CS2105 Introduction to Computer Networks

Basics

- **Circuit switching**:
  Call setup required
  Circuit-link (guaranteed) performance
  Circuit segment idle if not used by call (no sharing)
- **Packet switching**:
  Share network resources
  Resources used on demand
  Excessive congestion is possible
  Sender breaks message into pkts; receiver reassembles them
- **Processing delay**: Check bit errors; determine output link
- **Queuing delay**: Waiting in queue for transmission
- **Transmission delay**: Time taken to push bits onto link
- **Propagation delay**: Time for bits to travel in link
- **End-to-end packet delay**: Time for packet to travel from source to destination
- **Throughput**: Bits transmittable per unit time for end-to-end communication

Both average & instantaneous throughputs do NOT depend on packet size

Application *message* **Transport** *segment*
Network *datagram* **Link** *frame*

Application Layer

SSH : 23 [TCP]

Common Protocols

App. Protocol	Tpt. Protocol	Port
HTTP	TCP	80 (default)
HTTPS	TCP	443 (default)
DNS	UDP	53
SMTP	TCP	25
DHCP	UDP	67 (svr) 68 (client)
RIP	UDP	520

Hypertext Transfer Protocol

2 * RTT + file transmission (per object)

- HTTP 1.0 closes connection after transmitting single object
- HTTP 1.1 uses persistent connection by default (possibly with pipelining)
- **HTTP request message**: (terminates with double CRLF)

```
GET /cs2105/demo.html HTTP/1.1
Host: www.comp.nus.edu.sg
User-Agent: Mozilla/5.0
Connection: close
Cookie: name=value; name2=value2; name3=value3
```

- **HTTP response message**:

```
HTTP/1.1 200 OK
Date: Thu, 15 Jan 2018 13:02:41 GMT
Content-Type: text/html
Content-Length: 150
Set-Cookie: name=value

data data data...
```

length of body

- **Conditional GET**:
  `If-Modified-Since:  Thu, 15 Jan 2018 13:02:41 GMT`
  Server may reply with `304 Not Modified`

Domain Name System

runs over UDP

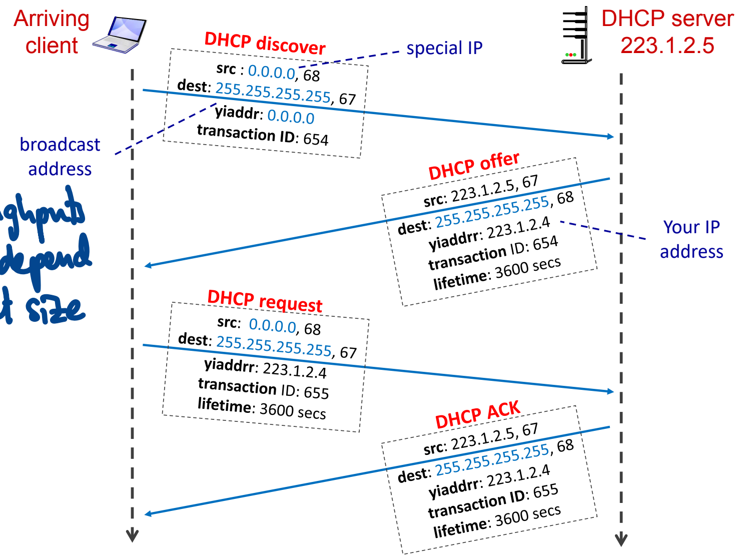
- Mapping between hostname and IP address (and others) are stored as resource records (RR)
- RR format: `(name, value, type, ttl)`

`type`	`name`	`value`
`A`	hostname	IP address
`NS`	domain (`nus.edu.sg`)	hostname of authoritative NS
`CNAME`	alias name	canonical name

- 13 root servers globally that answer `NS` queries for TLDs
- **Local DNS server** caches mapping and acts as proxy
  ↳ expires only after TTL

Dynamic Host Configuration Protocol

runs on UDP



- May also provide other network information:
  - first-hop router, local DNS server, subnet mask
  (default gateway)

