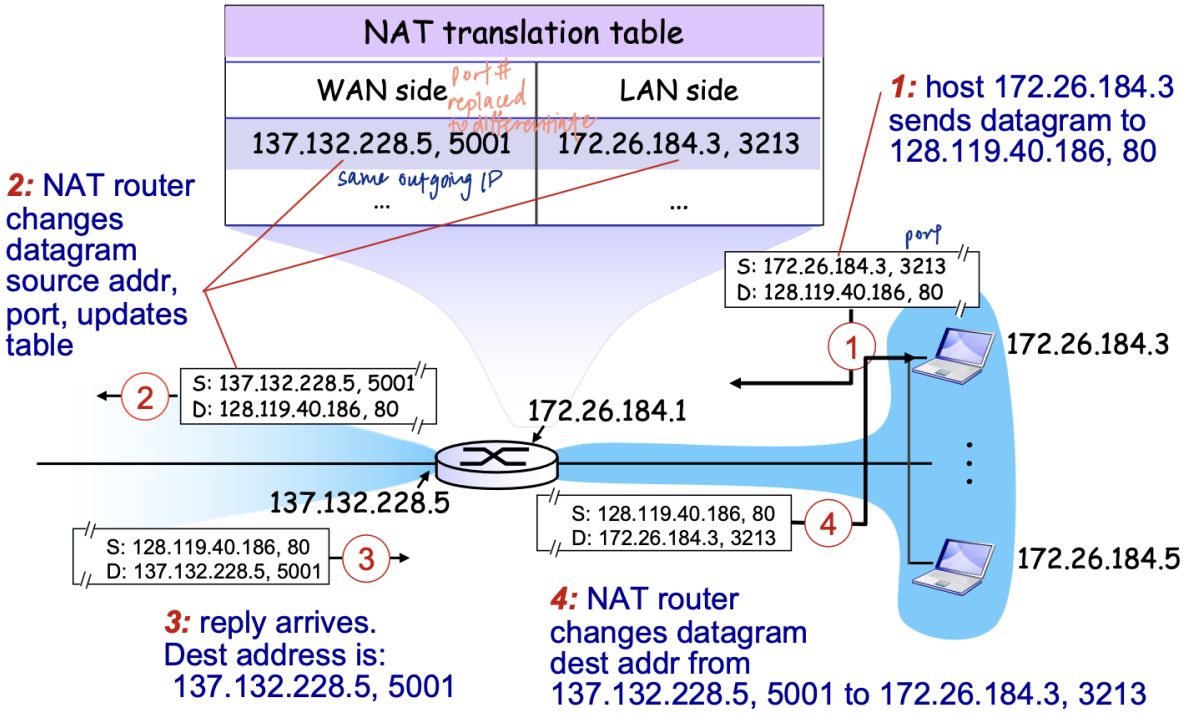
1. Public: globally unique and routable
2. Private: not globally unique ad not routable on the backbone Internet, but are routable within an organisation

- NAT translates local to public and vice versa
- not every router has NAT

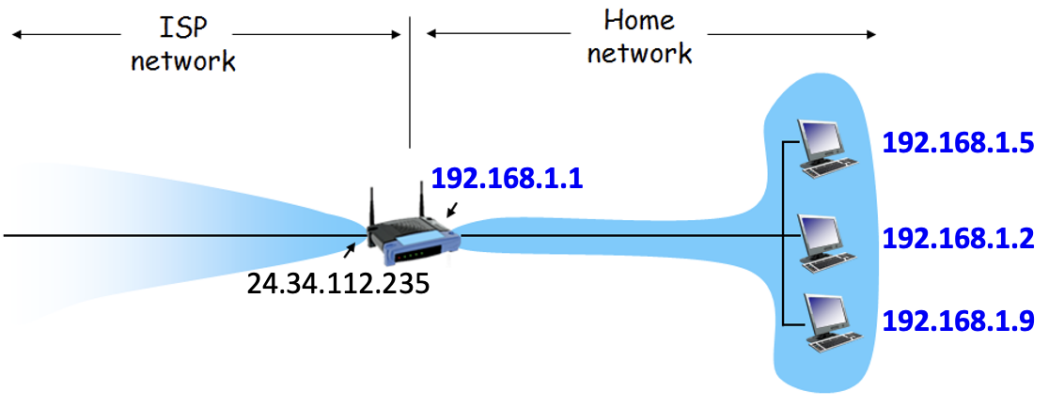


NAT routers must:

- Replace (source IP address, port #) of every outgoing datagram to (NAT IP address, new port #).
- Remember (in NAT translation table - stateful) the mapping from (source IP address, port #) to (NAT IP address, new port #).
- Replace (NAT IP address, new port #) in destination fields of every incoming datagram with corresponding (source IP address, port #) stored in NAT translation table.


NAT translation table
WAN side
LAN side
137.132.228.5, 5001
172.26.184.3, 3213
...
...
port # replaced
same outgoing IP
1: host 172.26.184.3 sends datagram to 128.119.40.186, 80
2: NAT router changes datagram source addr, port, updates table
S: 172.26.184.3, 3213
D: 128.119.40.186, 80
port
1
172.26.184.3
2
S: 137.132.228.5, 5001
D: 128.119.40.186, 80
172.26.184.1
137.132.228.5
S: 128.119.40.186, 80
D: 137.132.228.5, 5001
3
S: 128.119.40.186, 80
D: 172.26.184.3, 3213
4
172.26.184.5
3: reply arrives. Dest address is: 137.132.228.5, 5001
4: NAT router changes datagram dest addr from 137.132.228.5, 5001 to 172.26.184.3, 3213

[Modified from KR, Chapter 4, P21] Consider the network setup in the following figure. Suppose that the ISP assigns the router the address 24.34.112.235 and that the network address (i.e. network prefix) of this home network is 192.168.1/24.



a) Give an example IP address assignment to all interfaces in this home network.

b) Suppose each host has two ongoing TCP connections, all to port 80 of a server at 128.119.40.86. Provide example corresponding entries in the NAT translation table.

NAT Translation Table	
WAN side	**LAN side**
24.34.112.235, 3000	192.168.1.5, 2105
24.34.112.235, 4000	192.168.1.5, 2106
24.34.112.235, 5000	192.168.1.2, 2105
24.34.112.235, 6000	192.168.1.2, 2108
24.34.112.235, 7000	192.168.1.9, 4000
24.34.112.235, 8000	192.168.1.9, 5000

Routing Algorithms

- AS: autonomous systems e.g. ISPs, each owns routers and links

Intra-AS	Inter-AS
finds a good path between two routers within an AS	handles the interfaces between ASs
commonly used protocols: RIP, OSPF	de facto (in effect but not officially recognised) standard protocol: BGP
single admin ⇒ no policy decisions needed	admin often wants to control over how its traffic is routed, who routes through its net etc
routing mostly focus on performance	policy may dominate over performance

Distance Vector Algorithm

Properties:

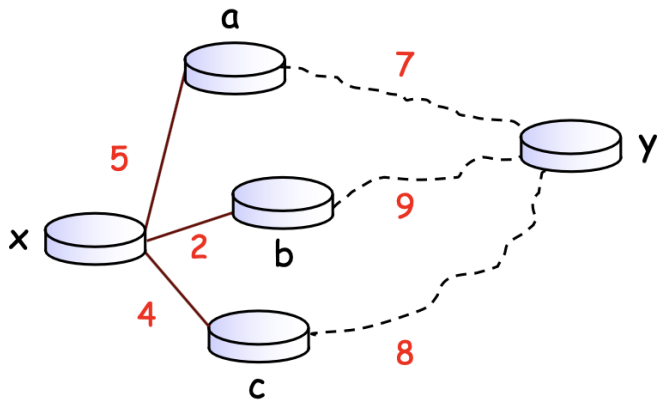
- Distributed: each node received information from one or more of its direct neighbours, performs calculation and then distributes the result back to its neighbours
- Iterative: process continues until no more information is available to be exchanged between neighbours
- Asynchronous: nodes do not need to operation in the lock step with each other

- vertices: routers
- edges: physical links between routers
- cost:
  - 1
  - inversely related to bandwidth
  - directly related to congestion
- Routing: finding a least cost path between two vertices in a graph

Iterative process of computation (Bellman-Ford)

1. swap local view with direct neighbours
2. update own local view
3. repeat steps 1-2 until no more change to local view

- **c(x, y):** the cost of link between routers x and y
  - = infinite if not direct neighbours
- $d_x(y)$: the least cost of path from x to y (from x's view)
  - use Bellman-Ford equation below
  - $d_x(y) = min_v\{c(x,v) + d_v(y)\}$ where min is taken over all direct neighbours v of x



$$d_x(y) = min_v \{ c(x,a) + d_a(y), \; c(x,b) + d_b(y), \; c(x,c) + d_c(y) \}$$

$$= min \{12, 11, 12\} = 11$$

Application in Network

1. Every router, **x**, **y**, **z**, sends its distance vectors to its directly connected neighbours.
2. When x finds out that y is advertising a path to z that is cheaper than x currently knows,
   a. **x** will update its distance vector to **z** accordingly.
   b. In addition, **x** will note down that all packets for **z** should be sent to **y**. This info will be used to create forwarding table of **x**.
3. After every router has exchanged several rounds of updates with its direct neighbours, all routers will know the least-cost paths to all the other routers.

Routing Information Protocol (RIP)

- implements the Distance Vector algo
- uses hop count as the cost
- entries in the routing table are aggregated subnet masks (so we are routing to destination subnet)
- exchange routing table every 30 seconds over UDP port 520
- “Self-repair”: no update from neighbouring router for 3min ⇒ failed

Internet Control Message Protocol (ICMP)

- used by hosts and routers to communicate network-level information
  - error reporting: unreachable host/network/port/protocol

    - echo request/reply (used by ping)
- ICMP header starts after IP header
    - IP | ICMP | Data

Type	Code	Description
8	0	echo request (ping)
0	0	echo reply (ping)
3	1	dest host unreachable
3	3	dest port unreachable
11	0	TTL expired
12	0	bad IP header

ICMP header: Type + Code + Checksum + others.

- `ping`: check if there is a connection, whether the host responds
- `traceroute` : sends a series of small packets across a network and attempts to display the route (path) that the packets take
- e.g. when TTL is 0,a packet is discarded and an ICMP error message is sent to the datagram's source address

L8: Link Layer Part 1 (frame)

- sends datagram between adjacent nodes (hosts or routers) over a single link
    - IP datagrams called frames here
    - different link-layer protocol may be used on different links: provide different set of services

Possible Link Layer Services

Framing

- adds header and trailer to datagram ⇒ frame

payload = IP datagram

Link Access Control

- When multiple nodes share a single link, need to coordinate which nodes can send frames at a certain point of time.

