CS2105 Finals Cheat Sheet

Delay and Propagation

D_{prop} - Time taken to propagate one bit over a link. Expressed as distance **d** over propagation speed **d/s.** For pipelined transfer, this delay only occurs once as consecutive data stream is sent.

- $D_{prop} = \frac{1}{2} \times RTT$

D_{trans} - Time taken to transmit data onto a link. Expressed as L bits of data over R which is the transmission rate of the link in bits-per-sec. This delay occurs for all data to be sent regardless of pipelining.

D_{proc} - Delay needed for processing of data at the node (i.e checksum, authenciation)

D_{queue} - Delay in queuing the packet at the buffer of the node.

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

Throughput - the average time taken to send all bits of a file. Expressed as F/T where F is the file size, T is the time taken in total where $T = D_{trans} + D_{prop}$

When multiple links are involved, consider the bottleneck of the link with lowest transmission rate

Link Utilization Rate: $\frac{WS \times D_{trans}}{RTT + D_{trans}}$ where WS is window size for a sliding window protocol

Application Layer

FTP - uses port 21 control channel to establish connection, port 22 to send data

DNS(Domain Name Server)

1. Host first checks it's web cache for the destination IP address given a host name. If not present, queries the local DNS server.
2. If local DNS server does not have the mapping for required IP address, it queries the root DNS server
3. Root DNS server then queries the Top-Level Domain (TLD) DNS servers, which return the authoritative DNS servers for the domain of the host name the host needs
4. The authoritative DNS server is queried to get the hostname to IP address mapping of the required destination.
5. At each level, if cache of DNS server does not have the required mapping, the query that it makes will be put into cache.

DNS queries can be recursive or iterative:

- Recursive: Each query is made by the next node in sequence.
- Iterative: local DNS server gets the result at each level of query and queries the next level of hierarchy.

DNS can be queried using the dig command.

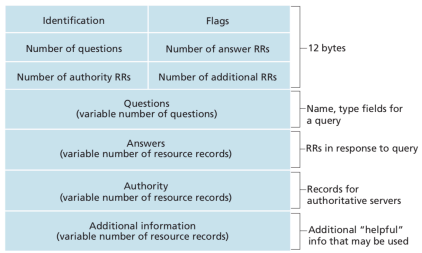
Each entry in DNS server is a 4 tuple called a resource record.

(Name, Value, Type, TTL)

Type - **A** - Authoritative, contains IP address of hosts to hostname mapping

MX - Mail Server, contains mapping for IP addresses of mail servers

NS - Name Servers, contains mappings for hostnames of authoritative DNS servers to it's domains



DNS runs on top of UDP

Question - contains information about query that was made

Answer - contains resource record about the original name queried

HTTP

Request Header

Method

POST - for writing data to remote server

GET - for retrieving objects from web server

For conditional GET and HEAD messages, body can be empty

Response Header

Status Code:

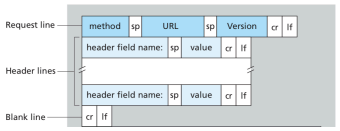
2XX - Successful Request

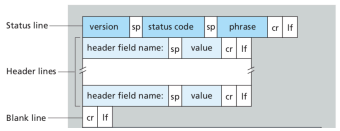
3XX - Object location wrong

4XX - Object not found

5XX - Server error

Common codes: 200 - OK, 301 - File moved permanently, 404 - File not found





For web pages that have multiple objects linked to it, the web page must first be retrieved(1 RTT for webpage) before the rest of the objects.