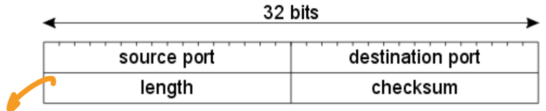
Transport Layer

TCP Header : 20 B
UDP Header : 8 B

TCP vs UDP

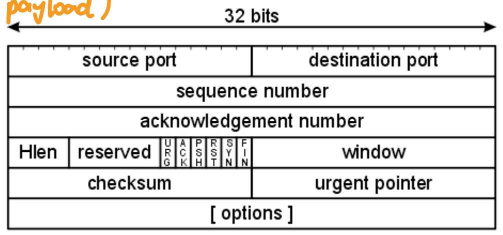
- **Transmission Control Protocol**:
  Reliable transport
  Flow control (sender won't overwhelm receiver)
  Congestion control (throttle sender in overloaded network)
  Does not provide timing, minimum throughput guarantee, security
- **User Datagram Protocol**:
  Unreliable data transfer
  Does not provide reliability, flow control, congestion control, timing, minimum throughput guarantee, security
- **Socket**: Interface between application and transport layers
  TCP uses a *stream socket*
  UDP uses a *datagram socket*
- TCP and UDP ports are distinct; port num may be reused
- TCP creates a new socket for each client (using the same server port), but uses client IP and client port to distinguish clients
- **Checksum**: 1's complement sum of 16-bit integers = 0b1111111111111111
  To compute checksum, remember to invert the sum (*)
- **Multiplexing**: When receiving packet from network layer, TCP/UDP must read transport header to decide which socket to deliver the message to (de-multiplexing); when sending messages from application layer, TCP/UDP must combine packets from different messages into the same network interface (multiplexing)
  - **UDP connectionless de-multiplexing**: decide using destination port only
  - **TCP connection-oriented de-multiplexing**: decide using (src IP addr, src port, dest IP addr, dest port)

UDP header format



size of segment

(Header (8B) + payload) TCP header format



Reliable Data Transfer (Stop & wait protocol)

- **rdt 1.0**: Perfectly reliable
- **rdt 2.0**: May corrupt packets
  Stop-and-wait protocol; receiver sends ACK or NAK back
  Fatal flaw if ACK is corrupted, because sender will resend packet and receiver will treat it as new packet
  seq # to detect duplicate packets
- **rdt 2.1**: To fix rdt 2.0, add 1-bit sequence number to each packet; receiver can now detect and discard duplicate packet (but must still send ACK for the duplicate packet)
- **rdt 2.2**: Same functionality as rdt 2.1, but is NAK-free; receiver ACKs sequence number of last received packet
- **rdt 3.0**: May corrupt packets, may lose packets, may incur arbitrary long packet delay
  Sender waits "reasonable" amount of time for ACK, and retransmits if ACK is not received before timeout; sequence number included in both data and ACK just like rdt 2.2
  ↳ if duplicate ACK at sender then does nothing

Pipelining

— sending multiple yet-to-be-Ack packets

- **Go-back-*N***: no buffer ; cummulative Ack
  Sender:
  - Up to N unACKed packets in pipeline
  - k-bit sequence number
  - "sliding window" to keep track of unACKed packets
  - timer for oldest unACKed packet
  - * on timeout(n) retransmit packet n and all subsequent packets in the window

  Receiver:
  - Only ACK packets that arrive in order
  - Discards out of order packets and ACK the last in-order sequence number *("cumulative ACK")*
- **Selective repeat**: Buffer ; individual Ack
  Receiver individually ACKs all correctly received packets; buffers out-of-order packets as needed
  Sender maintains timer for *each* unACKed packet; if timer expires, retransmit only that unACKed packet

TCP Reliability

Max Segment Size = 1460B (app data excl TCP header)