Reliable Delivery

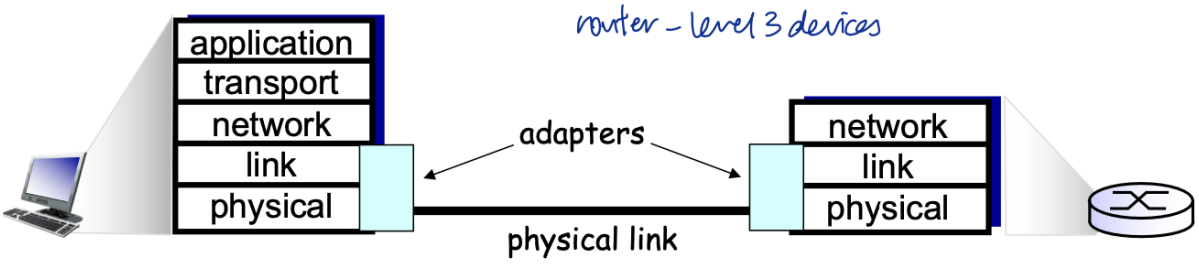
- often used on error -prone links (e.g. wireless)

Error Detection

- errors are usually caused by signal attenuation or noise
- receiver detects errors and may signal sender for retransmission or simply drop frames

Error Correction

- receiver identifies and corrects bit errors without resorting to retransmission



- Link layer is implemented in “adapter” (aka Network Interface Controller (NIC)) or on a chip
  - e.g. Ethernet card/chipset, 802.11 card
- Adapters are semi-autonomous, implementing both link and physical layers

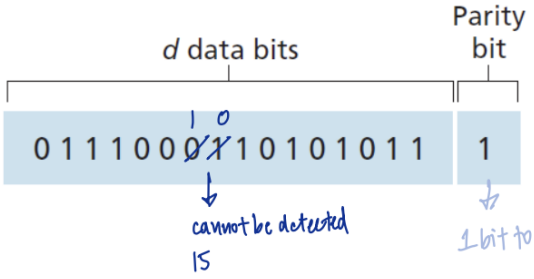
Error Detection and Correction

Parity

- even parity: even number of 1s → 0

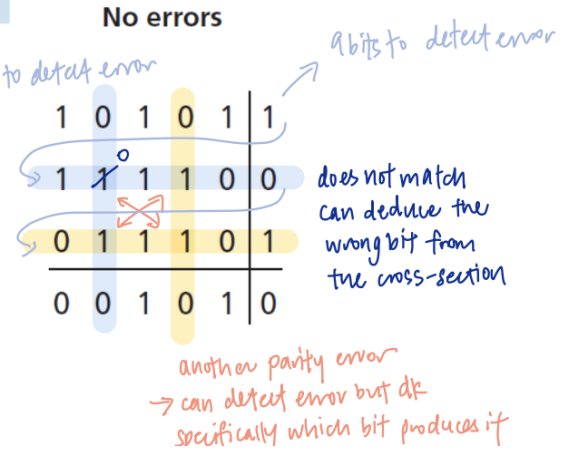
Single bit parity

- can <u>detect</u> single bit errors in data.



Two-dimensional bit parity

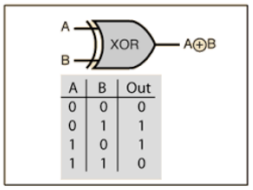
- can <u>detect and correct</u> single bit errors in data.
- can <u>detect</u> any two-bit error in data.



Cyclic Redundancy Check (CRC)

- CRC overhead: amount of data that is used for error detection (and correction) divided by the useful data.
  - e.g. r = 3, d = 6 bits ⇒ overhead = 50%

- Powerful error-detection coding that is widely used in practice (e.g., Ethernet, Wi-Fi)
  - D: data bits, viewed as a binary number.
  - G: generator of $r + 1$ bits, agreed by sender and receiver beforehand.
  - R: will generate CRC of r bits.



↑r ↓error ↑overhead

Example: $r = 3$

add 3 zeros

$$1001 \overline{)\,101110000}$$

XOR

$$\begin{array}{r} 1001 \\ \hline 1010 \\ 1001 \\ \hline 1100 \\ 1001 \\ \hline 1010 \\ 1001 \\ \hline 011 \end{array}$$

G D R

- CRC calculation is done in bit-wise XOR operation without carry or borrow.
- Sender sends (D, R)

  101110011
- Receiver knows G, divides (D, R) by G.
  - If non-zero remainder: error is detected!

Example: $r = 3$

$$1001 \overline{)\,101110000}$$

$$\begin{array}{r} 1001 \\ \hline 1010 \\ 1001 \\ \hline 1100 \\ 1001 \\ \hline 1010 \\ 1001 \\ \hline 011 \end{array}$$

G D R

Multiple Access Links & Protocols

Type 1: point-to-point link

- a sender and a receiver connected by a dedicated link

- e.g. protocols: Point-to-Point Protocol (PPP), Serial Line Internet Protocol (SLIP)
    - no need for multiple access control

Type 2: broadcast link (shared medium)

- multiple nodes connected to a shared broadcast channel
- when a node transmits a frame, the channel broadcasts the frame and each node receives a copy
- e.g. Wi-Fi, Satellite, Ethernet with bus topology
- if ≥ 2 nodes transmit simultaneously
    - every node receives multiple frames simultaneously → frames collide at nodes and none would be correctly read

Multiple Access Protocols

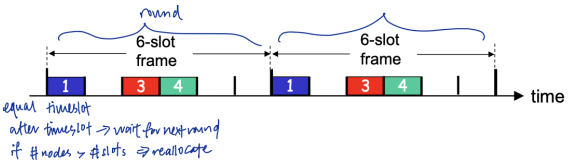
- distributed algo that determines how nodes share channel, i.e. when a node can transmit
- however, coordination about channel sharing must use channel itself → no out-of-band channel signalling

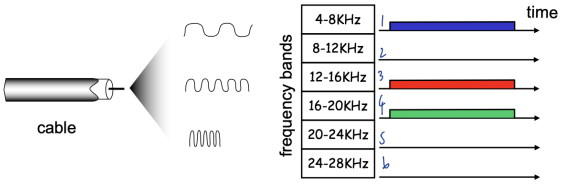
1. Channel Partitioning
    a. divide channel into fixed, smaller partitions (e.g. time slots, frequency)
    b. allocate piece to node for exclusive use
2. “Taking turns”: nodes take turn to transmit
3. Random Access
    a. channel is not divided, collisions are possible
    b. “recover” from collisions

Channel Partitioning Protocols

	Time Division Multiple Access (TDMA)	Frequency Division Multiple Access (FDMA)
How it works	• access channel in “rounds” • each node gets fixed length slot (length = frame transmission time)	• channel spectrum is divided into frequency bands • each node is assigned a fixed frequency band •

	Time Division Multiple Access (TDMA)	Frequency Division Multiple Access (FDMA)
	in each round • unused slots go idle → resource wastage	unused transmission time in frequency bands go idle

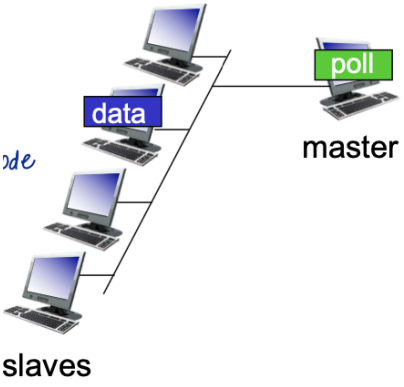




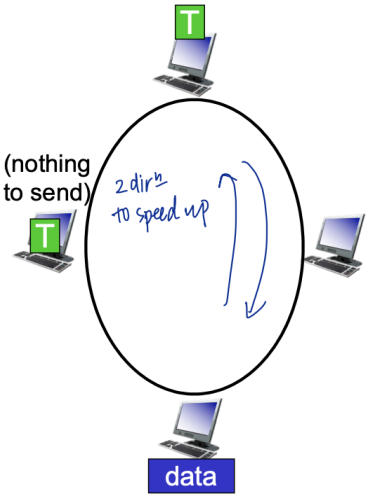
TDMA - 6 nodes sharing a link: 1, 3, 4 have frames, 2, 5, 6 are idle

“Taking Turns”

	Polling	Token Passing / Token Ring
How it works	master node “invites” slave nodes to transmit in turn	control token is passed from one node to next sequentially
Concerns	• polling overhead (to figure out if the next node has sth to send) • single point of failure (master node) → need an algo to elect the new master	• token overhead • single point of failure (token can be lost)



Polling



Token Passing

Random Access Protocols

- When a node has packet to send:
  - no a priori (deductive) coordination among nodes
  - ≥ 2 transmitting nodes → collision
- Random access protocols specify:
  - how to detect collisions
  - how to recover from collisions (e.g. via delayed transmissions)

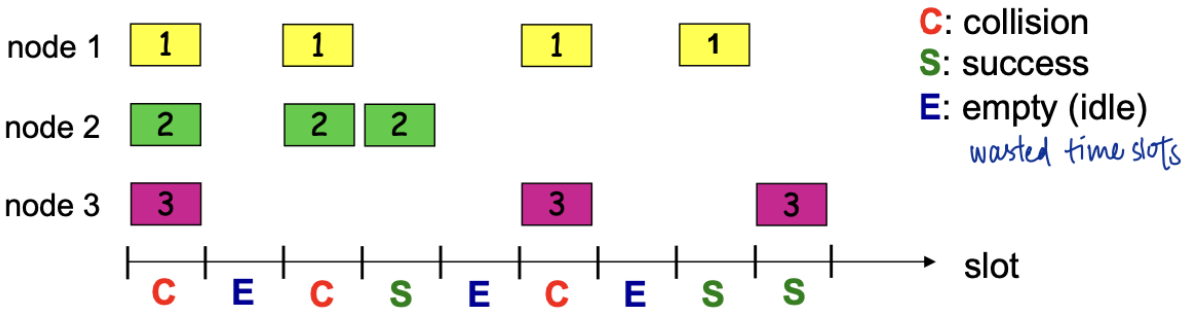
Slotted ALOHA

Assumptions

- all frames are of equal size
- time is divided into slots of equal length
  - length = time to transmit 1 frame
- nodes start to transmit only at the beginning of a slot

Operations

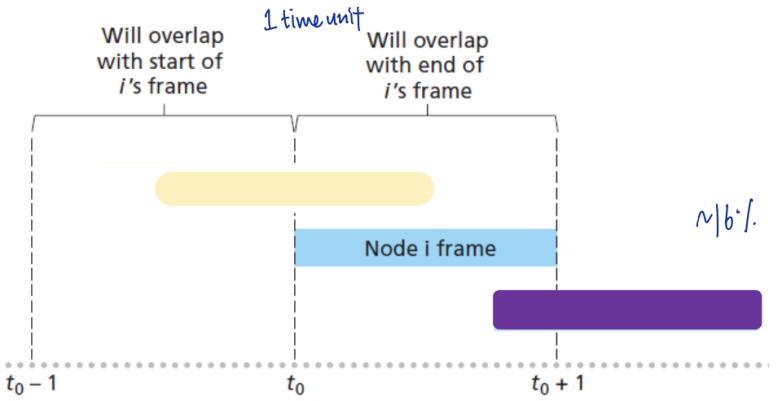
- listens to the channel while transmitting (collision detection)
- if collision happens: node retransmits a frame in each subsequent slot with probability p until success



Pure (unslotted) ALOHA

- no slot, no synchronisation

- when there is a new frame: transmit immediately
- chance of collision increases
  - frame sent at t0 collides with other frames sent in (t0 - 1, t0 + 1)



Carrier Sense Multiple Access (CSMA)

- sense channel before transmission
- if channel sensed idle → transmit frame
  - if 2 nodes sense idle simultaneously and both start transmission → collision
  - propagation delay means two nodes may not hear each other's transmission immediately
- if channel sensed busy → defer

CSMA/CD (Collision Detection)

- when collision is detected, transmission is aborted to reduce channel wastage
- retransmit after a random amount of time
- if frame size is too small → collision happens but may not be detected → no retransmission
- CSMA/CD is used in Ethernet. Ethernet requires a min frame size of 64 bytes

CSMA/CA (Collision Avoidance)

- collision detection is difficult in wireless LANs
- WiFi (802.11) uses CSMA/CA → receiver needs to return ACK if frame is received ok

▼ **There are many nodes in a shared medium network and most nodes are likely to transmit frequently. Which of the following multiple access protocol(s) is (are) suitable? (1) TDMA; (2) CSMA; (3) Token passing.**

- TDMA and token passing are suitable because there is sufficient work to do to utilise the "fixed" resources allocated.
- CSMA is not because many nodes competing for the shared channel can result in lots of collision. Utilisation will be low.

▼ Nodes A and B are accessing a shared medium using CSMA/CD protocol, with **propagation delay of 245 bit times** between them (i.e., propagation delay equals to the amount of time to transmit 245 bits onto the link). Minimum frame size is 64 bytes.

Suppose node A begins transmitting a frame at t = 0 bit time. Before A finishes, node B begins transmitting a frame. Assume no other nodes are active.

(a) When is the latest time, by which B can begin its transmission?