HTTP Version - 1.1/2.0 Persistent vs 1.0 Non-persistent by default

- For persistent connections, the TCP connection remains open for a while after it is first established, allowing multiple objects to be sent over the same connection for a period of time - Only 1 handshaking RTT for whole send process
- For non-persistent connections, a connection is opened for each object sent, and closed after the object is sent successfully - 1 RTT per object sent

Non-persistent, Non-pipelined - 2 RTT per object sent

Persistent, Non-pipelined - 1 RTT for connection, 1 RTT for webpage, 1 RTT for each linked object

Persistent, Pipelined - 1 RTT for connection, 1 RTT for webpage, 1 RTT for rest of objects linked to webpage sent continuously

Conditional GET

- When a host requests a webpage, the proxy server/web server checks the requested webpage in it's cache. If present, it checks the LAST MODIFIED header in the cached webpage against the IF MODIFIED SINCE in request message. If the two dates are the same(i.e no changes in the object), the proxy server forwards the cached object with the following header:

```
HTTP/1.1 304 Not Modified
```

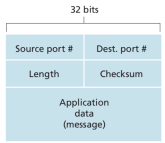
- Else, if the two dates are different/ object is not in cache, the proxy server requests the object from the origin server, and caches it when received before forwarding to the host.

Transport Layer

UDP

- Connectionless - no handshaking required before sending data
- Unreliable - data sent without acknowledgement of integrity of data received
- Used for transfer of data that requires low latency, or used to do load balancing on servers as connection establishment can be taxing

UDP Segment



- UDP uses a 4 tuple to identify sender and destination: (Source IP, Source Port, Destination IP, Destination Port)
- Uses checksum to do error detection in bit flips

Checksum - All the words in the data are added up, and the 1s complement of the sum is taken to be the checksum

At the receiving end, the sum of the words and the checksum is added together to give a string of 1s → has additional overflow

Words of the checksum include the header data as well as content data, as header data could be corrupted as well

Reliable Data Transfer Protocol

Stop-And-Wait Protocols

RDT 1.0 - Receives data and send to upper layer as it is. Cannot handle data corruption, packet loss and reordering

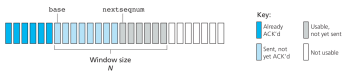
RDT 2.0 - Receiver sends NAK if packet is corrupted and discards it, else sends an ACK and sends data to upper layer. Can handle data corruption, but not packet loss and reordering. Also has problem of duplicate NAK/ACK packets that are unidentifiable by sequence

RDT 2.1/2.2 - Each packet has a sequence bit that alternates between 1 and 0. Sender sends next packet on receiving ACK with the same number it is waiting on, else resends the current packet. Receiver sends ACK X if it receives the packet X successfully, else it sends ACK Y

RDT 3.0 - Sender now has a timeout interval, and resends the current packet after the timeout interval elapses without response from the sender. Uses the same acknowledgment scheme as RDT 2.2, except that it discards ACKs with sequence bits not matching the one it is waiting for. Can handle data corruption, packet loss and packet reordering to a certain degree.

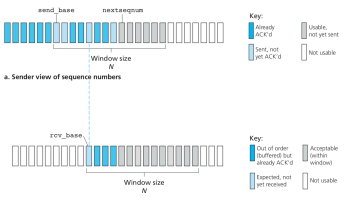
Pipelined Protocols

Go-Back-N



- Sender maintains a window, and sends all un-ACKed packets in this window at once.
- Cumulative ACK: ACK X received by the Sender indicates that all packets X and before received successfully and shift it's send base to X + 1
- Single Timer for whole window: upon timeout, all packets numbered from send base to nextseqnum sent
- Sender's window does not tell us everything about receiver's window, data/ACK packets sent could be en-route between sender and receiver

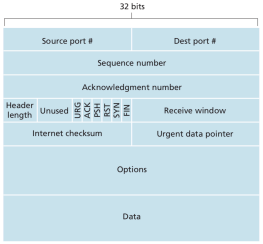
Selective Repeat



- Sender and receiver maintains separate windows.
- Separate Timer: Each packet has it's own timeout interval and is resent and ACKed independent of other packets.