- **TCP sequence number**: "byte number" of first byte of data in a segment
  ↳ data (excl TCP header)
- **TCP acknowledgement number**: sequence number of next byte expected *("cumulative ACK")*
- **Maximum segment size**: maximum number of *data* bytes
  Maximum packet size: includes header bytes
- **TCP delayed ACK**: Wait up to 500ms for second segment; use one ACK for two segments only
- **Dynamic TCP timeout**:
  $SampleRTT :=$ RTT of new packet
  $EstRTT \leftarrow (1-\alpha) \times EstRTT + \alpha \times SampleRTT$
  (typically $\alpha = 0.125$)
  $DevRTT \leftarrow (1-\beta) \times DevRTT + \beta \times |SampleRTT - EstRTT|$
  (typically $\beta = 0.25$)
  $TimeoutInterval \leftarrow EstRTT + 4 \times DevRTT$
- **TCP fast retransmission**: If 3 duplicate ACKs (i.e. 4 in total) are received, next segment is treated as lost and thus retransmitted immediately
- Maintains single timer and resends oldest unACKed packet on timeout; timer started only when prev. ACK is received

Network Security

K_S: session key
K_A^+: public key K_A^-: private key **2 keys per user**

- **Integrity / Authenticity**: Bob can verify Alice is sender
  Message authentication code: Send $H(m + K_S) \oplus m$
  Digital signature: Send $K_A^-(m) \oplus m$
  Digital sign.: Bob can prove to third party Alice is sender
  Signed message digest as digital sign: Send $K_A^-(H(m)) \oplus m$
- **Confidentiality**: Send K_B^+(<everything from above>)
- **Hybrid**: Send $K_B^+(K_S) \oplus K_S(m \oplus K_A^-(H(m)))$

Network Layer

For Qns involving end-to-end delay w varying bandwidth of links

(*) $\left(\frac{L}{R_1} + \frac{L}{R_2} + \dots + d_{prop}\text{ of each link}\right) + (N-1)\frac{L}{\min\{R_1, R_2, \dots\}}$

IP Addressing

- 172.16.0.0/12 → subnet mask starts with 12 '1's
  first: 172.16.0.0 (subnet); last: 172.31.255.255 (broadcast)
  - all other addresses are usable
- **Valid subnet masks**:

Subnet size	256	128	64	32	16	8	4	2	1
Subnet mask	0	128	192	224	240	248	252	254	255

- **Longest prefix match** is used to determine next hop from router forwarding table
- **Special IP addresses**:

0.0.0.0/8	Local subnet (non-routable)
127.0.0.0/8	Loopback
255.255.255.255/32	Broadcast (within subnet)
10.0.0.0/8 172.16.0.0/12 192.168.0.0/16	Private

- Routers have one IP address per subnet

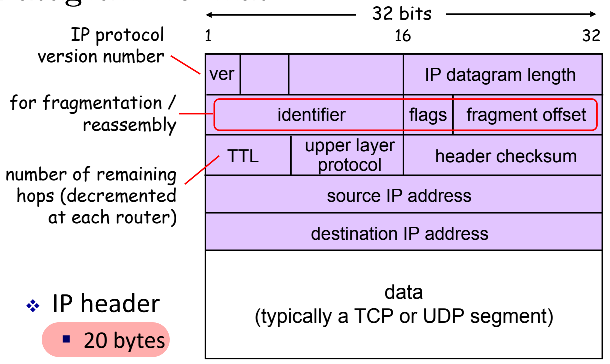
Network of Networks

- The Internet is a “network of networks” – a hierarchy of autonomous systems (AS)
- **Intra-AS routing**: RIP, OSPF; **Inter-AS routing**: BGP
- “link-state” algorithms – all routers have complete knowledge of network topology and link cost; compute least-cost path using Dijkstra’s algorithm
- “distance vector” algorithms – routers know physically-connected neighbours and link costs to them, and exchange and update “local views” periodically; compute using Bellman-Ford equation *(cost = total distance)*
- **Routing Information Protocol** (RIP) implements “distance vector” (DV) algorithm, measuring *hop count*
  - Entries in routing table are aggregated subnet masks (so we are routing to destination subnet)
  - Exchange routing table every 30 secs over UDP port 520
  - If no update for 3 minutes, assume neighbour has failed