(b) Suppose B begin its transmission at the time computed in a), can A detects that B has transmitted before it finishes transmission?

(a) latest time is before the signal from A reaches B ⇒ t = 244 bit times

(b) B begins transmission at t = 244 bit time. Signal propagates to A at t = 244 + 245 = 489 bit time ⇒ A can detect collision before it finishes transmission at t = 512 bit time.

At the link layer there exist several link access protocols. In some of the protocols the access is deterministic, i.e., a node can calculate and guarantee when it will be able to access the channel and successfully transmit the next frame. Some other protocols are stochastic, i.e., a sender can not know beforehand exactly when it will be able to successfully send the next frame.

Which of the access protocols below are deterministic?

- FDMA (frequency division multiple access)
- TDMA (time division multiple access)
- CSMA
- CSMA/CD
- Pure ALOHA

L9: Link Layer Part 2

Link Layer Addressing & ARP

Media Access Control (MAC) Address

- every adapter (NIC) has a MAC address (aka physical or LAN address)
- adapter used to send and receive link layer frames
- when an adapter receives a frame, it checks if the destination MAC address of the frame matches its own MAC address
  - if yes: adapter extracts the enclosed datagram and passes it to the protocol stack
  - if no: adapter discards the frame without interrupting the host
- MAC Address is typically 48 bits, burned in NIC ROM
  - e.g. `5C-F9-DD-E8-E3-D2` (hexadecimal notation)
  - MAC address allocation is administered by IEEE
  - the first 3 bytes identifies the vendor of an adapter

	IP address	MAC address
length	32 bits	48 bits
layer	network-layer address used to move datagrams from source to destination	link-layer address used to move frames over every single link
dynamic/Permanent	dynamically assigned, hierarchical (to facilitate routing)	permanent, to identify the hardware (adapter)
analogy	postal address	NRIC

- application-layer identifier is host name

Address Resolution Protocol (ARP)

Each IP node (host, router) has an ARP table

- stores the mappings of IP address and MAC address of other nodes (that have interactions) in the same subnet
- <IP; MAC; TTL>

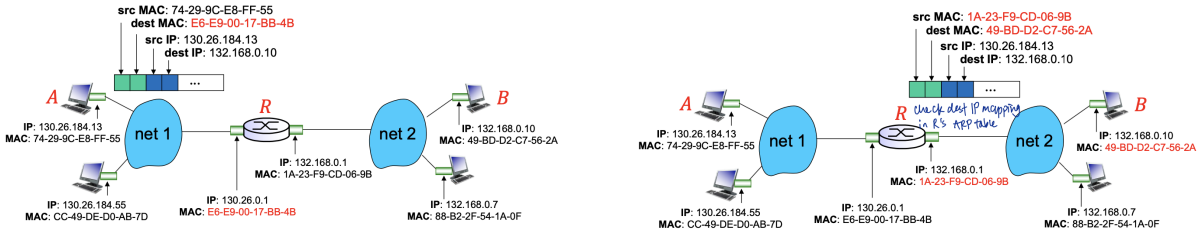
Sending Frame in the same subnet

- Suppose A wants to send data to B

1. If A knows B's MAC address from its ARP table
    a. creates a frame with B's MAC address and send it
    b. only B will process this frame
    c. other nodes may receive but will ignore this frame
2. if A does not know B's MAC address
    a. A broadcasts an ARP query packet, containing B's IP address
    b. dest MAC address set to `FF-FF-FF-FF-FF-FF`
    c. all other nodes in the same subnet will receive this ARP query packet but only B will reply

Sending Frame to another subnet

1. A create a link-layer frame with
    a. R's MAC address, and
    b. B's IP address as destination
2. R will move datagram to outgoing link and construct a new frame with B's MAC address



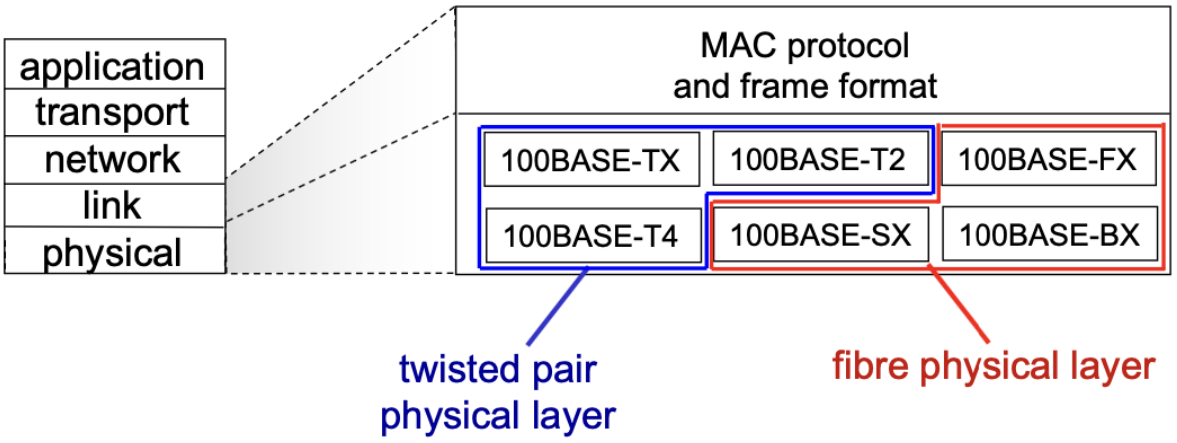
Local Area Network (LAN)

- LAN is a computer network that interconnects computers within a geographical area (e.g. office building, university campus)
- LAN Technologies
  - IBM token ring: IEEE 802.5 standard
  - ethernet: IEEE 802.3 standard
  - Wi-Fi: IEEE 802.11 standard
    - At least three non-overlapping channels are available for transmissions.
    - Unicast frames are ACKed.
  - etc
- LAN consists of ≥ 1 subnet
- as long as there is a router, the router and its hosts are considered a subnet

Ethernet

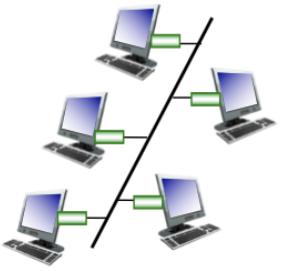
- Different speeds
- different physical layer media: cable, fibre, optics
- MAC protocol and frame format remain unchanged
- twisted pair copper connectors
  - RJ45
  - CAT 6: max speed - 100Gbps; max len - 100m
- optical fibre connectors:
  - left: LC/PC connectors
  - right: SC/PC connectors

- single-mode fibre: max speed - 10/40 Gbps; max len > 80km

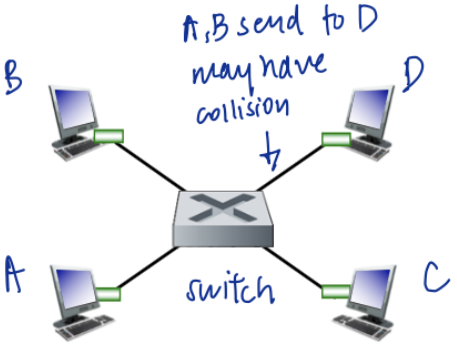


Physical Topology

Bus topology	Star topology (prevalent today)
all nodes can collide with each other	switch in center
	nodes are less likely to collide with each other



Ethernet with bus topology



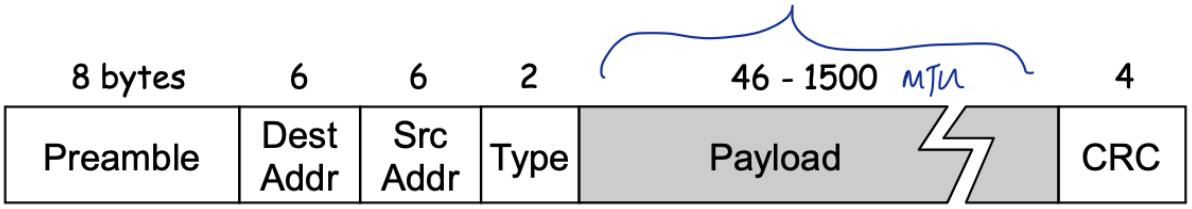
Ethernet with star topology

Frame Structure

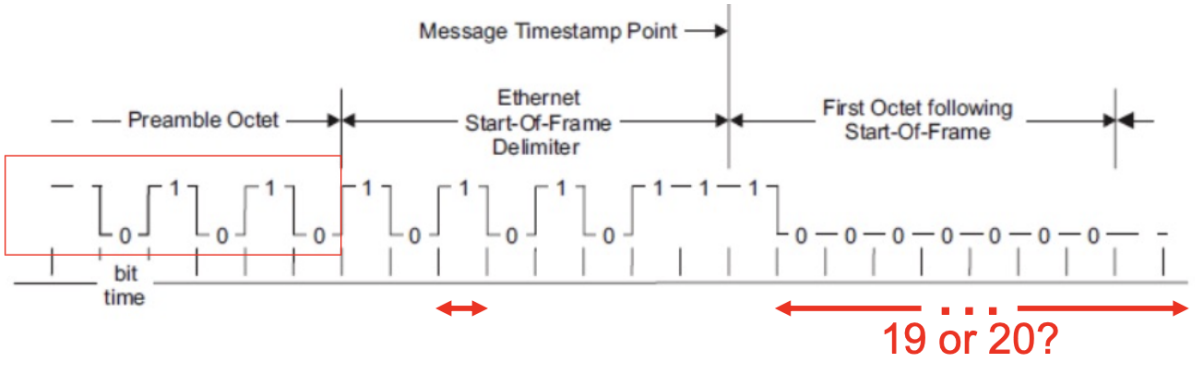
- Sending NIC (adapter) encapsulates IP datagram in Ethernet frame
- Preamble:
  - 7 bytes with pattern `10101010` (AA in hex) followed by 1 byte with pattern `10101011` (AB in hex)
  - used to synchronise receiver and sender clock rates

- provides a “square wave” pattern that tells the receiver the sender’s clock rate and the width of a bit (important if there is a long string of bits of the same values)

- Source and destination MAC address
  - NIC receives a frame with matching destination address/broadcast address → passes data in the frame to network layer protocol
  - otherwise, NIC discards frame
- Type: indicates higher layer protocol (mostly IP)
- CRC: corrupted frame will be dropped



min len = 6 + 6 + 2 + 46 + 4 = 64 bytes; IP datagram = payload



Ethernet Data Delivery Service

- Connectionless: no handshake between sending and receiving NICs
- Unreliable: receiving NIC does not send ACK/NAK
  - Ethernet drops a frame that fails error checking without retransmission (no NAK)
  - data in dropped frames will be recovered only if initial sender uses higher layer RDT (e.g. TCP); otherwise dropped data is lost
- use CSMA/CD with binary (exponential backoff)

1. NIC receives datagram from network layer, creates frame

2. If NIC senses channel idle, starts frame transmission. Otherwise, wait until channel idle then transmit
3. If NIC transmits entire frame without detecting another transmission → done
4. If another transmission detected → aborts and sends jam signal
5. after aborting, NIC enters binary backoff
    - after m-th collision, NIC chooses K at random from $\{0, 1, 2, ..., 2^m - 1\}$
    - NIC waits K * 512 bit times (time to send 512 bits) returns to step 2
    - more collisions → heavier load → longer back-off interval

Link-layer Switches

- A switch is a link-level device to interconnect between hosts.
- A switch interface DOES NOT have MAC address and IP address
- An IP address is associated with a network interface ⇒ router can have ≥ 1 IP address(es)

Ethernet Switch

- link-layer device used in LAN
- store and forward Ethernet frames
- examine incoming frame's MAC address, selectively forward frame to one-or-more outgoing links
- transparent to hosts: switches have no IP address
- in star topology ethernet, hosts have dedicated connection to switch
- switch buffers frames and is full duplex: A & D can send frames to each other simultaneously
- ethernet protocol is used on each link, but no collisions

Switch Forwarding Table

- each switch has a switch table
- format: <host MAC, interface to reach host, TTL>
- starts fresh and learns which hosts can be reached

- e.g.
  - when receiving a frame from A, note down the location of A in switch table
  - if destination B is found in table, forward to that link
  - otherwise, broadcast frame to all outgoing links

Q: Suppose 4 nodes (A-D) are connected in star topology to an Ethernet switch.

Show the state of the switch table after each of the above events (ignore TTL). For each event, identify the link(s) on which the transmitted frame will be forwarded.

Event	Switch table after event	Link(s) a frame is forwards to
B sends a frame to D	(B, 4)	1, 2, 3
D replies with a frame to B	(B, 4), (D, 3)	4
D sends a frame to A	(B, 4), (D, 3)	1, 2, 4

Differences between Routers and Switches

Routers	Switches
check IP address	check MAC address
store and forward	store and forward
compute routes to destination	forward frame to outgoing link or broadcast
Layer-3 Device (network)	Layer-2 device (link)
need manual configuration	self-learning
forward IP datagrams	forward link layer frames
used to connect subnets	used in a subnet

Switch table	ARP table
allow the switch to know which interface it should forward the frame to so that it does not need to broadcast to all the interfaces every time.	allow the sending host to map the IP address to the MAC address so that the sending host can insert the correct destination MAC address in the frames it sends.
improve performance	improve performance
hosts do not know its existence	

▼ **If all the links in the Internet were to provide reliable delivery service, would the TCP reliable delivery service be redundant? Why or why not?**

- IP datagrams in the same TCP connection can take different routes in the network, and therefore arrive at receiving host out of order.
- TCP is still needed to sort out received data in the correct order before passing them to application.
- Also, IP datagrams can be lost due to routing loops, equipment failures, etc.
- e.g., a router holding a frame can crash

▼ Suppose nodes A, B and R are star connected into a switch S. A, B and R are aware of the IP addresses of each other. Consider sending an IP datagram from host A to B. Enumerate all the steps the host and switch take to move the packet from A to B.

(a) Assume all of the ARP tables and switch table are up to date
(b) Assume that ARP table in the sending host is empty but all other tables are up to date
(c) Assume all tables are empty
(d) Suppose A sends an IP datagram to a host in another subnet. All of the ARP tables and switch table are up to date. Enumerate the steps the host, switch and router take to move the packet to another subnet.

(a)

1. A creates a frame with destination MAC address `CC-49-DE-D0-AB-7D` (B's address found in ARP table)
2. this frame travels to switch S and is forwarded to B (interface to B is found in switch table)

(b)

1. A broadcasts an ARP query packet, with destination MAC address `FF-FF-FF-FF-FF-FF`
2. Switch S forwards this ARP query packet to both B and R since destination MAC address is a broadcast address
3. R will ignore this ARP query packet but B will reply to A. Switch S forwards the reply frame towards A (interface to A is found in switch table)

4. A can then send IP datagram to B as in part (a)

(c)

1. A issues an ARP query packet to know to the MAC address of B
2. The query packet travels to switch S and is forwards to both B and R. Switch S learns that A is reachable via the interface query packet arrives at
3. R will ignore this ARP query but B will reply to A. Switch S forwards the reply frame to A (interface of A is found in switch table). Switch S learns that B is reachable via the interface reply frame arrives at.
4. A can then send IP datagram to B as in part (a)

(d)

1. A creates a frame with destination MAC address `E6-E9-00-17-BB-4B` (R's address is found in ARP table).
2. This frame travels to switch S and is forwards to R (interface of R is found in switch table)
3. R checks the destination IP of the datagram and forward it to external network. It encapsulates the IP datagram in a new frame with source MAC address as `1A-23-F9-CD-06-9B` (dest MAC not specified in qns) and sends it through the interface towards external network

Switch A has full-duplex interfaces (i.e., it can send and receive on each interface concurrently) and it can learn which computer is connected to which interface. Switch B can do neither (it always sends out frames on all interfaces, except on the one it received the frame). Both switches have *n* interfaces and each interface has a max. bandwidth of 1 Gbps in either send or receive direction. Throughput means the total number of frames that a switch can process (send and receive) per second.

Which are the properties of switch A versus switch B?

i) Switch A achieves higher total throughput because of full-duplex interfaces.

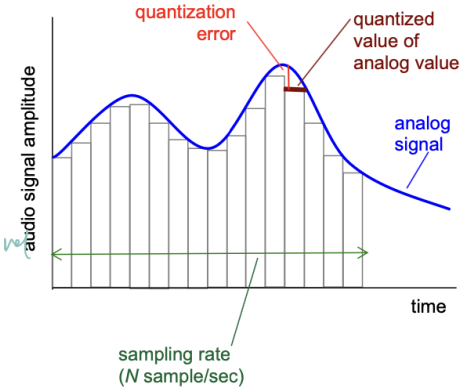
ii) Switch A achieves higher total throughput because it can learn on which interface a computer is reachable.

iii) Switch A needs more internal memory (e.g., space for the learned forwarding table).

L10: Multimedia Networking

Audio

- analog audio signal sampled at constant rate
- Nyquist-Shannon sampling theorem: $f_s \geq d \times 2$
  - f_s: sampling frequency
  - d: highest signal frequency
  - e.g. human voice: 85 - 3500 Hz ⇒ 3500*2 = 7000
  - telephone: 8000 samples/sec
  - e.g. human audible range: 20-20kHz ⇒ 20k * 2 = 40k
  - CD music: 44100 samples/sec
- each sample is quantised, i.e. rounded
  - e.g. $2^8 = 256$ possible quantised values
  - each quantised value represented by bits ⇒ 8 bits for 256 values
  - CD: $2^{16} = 65,536$
- example rates:
  - 8000 samples/sec, 256 quantised values (8 bits) ⇒ 8 * 8000 = 64,000 bps
  - CD: 16 bits/sample * 44100 samples/sec * 2 (L&R) = 1.411 Mbps
  - MP3: 96,128,160kbps
  - Intenet telephony: 5.3kbps
- receiver converts bits back to analogy signal (digital-to-analogy converter (DAC))
  - some quality reduction



Video

- videos are sequence of images displayed at constant rate
- digital image: array of pixels (each pixel represented by bits)
- two types of redundancies

	Spatial Redundancy	Temporal Redundancy
Description	Redundancy within the same image, e.g. consecutive pixels with the same colour.	Redundancy between multiple images, e.g. consecutive frames that are very similar and only have a small change between them
How to exploit for efficient compression	compress the video frame by sending just the colour value and its count (instead of the colour value *N* times).	compress the second video frame by sending only its difference from the original frame (instead of the full uncompressed frame).

- Constant Bit Rate (CBR): fixed video encoding rate
- Variable Bit Rate (VBR): video encoding rate changes as amount of spatial, temporal coding changes
- e.g.
  - MPEG1 (CD-ROM) 1.5Mbps
  - MPEG2 (DVD) 3-6Mbps
  - MPEG4/H.264 (often used in Internet, < 2Mbps)

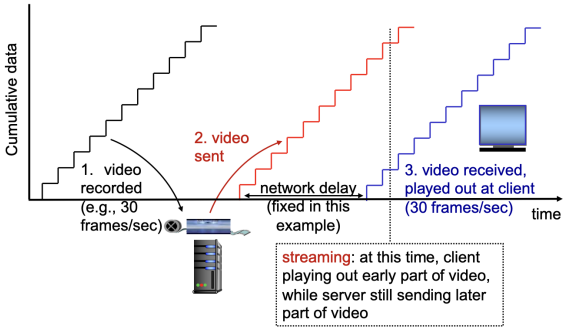
- H.265
- 4K < 85Mbps

3 application types

	Video-on-Demand (VoD)	Live Streaming	Voice over Internet Protocol (VoIP), Teleconferencing
Properties	one-way, on-demand, media is stored, long latency is ok	one-way, live-source, latency should not be too long	two-way, low latency is critical
Applications	YouTube, Netflix	Twitch.tv, Periscope	Skype, Zoom, Webex
Protocols	DASH	DASH	RTP (WebRTC)

Streaming, stored audio, video

- streaming: can begin playout before downloading entire file
- stored (at server): can transmit faster than audio/video will be rendered (implies storing/buffering at client) e.g. YouTube

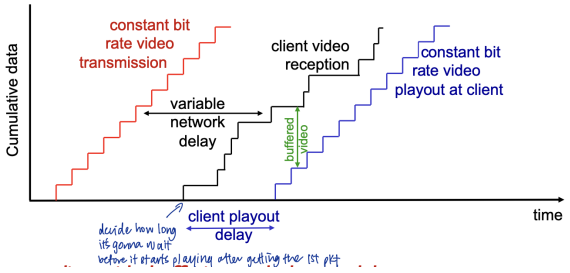


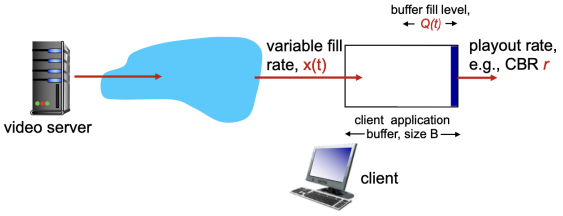
Challenges

- continuous play out constraint: network delays are variable (jitter) → need client-side buffer to match play out requirements
- client interactivity: pause, fast-forward, rewind, jump through video
- video packets may be lost, retransmitted

Client-side buffering and playout delay

- compensate for network-added delay, delay jitter





1. Initial fill of buffer until play out begins at t_p
2. play out begins at t_p
3. buffer fill level varies over time as fill rate x(t) varies and play out rate r is constant
    a. $\overline{x} < r$: buffer eventually empties
    b. $\overline{x} > r$: buffer will not empty, provided initial playout delay is large enough to absorb the variability in x(t)
        - initial playout delay tradeoff: buffer starvation less likely with larger delay, but larger delay until user begins watching

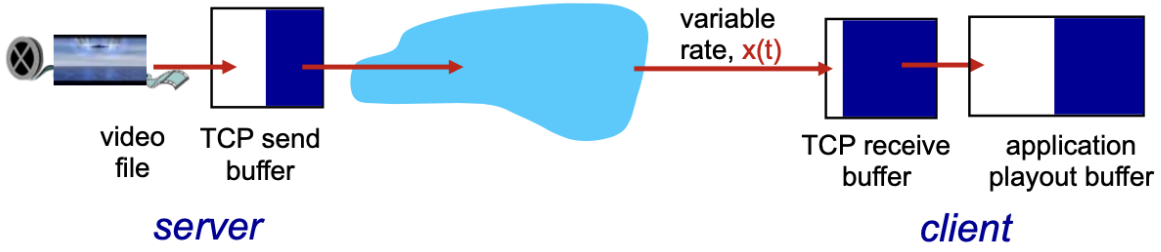
Streaming media: UDP

- (server) push-based streaming
- send rate = encoding rate = constant rate
- transmission rate can be oblivious to congestion levels
- short playout delay to remove network jitter
- error-recovery: application-level, time permitting
- Real Time Protocol (RTP): multimedia payload types
- UDP may not go through firewalls

Streaming media: HTTP TCP

- (client) pull-based streaming
- multimedia file retrieved via HTTP GET
- send at maximum possible rate under TCP
- fill rate fluctuates due to TCP congestion control, retransmissions (in-order delivery)

- larger playout delay ⇒ smooth TCP delivery rate
- HTTP/TCP passes more easily through firewalls



Conversational (”two-way live”) voice/Voice over IP (VoIP)

- interactive nature of human-to-human conversation limits delay tolerance e.g. Skype, Zoom
- Requirement
  - Good delay: < 150ms
  - Bad: > 400ms
  - includes application level (packetisation, play out) network delays
- Session initialisation: callee advertise IP address, port number, encoding algo
- value-added services: call forwarding, screening, recording
- emergency services: 999

In many VoIP applications, the audio data is sent in packets that contain 20 milliseconds worth of audio data. This is only about 160 bytes, i.e., much less than the MTU of Ethernet. The reasons for using 20 ms are as follows:

- Packets that are much shorter than 20 ms would create a relatively high overhead (the header size in comparison to the payload data size) and also a high number of packets would be generated. Therefore, 20 ms is a good compromise.
- The data length (in milliseconds) directly and proportionally adds to the end-to-end delay. Therefore, for interactive VoIP applications we do want to use fairly short packets.