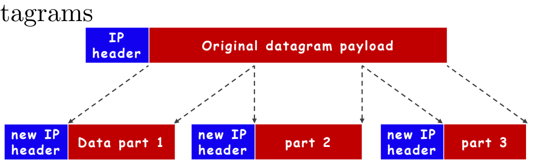
Network Address Translation

- Maintains mapping between (external IP Address, external port) and (destination (LAN) IP address, destination port)

IP Datagram Format



- IP datagram length includes IP header
- Header checksum only for header bytes; 16-bit 1’s complement sum (just like TCP)
- Different links have different maximum transfer unit (MTU) *(MTU includes IP header)*; routers may fragment IP datagrams



- Total data transferred increases due to extra IP headers
- Destination host will reassemble the packet
- **Header field changes for fragmentation**:
  IP datagram length is set to fragment size
  More frags. (MF) flag is set for all fragments except the last
  Fragment offset is the fragment offset in the original data payload, measured in 8-byte units
  Header checksum is recomputed

Internet Control Message Protocol (ICMP)

- Used to communicate network-level information: error reporting, echo request/reply (ping)
- When TTL for a packet is zero, the packet is discarded and an ICMP message is sent to source address

Link Layer

Required services
- Framing: Encapsulate datagram to frame, add header/trailer

Optional services
- Link access control: If multiple nodes share a single link, need to coordinate which nodes can send frames at a certain point in time
- Reliable delivery: Often used on error-prone links (e.g. wireless) - Error detection - Error correction

- Link + physical layer is implemented in hardware in network adapter or on a chip
- **Single bit parity** can detect single-bit errors
- **Two-dimensional bit parity** can detect and correct single-bit errors; can detect two-bit errors
- **Cyclic Redundancy Check (CRC)**:
  Used widely in practice (on Ethernet & Wi-Fi)
  - D: data bits (dividend)
  - G: generator of $r+1$ bits, pre-agreed (divisor)
  - R: resultant CRC checksum (remainder)
  Bitwise XOR division is used

  Sender computes R and sends (D, R)
  Receiver divides (D, R) by G and checks if remainder is zero

Multiple Access Protocols

- Required in broadcast links
  - multiple nodes connect to a shared broadcast channel
  - when a node transmits a frame, every other node receives a copy
  - if two nodes transmit simultaneously, frames *collide* and none would be correctly read
- Categories:
  Channel partitioning: divide channel into smaller “pieces” (e.g. time slots, frequency); each node exclusively allowed to transmit in given piece (unused pieces go idle)
  Taking turns: nodes take turns to transmit (but can cooperatively forfeit turn if there is nothing to transmit)
  sends upto max # frames
  won't get full bandwidth because of polling overhead
  Random access: channel is not divided and collisions are possible; focus on “recovering” from collisions
  can detect and recover
- **Time division multiple access (TDMA)**:
  Channel partitioning by fixed-length time slot
  Slot long enough to send one frame
- **Frequency division multiple access (FDMA)**:
  Channel partitioning by frequency band
- **Polling**:
  Taking turns; master node “invites” slave nodes to transmit in turn
  (polling overhead; single point of failure of master node)
- **Token passing**:
  Control token is passed from one node to next sequentially (token overhead; single point of failure (lost token))
- **Slotted ALOHA**: → Transmit frame at start of slot, if collision then retransmit w prob p.
  Assumptions:
  - All frames of equal size
  - Time divided into slots of equal length (1 slot = 1 frame)
  - Nodes start to transmit only at the beginning of a slot
  Operations:
  - Listens to the channel while transmitting (detect collision)
  - If collision, re-transmit frame in each subsequent slot with probability p until success

- **Pure (unslotted) ALOHA**:
  No slots; transmit immediately
  but if collision,
  ① wait for one frame time
  ② retransmit w probability p
  - Chance of collision increases
  → collision window doubled
- **Carrier Sense Multiple Access (CSMA)**:
  - Sense the channel before transmission; don’t interrupt ongoing transmission
  - Collisions may still occur due to to propagation delay and propagation distance
- **CSMA/CD (Collision Detection)**: $2 \max(d_{prop}) \leq d_{trans}$
  Abort transmission when collision is detected
  Minimum frame size is usually specified as collision may not be detected for overly small frames due to propagation delay *(e.g. Ethernet requires minimum frame size of 64 bytes)*