Characteristics

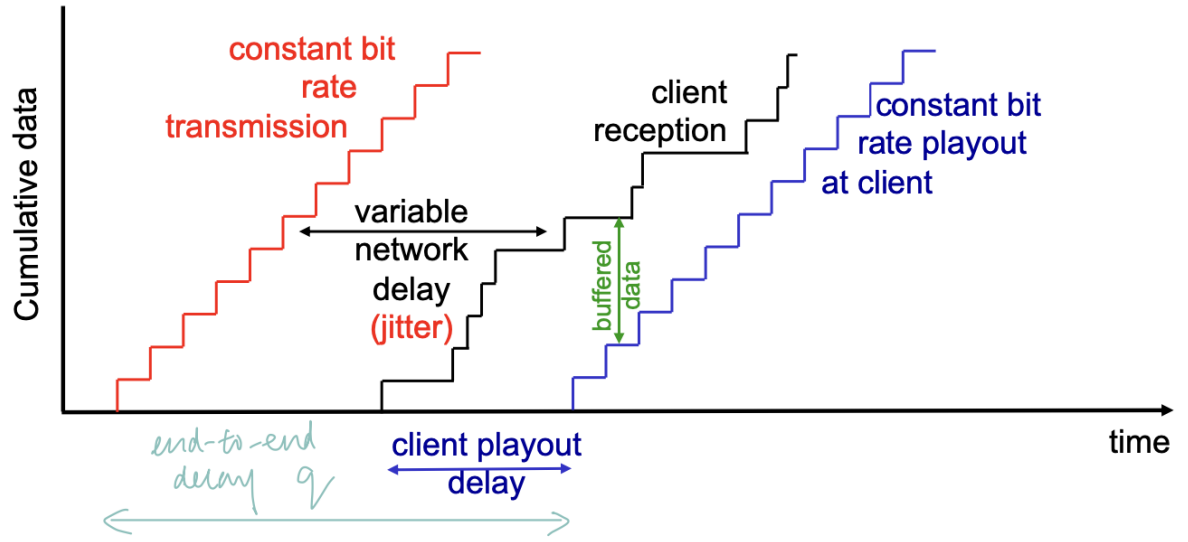
- Speaker’s audio: alternating talk spurts and silent periods

- 64kbps during talk spurt
- pkts generated only during talk spurts
- 20ms chunks at 8KB/sec ⇒ 160 bytes of data in the 20ms chunks

- application-layer header added to each chunk
- chunk+header encapsulated into UDP or TCP segment
- application sends segment into socket every 20ms during talk spurt

Packet loss & delay

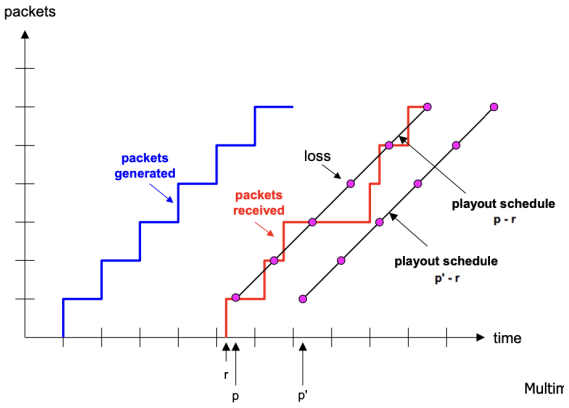
Network Loss	Delay Loss	Loss Tolerance
IP datagram lost due to network congestion (router buffer overflow)	IP datagram arrives too late for play out at receiver - delay due to processing, queueing in network, end-system (sender/receiver) delays - typical maximum tolerable delay = 400ms	depending on voice encoding, loss concealment, packet loss rates 1-10% can be tolerated



chunk arriving after t+q is too late for play out → data “lost”

- large q: less packet loss, less interactive
- small q: more packet loss, more interactive

⇒ schedule 2 play out times



Adaptive Playout Delay

- estimate network delay, adjust playout delay at the beginning of each talk spurt
- silent periods compressed and elongated
- chunks still played out every 20ms during talk spurt

estimate):

history ← delay of latest pkt

$$d_i = (1-\alpha)d_{i-1} + \alpha\,(r_i - t_i)$$

d_i: delay estimate after ith packet

α: small constant, e.g. 0.1

$r_i - t_i$: time received - time sent (timestamp) = measured delay of ith packet

- also useful to estimate average deviation of delay, v_i:

$$v_i = (1-\beta)v_{i-1} + \beta\,|r_i - t_i - d_i|$$

- estimates d_i, v_i calculated for every received packet, but used only at start of talk spurt
- for first packet in talk spurt, playout time is:

$$\text{playout-time}_i = t_i + d_i + Kv_i$$

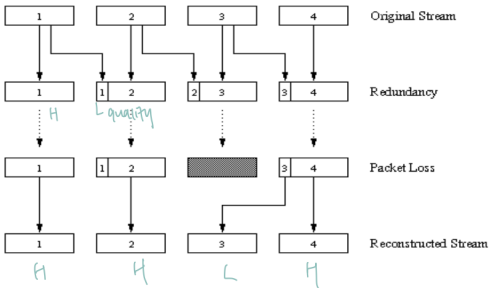
↪ 3,4

- remaining packets in talkspurt are played out periodically

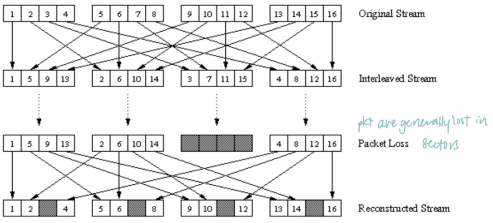
K = 3-4 std dev

Recovery from Packet Loss

- each ACK/NAK take ~1 RTT ⇒ + small tolerable delay between original transmission and playout
- Forward Error Correction (FEC)
  - send enough bits to allow recovery without retransmission
- Simple FEC
  - for every group of n chunks, create redundant chunk by XOR n original chunks
  - send n+1 chunks, increasing bandwidth by factor 1/n
  - reconstruct original n chunks if at most 1 lost chunk from n+1 chunks, with playout delay
- "Piggyback lower quality stream" FEC
  - send lower resolution audio stream as redundant information
  - non-consecutive loss → receiver can conceal loss
  - generalisation: can also append (n-1)th and (n-2)th low-bit rate chunk



- Interleaving to conceal loss
  - audio chunks divided into smaller units (e.g. four 5ms units per 20ms chunk)
  - packet contains small units from different chunks
  - if packet lost, still have most of every original chunk
  - no redundancy overhead, but increases playout delay



▼ Recall the two FEC schemes for VoIP described in lecture. Suppose the first scheme (Scheme 1) generates a redundant chunk for every four original chunks. Suppose the second scheme (Scheme 2) uses a low-bit rate encoding whose transmission rate is 25 percent of the transmission rate of the nominal stream.

(Note: we ignore the effects of playout delay in this question as we assume that all packets, including FEC packets, will be received prior to reconstruction and playback)

(a) How much additional bandwidth does each scheme require?
(b) How do the two schemes perform if the first packet is lost in every group of five packets? Which scheme will have better audio quality?
(c) How do the two schemes perform if the first packet is lost in every group of two packets? Which scheme will have better audio quality?

(a) Scheme 1: Every 4 original chunks will have 1 redundant chunk = 25% additional bandwidth; Scheme 2: Every chunk will have its redundant low-quality chunk “piggyback” on the next transmission = 25% additional bandwidth. Hence, both schemes will require additional 25% bandwidth.

(b) Scheme 1 will be able to reconstruct the original high-quality audio, while Scheme 2 will get a low-quality audio packet in every 5 packets. Hence, Scheme 1 will have better overall audio quality.

(c) Scheme 2 will have better audio quality as Scheme 1 is unable to recover from the packet losses while Scheme 2 can (albeit with low-quality). Note, redundancy-based FEC (e.g., XOR in Scheme 1) has a fixed loss limit up to which we can recover the data. If the losses exceed that limit, then all data lost in that group cannot be recovered.

Real-Time Protocol (RTP)

- runs in end systems

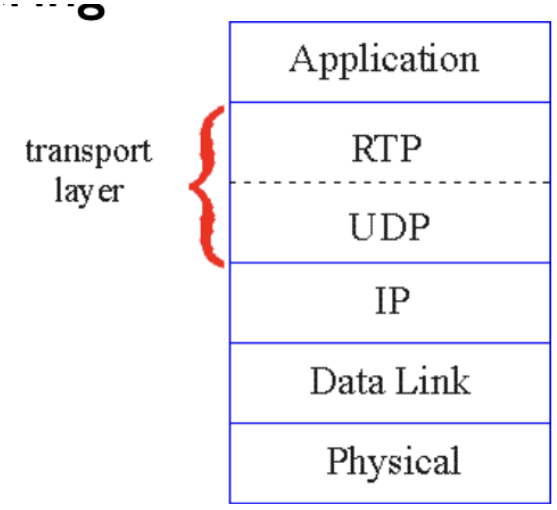
RTP header

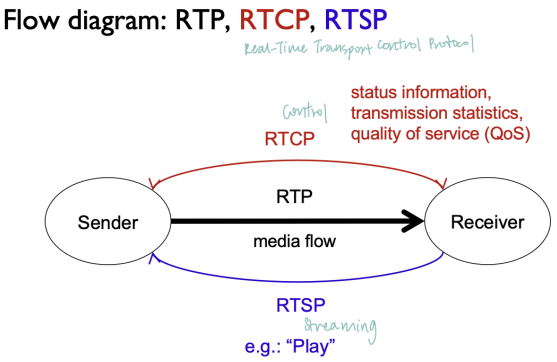
RTP header

- payload type (7 bits): indicates type of encoding currently being used. If sender changes encoding during call, sender informs receiver via payload type field

Type #	Indication
0	PCM mu-law, 64kbps
3	GSM, 13kbps
7	LPC, 2.4kbps
26	Motion JPEG
31	H.261
33	MPEG2 video
etc	

- sequence num (16 bits): detect packet loss, restore order
- timestamp (32 bits): sampling instant of first byte in this RTP data packet
  - audio: timestamp clock increments by one for each sampling period
  - e.g. each 125 us for 8KHz sampling clock
  - if application generates chunks of 160 encoded samples, timestamp increases by 160 for each RTP packet when source inactive. Time stamp clock continues to increase at a constant rate when source is inactive
- Synchronisation Source ID (SSRC) field (32 bits): identifies source of RTP stream. Each stream in RTP session has distinct SSRC
- (audio chunk + RTP header ⇒) RTP packets encapsulated in UDP segments, provides data in header and:
  - port num, IP addresses
- interoperability: if 2 VoIP applications run RTP, they may be able to work together





- RTP does not ensure timely data delivery or other QoS guarantees
    - Quality of service (QoS) is the use of mechanisms or technologies that work on a network to control traffic and ensure the performance of critical applications with limited network capacity
- RTP encapsulation only seen at end systems (not by intermediate routers)

Session Initiation Protocol (SIP)

- provides mechanisms for call setup
    - for callee to let callee know he/she wants to establish a call
    - caller, callee agree on media type, encoding
    - to end call
- determine current IP address of callee
    - maps mnemonic identifier to current IP address
- call management
    - add new media streams during call
    - change encoding during call
    - invite others
    - transfer, hold calls
- long term vision
    - all calls happen over the internet
    - people identified by names or email instead of phone number
    - can reach callee, no matter where caller roams, no matter what IP device callee is using

	RTP/RTSP/RTCP	DASH
Challenges	- Special-purpose server for media e.g. fine-grained packet scheduling, keep state (complex) - protocols use	- DASH is based on media segment transmissions (~2-10s) - by buffering a few segments at the client side, DASH does not provide low latency for

	RTP/RTSP/RTCP	DASH
	TCP&UDP transmissions (firewalls) - difficult to cache data (no “web caching”)	interactive, two-way applications (e.g. video conferencing)
Advantage	- short end-to-end latency ⇒ still used in WebRTC	- DASH divide media into small chunks (i.e. streamlets) - server is simple, i.e. regular web server (no state, proven to be scalable) - no firewall problems (use port 80 for HTTP) - standard (image) web caching works

Dynamic Adaptive Streaming over HTTP (DASH)

- Video-on-Demand (VoD) video streaming increasingly uses HTTP streaming
- HTTP streaming just GETs a whole video file from an HTTP server → can be wasteful, needs a large client buffer
- TCP

How DASH works

- Pull-based streaming → scheduling and computation is at the client side
  - The client is measuring the bandwidth and detecting when it changes, and then deciding which segment to download next.
  - As a very simple calculation, the client can compute bandwidth = (last segment size)/(download time).
- make playback adaptive
  - encode media into multiple different streamlet files (e.g. L, M, H quality version) for different bandwidths
- Web server provides a playlist (.m3u8/.mpd)
  - the playlist is a file in a specific format that lists all the available qualities and all the streamlets for each quality
  - Content Preparation
    - original media file needs to be split into streamlist
    - streamlets need to be transcoded into different qualities
- Data is encoded into different qualities and cut into short segments (streamlets, chunks)

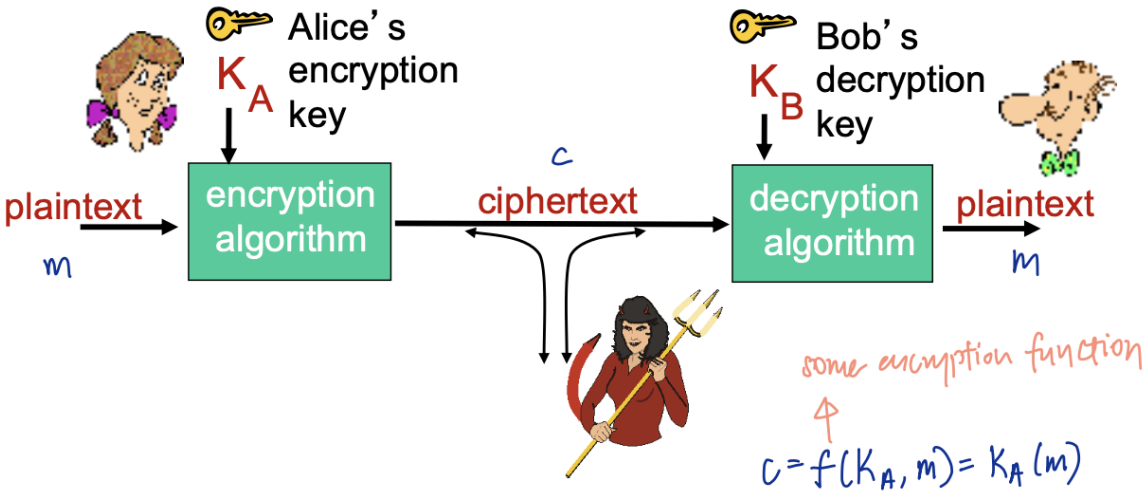
- Client first downloads MPD, which describes the available videos and qualities
- Client/player executes an adaptive bitrate algorithm (ABR) to determine which segment to download next
- applications: YouTube, Netflix, Facebook etc

L11: Network Security

Confidentiality	only sender, intended receiver should “understand” message contents - sender encrypts message - receiver decrypts message
Authentication	sender, receiver want to confirm the identity of each other
Message Integrity	sender, receiver want to ensure message is not altered without detection
Access & Availability	services must be accessible and available to users

What can an intruder do?

- eavesdrop: intercept messages
- actively insert messages into connection
- impersonation: can fake (spoof) source address in packet (or any field in packet)
- hijacking: “take over” ongoing connection by removing sender or receiver, inserting himself in place
- denial of service (DoS): prevent service from being used by others (e.g. by overloading resources)



m plaintext message

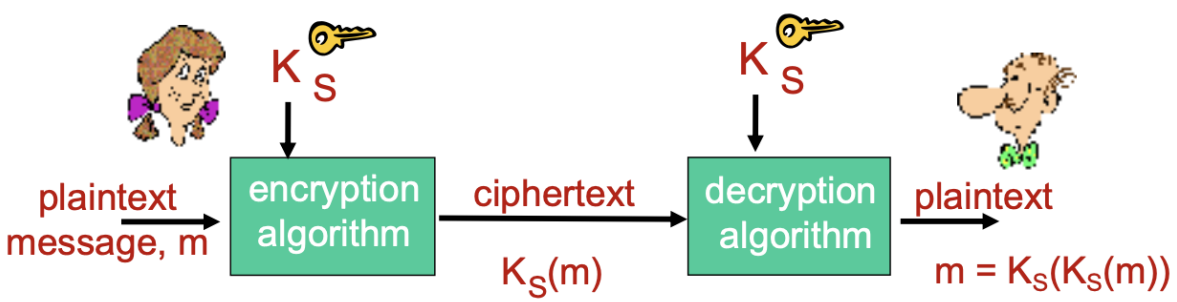
$K_A(m)$ ciphertext, encrypted with key K_A

$m = K_B(K_A(m))$

Breaking an Encryption Scheme

- Cipher-text only attack: intruder has ciphertext he can analyse
- Two approaches:
  - brute force: search through all 2^32 keys
  - statistical analysis
- Known-plaintext attack: intrude has plaintext corresponding to ciphertext
  - e.g. in monoalphabetic cipher, intruder can determine the pairings for each letter
- Chose-plaintext attack: intruder can get ciphertext for chosen plaintext

Symmetric Key Cryptography



- both end users share the same (symmetric key): K_S

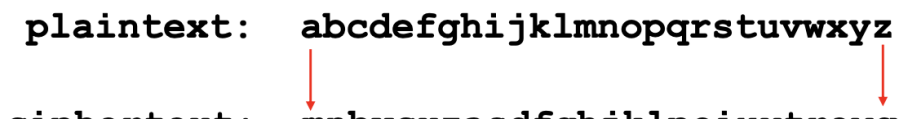
- Data Encryption Standard (DES): 56-bit symmetric key, 64-bit plaintext input
  - 56-bit-key-encrypted phrase decrypted (brute force) in less than a day
  - more secure: 3DES → encrypt 3 times with 3 different keys
- Advanced Encryption Standard (AES)
  - processes data in 128 bit blocks
  - 128, 192, or 256 bit keys

Substitution Cipher

- substituting one thing for another

Monoalphabetic Cipher

- substitute one letter for another
- encryption key: mapping from set of 26 letters to set of 26 letters

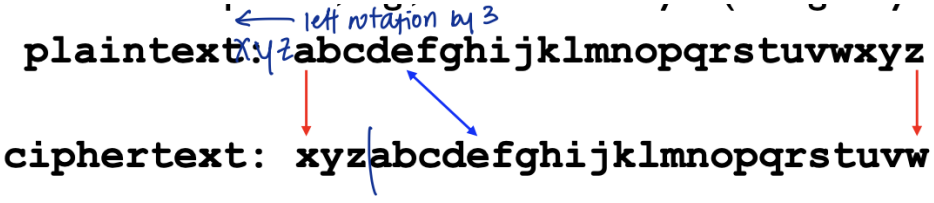


e.g.: Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc

however, frequency analysis (of each letter) can be used to determine mapping

Caesar's Cipher

- fixed shift of alphabet (e.g. left rotation by 3)
- encryption key: only need shift number

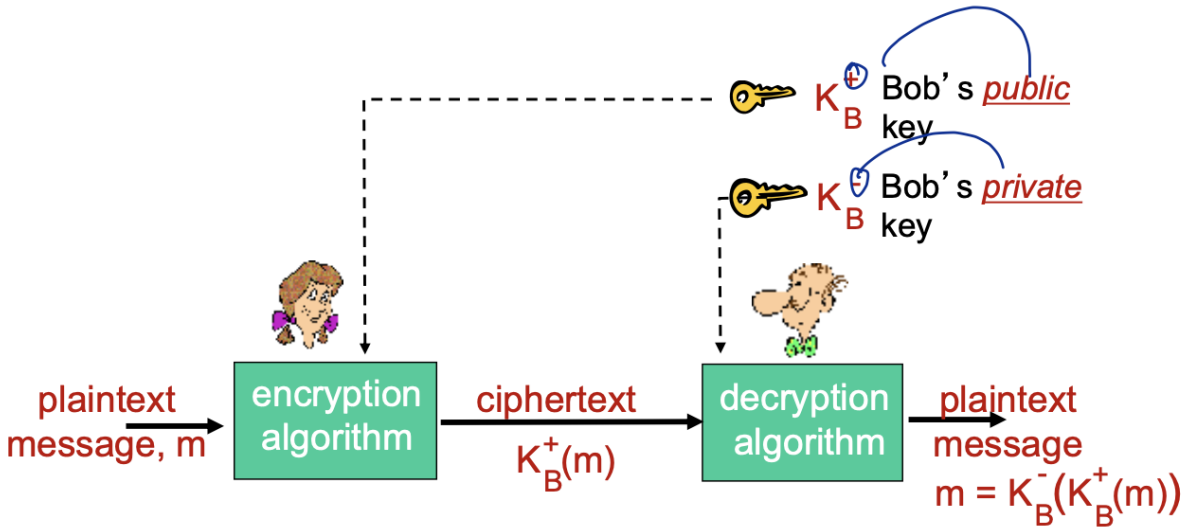


e.g.: plaintext: the quick brown fox
ciphertext: qeb nrfzh yoltk clu

n substitution ciphers: M1, M2, ..., Mn

- cycling pattern:
  - e.g. n = 4; $M_1, M_3, M_4, M_3, M_2;\ M_1, M_3, M_4, M_3, M_2; ...$
- For each new plaintext symbol, use subsequent substitution pattern in cyclic pattern
  - e.g. encrypt “dog”: d from M1, o from M3, g from M4
- encryption key: n substitution ciphers and cyclic pattern → key need not be just n-bit pattern

Public Key Encryption



1. needs $K_B^+(.)$ and $K_B^-(.)$ such that $K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$
2. given public key K_B^+, it should be impossible to compute private key K_B^-

[(a mod n) + (b mod n)] mod n = (a+b) mod n
[(a mod n) - (b mod n)] mod n = (a-b) mod n
[(a mod n) * (b mod n)] mod n = (a*b) mod n
$(a\ mod\ n)^d\ mod\ n = a^d\ mod\ n$

- message is a bit pattern that can be uniquely represented by an integer

- encrypt message = encrypt number

Rivest, Shamir, Adleman (RSA) Algorithm

1. choose two large prime numbers p, q. (e.g., 1024 bits each)
2. compute $n = pq$, $z = (p-1)(q-1)$
3. choose e (with $e<n$) that has no common factors with z (e, z are "relatively prime").
4. choose d such that $ed-1$ is exactly divisible by z. (in other words: $ed \bmod z = 1$).
5. *public* key is $\underbrace{(n,e)}_{K_B^+}$. *private* key is $\underbrace{(n,d)}_{K_B^-}$.

0. given (n,e) and (n,d) as computed above
1. to encrypt message m ($<n$), compute $c = m^e \bmod n$
2. to decrypt received bit pattern, c, compute $m = c^d \bmod n$

magic happens! $m = (\underbrace{m^e \bmod n}_{c})^d \bmod n$

Public Key Crypto + Symmetric Session Key

- public key crypto to establish secure connection (and exchange a symmetric key K_S)
- symmetric session key for encrypting data

Symmetric Key Crypto	Public Key Crypto
requires sender, receiver to know the shared secret key	- sender, receiver do not share secret key - public encryption key known to all - private decryption key known only to receiver
difficult to agree on key (particularly if they never "met")	

Digital Signatures

- verifiable, non-forgeable: receiver can prove that sender must have signed the document
- encrypt message with private key

- Suppose Alice receives msg m, with signature: m, $K_B^-(m)$.
- Alice verifies m signed by Bob by applying Bob's public key K_B^+ to $K_B^-(m)$ then checks $K_B^+(K_B^-(m)) = m$.
- If $K_B^+(K_B^-(m)) = m$, whoever signed m must have used Bob's private key.