  Has “Hidden node problem”: due to propagation distance, collisions at receiver may not be detectable by source
  ↳ resend using exponential method
- **CSMA/CA (Collision Avoidance)**:
  Receiver needs to return an acknowledgement if frame is received successfully *(e.g. Wi-Fi)*

MAC Addressing

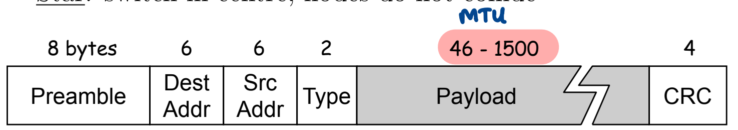
- 48 bits long
- Permanently assigned to network interface card (NIC)
- Each network node will only process frames that are addressed to its MAC address (or the broadcast address `FF-FF-FF-FF-FF-FF`)

Address Resolution Protocol (ARP)

- Resolves IP address to MAC address
- Each IP node has an ARP table which stores the mapping of IP address to MAC address (and TTL) of other nodes in the same subnet
- If the next hop node is not yet in the ARP table, an ARP query packet (with required IP address) is broadcasted to subnet; node with correct IP address will respond with its MAC address, sent back to source MAC address

Ethernet

- **Topology**
  Bus: all nodes can collide with each other
  Star: switch in centre, nodes do not collide



- Preamble: `10101010 10101010 10101010 ··· 10101011`
  Provides bit-level syncing, not part of 64-bit min. frame size
- Type: Higher-level protocol; 0x0800 for IPv4
- **Ethernet CSMA/CD algorithm**: ✱
  1) If channel idle, start transmitting immediately. Otherwise wait until idle.
  2) If collision while transmitting, abort and send jam signal. Then do binary back-off: after m^{th} collision, choose K at random from range $[0, 2^m)$, and wait $512 \times K$ bit times, then go back to step 1.

  Binary back-off aims to adapt re-transmission attempts to estimated current load

- **Ethernet switch**: → Always ensure switch topology is loop-free. Else, buffer will overflow.
  - Hosts have dedicated connection to switch; switch buffers frames (store-and-forward) and is full duplex (simultaneous bidirectional transfer)
  - CSMA/CD protocol is used even though no collisions
  - Maintains switch table – maps MAC address to interface (and TTL); if destination interface is known then frame is forwarded only to that link; if destination is not known then frame is broadcast
  - Switch learns source MAC address when frame is sent through it
  - Nodes do not need to know about the presence of the switch (switch is transparent to nodes)

Network Security

- Principles of Security
  ① confidentiality
  ② Integrity
  ③ Authentication
- Types of Cryptography
  - Symmetric key ($K_A = K_B$)
  - Asymmetric key ($K_A \neq K_B$)
- Encryption [confidentiality]
  - DES (Sym)
  - AES (Sym)
  - RSA (public)
  100 x faster — So choose K_s then pass via RSA. Then continue using DES. K_s is called session key

Hash function:
- If a function $H(.)$ that takes an input m and produces fixed-size msg digest (*fingerprint*) → Cannot get m from hash
  ex SHA-1, MD5

- Message Integrity
  - sender, receiver want to ensure message *not altered* (in transit, or afterwards) *without detection*

  eg ↳ Message Authentication Code (MAC)
  - *The sender and receiver share a “Authentication key” s*
  - *To ensure Message integrity:*
    - Send $(m, \underbrace{\mathrm{H}(m+\mathrm{s})}_{\text{Message Authentication Code}})$

Authentication
- sender, receiver want to *confirm identity* of each other

eg ↳ Digital Signature — A → B
- use Alice private key to encrypt
- Then let Bob or anyone else use Alice's public key to verify
- can be optimized further by hashing message
↳ Message Authentication Code (MAC) — cons: only A and B can verify