Alice thus verifies that:

- Bob signed m
- no one else signed m
- Bob signed m and not m'

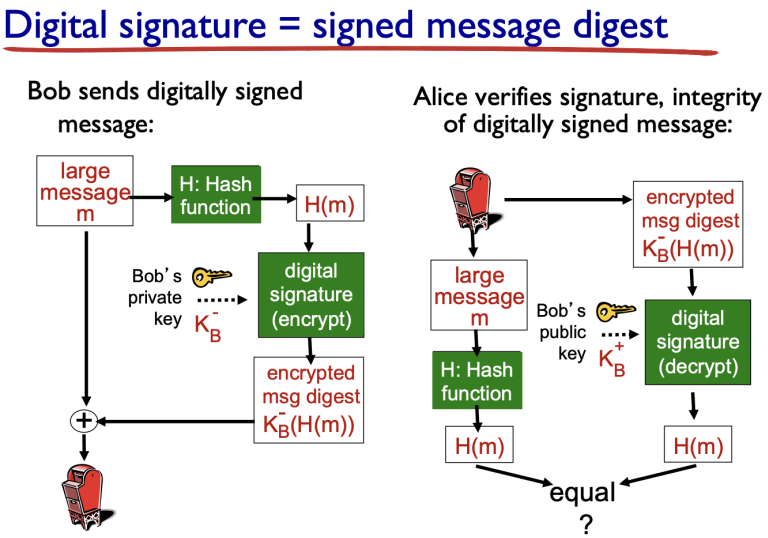
non-repudiation:

✓ Alice can take m, and signature $K_B^-(m)$ to court and prove that Bob signed m

Message digests (Hash)

Hash function properties:

- many-to-1
- produces fixed-size msg digest (fingerprint)
- given message digest x, computationally impossible to find m such that x = H(m)



- MD5 hash function widely used (RFC 1321)
  - computes 128-bit message digest in 4-step process
- SHA-1: 160-bit message
- SHA-256 is more common

Hash function, e.g. md5sum

- generate short, fixed-length outputs (or digests) of 128 bits
- small change can be detected
- e.g. used for password hashing

Summary of encryption methods

Encrypted hash of the message	digital signature of Alice used to prove her identity to Bob → Alice's private key is used
Encrypted message	message encrypted with the session key to ensure confidentiality of the message. Session key is a symmetric key
Encrypted session key	session key encrypted with Bob's public key → purpose is to share the session key with Bob

Public-key certification

- Certification Authority (CA): binds public key to particular entity E
- E (person, router) registers its public key with CA
  - E provides "proof of identity" to CA
  - CA creates certificate binding E to its public key, with CA's digital signature
- When Alice wants Bob's public key:
  - Alice gets Bob's certificate
  - apply CA's public key to Bob's certificate to get Bob's public key

▼ Suppose N people each want to communicate with $N-1$ other people. All communication between any two people, i and j, is visible to all other people but no other person should be able to decode their communication. In total, how many keys are required in this group if:

(a) Symmetric key encryption is used in each communication?
(b) Public key encryption is used in each communication?

(a) N(N-1)/2
(b) N public key + N private keys = 2N keys needed

▼ In the BitTorrent P2P file distribution protocol, the seed breaks a file into blocks, and the peers redistribute the blocks to each other. Without any protection, an attacker can easily wreak havoc in a torrent by masquerading as a benevolent peer and sending bogus blocks to a small subset of peers in the torrent. These unsuspecting peers then redistribute the bogus blocks to other peers, which in turn redistribute the bogus blocks to even more peers. Thus, it is critical for BitTorrent to have a mechanism that allows a peer to verify the integrity of a block, so that it doesn't redistribute bogus blocks.

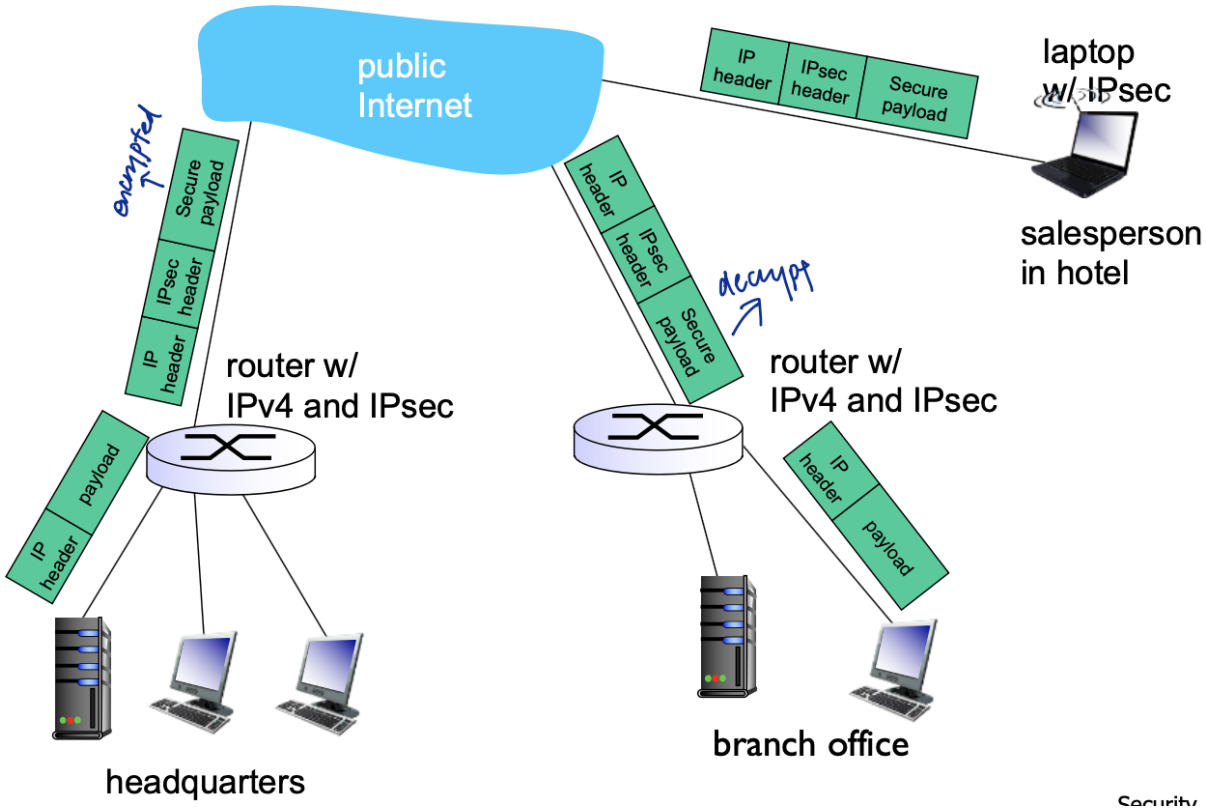
Assume that when a peer joins a torrent, it initially gets a .torrent file from a *fully trusted* source. Describe a simple scheme that allows peers to verify the integrity of blocks.

A file is broken into a number of blocks of identical sizes. For each block, a hash is calculated (e.g. using MDS or SHA-1). The hashes for all of the blocks are saved in the .torrent file.

When a block is downloaded, a peer calculates the hash of this block and compares it to the recorded hash in the .torrent file. If the two hashes are equal, this block is error free. Otherwise, the block is bogus and should be discarded.

Virtual Private Networks (VPNs)

- institutions often want private networks for security but separate routers, links, DNS infrastructures etc are costly
- VPN: institution's inter-office traffic is sent over public Internet instead
    - encrypted before entering public internet
    - logically separate from other traffic



When generating a 256-bit long symmetric key, i.e., a 256-bit number, to be used for encryption, which of the following provides the most security?

- A key should be completely random and not be derived from any known value that an attacker can guess in order to provide the highest security.
- We can take the computer name and IP address (in binary format) as a part of the key.
- We can take the MAC address (in binary format) of the computer as part of the key.
- We can take the current time (in binary format) and use it as part of the key.
- We can take the number of seconds (in binary format) since the computer booted as part of the key.

Summary of Web Request Process

e.g. accessing www.facebook.com

Step 1:

- On start-up, your PC needs an IP from DHCP server
    - DHCP request encapsulated in UDP segment (transport), then in IP datagram (network), then in Ethernet frame (link)
    - frame is broadcast on subnet
- DHCP server receives and processes this frame, starts negotiation with your PC for IP
- Intermediate switches learn your position when forwarding your frames

Step 2:

- DHCP server also tells your IP addresses of first-hop router (gateway router) and local DNS server
- After you type www.facebook.com, browser needs to know IP of this website from local DNS server
    - to know the MAC address of local DNS server, PC broadcasts the ARP query. Local DNS server replies with its MAC address
    - DNS query encapsulated in UDP segment, then in IP datagram, then in Ethernet frame, sent to local DNS server.
    - Local DNS server replies to your PC with IP of Facebook

Step 3:

- PC sends HTTP request to Facebook
    - TCP socket opened; 3-way handshake with Facebook server
    - HTTP messages exchanged after TCP connection setup
- Frames sent to first-hop router
- IP datagrams forwards from campus network to ISP SingNet
    - Private IP translated by NUS NAT router
    - IP datagram routed on the Internet using RIP or other routing protocols

Step 4:

- When Facebook is contacted
    - negotiate secure connection
    - HTTPS = HTTP + SSL/TLS

- Digital certificate of Facebook verified
- message encryption and authentication

PYP

1314s2

A node x is part of a network running distance vector routing protocol. x has three entries in its routing table:

Destination	(Min) Cost (dx(y))	Next Hop
w	4	w (x → w directly)
y	α	z (x → z → y)
z	β	w (x → w → z)

α and β are two unknown values (unknown to you, but not to x).

Assume that the distance vector routing protocol has converged and the minimum cost path from x to every other node has been found. We denote `c(x,y)` as the link cost between x and y, and `dx(y)` as the cost of the minimum cost path from x to y. The link cost is a positive integer.
We know that `c(x, w)` is 4, and `c(x, z)` is 10.

Min possible value of α? $\alpha \geq 11$

Since $d_z(y) \geq 1, c(x,z) = 10$ and

$\alpha = d_x(y) = c(x,z) + d_z(y) \geq 10 + 1 = 11$

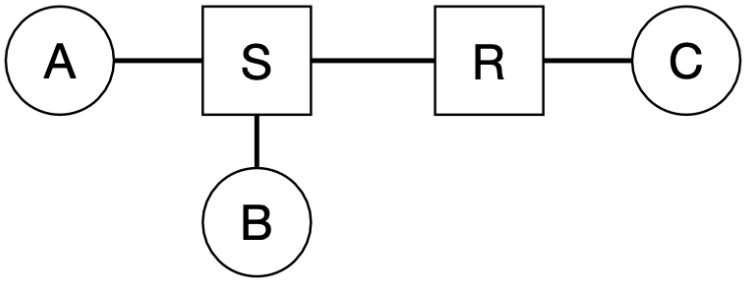
Max possible value of $d_w(z)$? $d_w(z) \leq 6$

Since $d_x(z) \leq c(x,z)$ (otherwise x would get to z directly instead of going through w) and therefore

$d_x(z) = c(x,w) + d_w(z) \leq c(x,z) = 10$

$d_w(z) \leq c(x,z) - c(x,w) = 10 - 4 = 6$

The diagram below shows a small network with five entities: hosts A and B are connected to a router R through a switch S. Host C connects to R directly. There is no other host, switch, or router in the network.



- A switch is a link-level device to interconnect between hosts. A switch interface does not have a MAC address, nor does it have an IP address.
- An IP address is associated with a network interface. Thus, R has two IP addresses.
- At the link layer, C only communicates with R, and A only communicates with B and R. Therefore,
  - the ARP table of C does not contain entries for A and B;
  - ARP table of A does not contain entry for C.

▼ **What is the maximum number of entries that could be in the switching table of S? Which entities do these entries correspond to?**

3: A, B, R

▼ **What is the maximum number of entries that could be in the ARP table of A? Which entities do these entries correspond to?**

2: B, R

▼ **What is the maximum number of entries that could be in the ARP table of C? Which entities do these entries correspond to?**

1: R

▼ **How many IP addresses are used in this network? Which entities do these addresses belong to?**

5: 2 for R and one each for A, B, C

Alice would like to send a message m to Bob over an insecure channel, where a malicious user Trudy can copy and delete any message exchanged between Alice and Bob. Furthermore, Trudy can also insert a message into this channel.

Bob and Alice wants to design a protocol such that the following two properties hold:

(I) Bob can ensure that m originated from Alice and not Trudy; and

(II) Bob can detect if m is a replay of an earlier message from Alice. Bob and Alice, however, do not care if Trudy eavesdrop on m or modify m.

The notations used to describe the protocol are:

- K_A^- and K_B^- are the private keys of Alice and Bob respectively;
- K_A^+ and K_B^+ are the public keys of Alice and Bob respectively;
- K_S is a shared symmetric key;
- s is the shared secret;
- H is a cryptographic hash function;
- + is the concatenation operator.

You can assume that H is known to everyone; CA can be trusted; K_S and s change for every message exchanged.
For each of the following protocols, indicate if it can ensure Property (I) only, Property (II) only, or both properties. If a property cannot be satisfied, explain.

▼ **Alice sends $K_A^-(m)$ to Bob.**

(I) only

- Trudy could buffer $K_A^-(m)$ and reply this message to Bob

▼ **Bob sends $K_B^-(K_S)$ to Alice. Alice sends $K_S(m)$ to Bob**

Neither.

- Trudy could get K_S using Bob's public key. Trudy can then delete Alice's $K_S(m)$ and insert her own $K_S(m')$. Bob cannot verify if m' is from Trudy or Alice.
- Since Trudy knows every K_S, Trudy can do the following: Trudy buffers `m` from Alice, and intercepts new K_S from Bob. Trudy then constructs $K_S(m)$, with new K_S, replaying m, and sends it to Bob.

▼ **Bob sends $K_A^+(s)$ to Alice. Alice sends $K_B^+(m+s)$ to Bob**

Both

- Only Alice can decode s and Bob can reject any message without correct s. Since s is used once, Bob can detect a replay attack.

▼ **Bob sends $K_A^+(s)$ to Alice. Alice sends m and H(m+s) to Bob**

Both

- Only Alice can decode s and Bob can reject any message without correct s. Since s is used once, Bob can detect a replay attack.

14s2

▼ **Suppose the propagation delay between furthest nodes is d and link rate is r. What is the minimal frame size L to ensure collision will always be detected in CSMA/CD protocol?**

2dr

Consider the worst case that *A* sends a frame to *B:*

- Just before this frame reaches B, B starts transmission.
- It takes *A* around RTT to receive the first bit from *B* and thus detect collision

▼ **Source and destination are connected by a single link that has packet loss probability of p. If at most k (re)transmissions are allowed until the source gives up, what is the probability that a packet would be successfully delivered to destination?**

$$(1-p)+p(1-p)+p^2(1-p)+...+p^{k-1}(1-p)=1-p^k$$

10 students want to communicate with each other confidentially (i.e., messages between any two students shouldn't be understandable to a third student).

▼ **In symmetric key cryptography, how many secret keys are needed in total?**

N(N-1)/2 = 10(9)/2 = 45

▼ **Suppose every student trusts the teacher. If a student needs to send a message to another, he first sends it to the teacher; the teacher then sends the message to the other student. The teacher is allowed to understand all messages sent to her. At a minimum how many keys are needed in total? State clearly if symmetric or public key cryptography is used.**

[Symmetric Key Crpto] 10

- Each student-teacher pair has 1 symmetric key

15s2

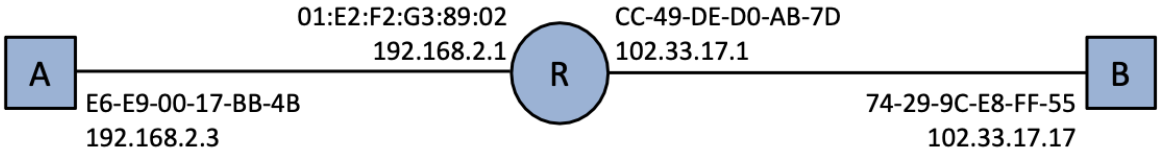
Which of the following field(s) in an IP datagram header will be changed by a router?

- TTL
- Checksum
  - The router must adjust the checksum if it changes the IP header (such as when decrementing the TTL)
- Identifier
- Upper layer protocol

Which of the following statement is FALSE?

- When a router receives an IP datagram with destination address 255.255.255.255, it must broadcast this IP datagram on all the interfaces except the interface this datagram is received.
- One of the benefits of segmenting a big chunk of data into smaller packets for transmission in the Internet is to lower end-to-end delay
- MSS specifies the maximum size of a TCP segment, exclusive of the size of the TCP header
- Hosts in the same subnet communicate with each other without intervening a router
- The max size of an IP datagram that can be transmitted over a link is restricted by the link MTU.

Host A and B are connected by a NAT-enabled router R as shown in the following diagram. Suppose A sends a packet to host B and B replies with a packet. What is the destination IP address and MAC address in the packet replied by B?

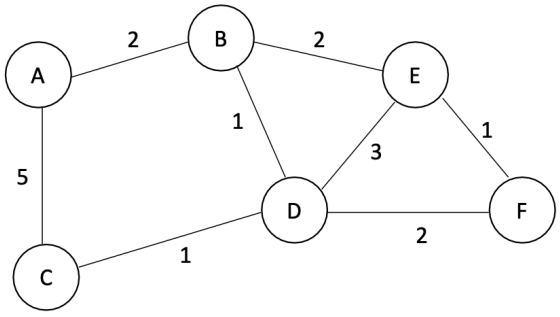


- Dest IP address: 102.33.17.1
- Dest MAC address: CC-49-DE-D0-AB-7D

A Go-back-N sender just receives an ACK packet with sequence number t. Before this ACK is received, sender's window is $[k, k + N - 1]$ where N is the window size. Suppose $k > N$, packets may be lost or corrupted but will not be reordered. What is the smallest possible value of t?

- k - 1
- May be a duplicate ACK from premature retransmission

Consider the network topology shown below. Each link is labelled with the cost (in dollars) of using that link. Every router runs distance vector routing protocol.



Initial distance vector table:

	cost to A	cost to B	cost to C	cost to D	cost to E	cost to F
from A	0	2	5	-	-	-
from B	2	0	-	1	2	-
from C	5	-	0	1	-	-

Final distance vector table:

	cost to A	cost to B	cost to C	cost to D	cost to E	cost to F
from A	0	2	4	3	4	5
from B	2	0	2	1	2	3
from C	4	2	0	1	4	3

Routers C to F are the gateway routers to the following subnets.

Derive the forwarding table of router A with the final distance vector table

Subnets	Gateway routers
137.132.58.128/28	D
137.132.89.0/26	C, D
137.132.80.128/25	C, F
137.132.82.0/24	E

To destination Net	Next hop
137.132.58.128/28	B
137.132.89.0/26	B
137.132.80.128/25	B
137.132.82.0/24	B

forwarding table for A

Assume all the links have the same transmission rate. Explain why the forwarding table for A leads to inefficient use of network bandwidth?

All traffic between (A, D), (A, E) and (A, F) is sent via B. The link between A and C is under-utilised while the link between A and B may be overloaded.

Mock Paper 1

[Public Key Cryptography] How Alice and Bob can ensure message confidentiality and integrity using only their private and public keys.

1. **Alice encrypts m with her private key to create digital signature $K_A^-(m)$.**
2. **Alice concatenates message with digital signature $m \oplus K_A^-(m)$, and encrypt the extended message with Bob's public key: $K_B^+(m \oplus K_A^-(m))$.**
3. **Alice sends $K_B^+(m \oplus K_A^-(m))$ to Bob.**
4. **Bob decrypts the received message using his private key: $K_B^-(K_B^+(m \oplus K_A^-(m))) = m \oplus K_A^-(m)$.**
5. **Bob then uses Alice's public key to derive message from digital signature: $K_A^+(K_A^-(m)) = m'$**
6. **If $m = m'$, message integrity is preserved.**
7. **Because message is encrypted during transmission, message confidentiality is preserved.**

(Another approach is for Alice to send $K_B^+(m) \oplus K_A^-(K_B^+(m))$)

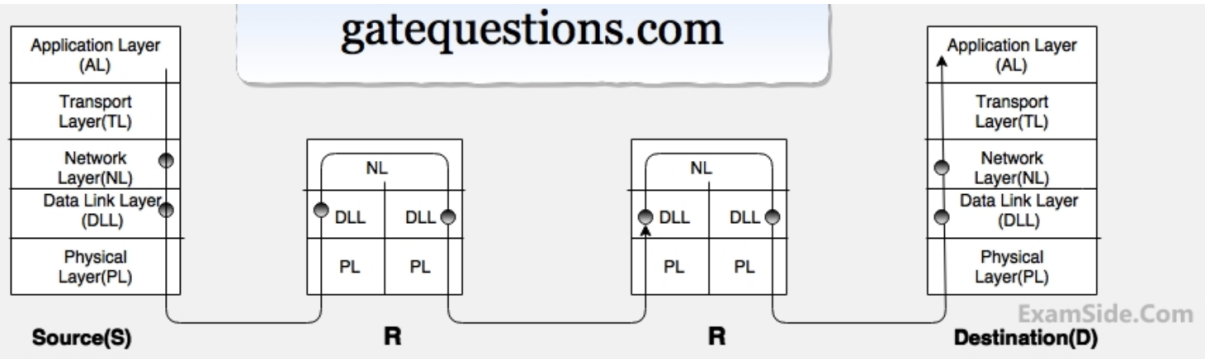
Mock paper 2

Which of the following statement about IP header is TRUE?

- The source and destination port numbers in the IP header determine which application on the receiving host will process the datagram.
- The TTL field in the IP header determines the time period within which the source IP address is valid.
- The 16-bit identifier field in the IP header is not changed during IP fragmentation.
- The checksum field in the IP header allows the receiver to check if the IP header or payload is corrupted.
    - checksum is only for detection of corruption in IP header
- The protocol field in the IP header determines which link layer protocol should be used to transmit the datagram.

Each network interface card has a MAC address. Why not simply use this MAC address for routing of packets on the Internet?

- An IP address logically comprises two parts: network prefix and host ID. This is designed to facilitate routing: routers check prefix and deliver a packet to an aggregated destination network.
- If MAC address is used instead, hierarchical routing cannot be achieved.
- e.g. MAC address is burnt in ROM and usually cannot be changed. When people carry their laptops around the world, devices in a subnet won't have common prefix in MAC addresses. This makes routing difficult as routers have to remember routing for every single MAC address.



- From above given diagram, it's easily visible that packet will visit network layer 4 times(once at each node [S, R, R, D]) and packet will visit Data Link layer 6 times (One time at S and one time at D, then two times for each intermediate router R as data link layer is used for link to link communication.)
- Once packet reaches to R(router) then it goes up from physical layer to DLL to Network layer and when packet coming out of router in order from Network layer to DLL to Physical layer.
- Remember: You can see, At router there are two Physical Layer and two Data link Layer.The reason is,on one side of the router there could be one network and other side of the router there could be other network.
- For ex.- One side there could be Ethernet and on other side there could be Token Ring.So,One DLL will understand the language of Ethernet and other DLL will convert the packet so that it could be read by the Token Ring.
- That is why two DLL and two PL is needed at the Router.