TCP

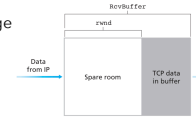
- Connection-oriented: connection established before data can be sent
- Reliable: Guarantees in-order sending of data without corruption or loss
- Sequence number: byte-sequence start number of the data sent. Advanced by the sender
- Acknowledgement number: byte-sequence number of the next packet expected. Advanced by the receiver
- Note that the sequence number notes start number, and not the size



Fast Retransmit - Packet X resent immediately 3 duplicate if ACK X received

rwnd - Receiver's window that dictates how much space left in buffer to receive data. Given by the following formula:

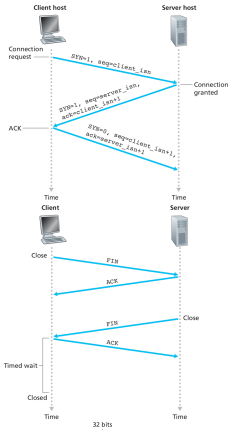
rwnd = RcvBuffer – [LastByteRcvd – LastByteRead]



- Upon *rwnd* = 0, sender continually sends empty 'probe' packets to the receiver until the received ACK contains a non-zero rwnd - indicates that receiver can now receive data

Event	TCP Receiver Action
Arrival of in-order segment with expected sequence number. All data up to expected sequence number already acknowledged.	Delayed ACK. Wait up to 500 msec for arrival of another in-order segment. If next in-order segment does not arrive in this interval, send an ACK.
Arrival of in-order segment with expected sequence number. One other in-order segment waiting for ACK transmission.	Immediately send single cumulative ACK, ACKing both in-order segments.
Arrival of out-of-order segment with higher-than-expected sequence number. Gap detected.	Immediately send duplicate ACK, indicating sequence number of next expected byte (which is the lower end of the gap).
Arrival of segment that partially or completely fills in gap in received data.	Immediately send ACK, provided that segment starts at the lower end of gap.

3-Way Handshake



- Client sends a packet with SYN bit set to 1 to server(SYN)
- Server acknowledges synchronization, and sends a packet to synchronise with the client with SYN and ACK bit set to 1(SYN-ACK)
- Client acknowledges the synchronisation and sends a packet with ACK bit set to 1(ACK). This packet can contain payload.

Connection Teardown

- Client sends special packet with FIN bit set to 1 to server (FIN)
- Server replies with an ACK. Server then sends a packet with it's FIN bit set to 1
- Client replies with an ACK. Resources for the connection between client and server is now torn down.

Network Layer

IP Datagram Header

TTL - Specifies how long a packet can exist before being discarded. Discarded at endpoint at TTL = 0

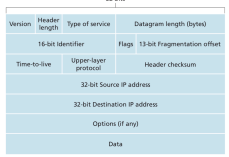
Datagram Fragmentation

Maximum Transmission Unit(MTU) - Max size of a datagram that can travel through link. Includes header

Maximum Segment Size - Maximum size of a TCP segment payload. Excludes header

If MTU of IP datagram exceeds the MTU supported by a link, the datagram is fragmented.

- Each fragment has it's own header data, and contains data = MTU - header
- Each fragment's header contains the same ID flag field, and also fragmentation offset field of 1 except for that last packet which contains 0 to identify last packet



- TTL → decremented by one hop at each router → checksum needs to be recomputed after

Fragment	Bytes	ID	Offset	Flag
1st fragment	1,480 bytes in the data field of the IP datagram	identification = 777	offset = 0 (meaning the data should be inserted beginning at byte 0)	flag = 1 (meaning there is more)
2nd fragment	1,480 bytes of data	identification = 777	offset = 185 (meaning the data should be inserted beginning at byte 1,480. Note that 185 · 8 = 1,480)	flag = 1 (meaning there is more)
3rd fragment	1,020 bytes (= 3,980–1,480–1,480) of data	identification = 777	offset = 370 (meaning the data should be inserted beginning at byte 2,960. Note that 370 · 8 = 2,960)	flag = 0 (meaning this is the last fragment)

IPv4 Addressing

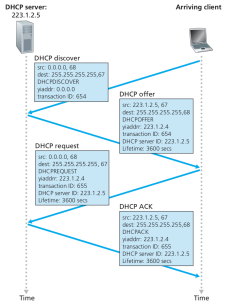
IP address divided into network address and host address.

Full IP address ->193.32.216.9/X<- Network Subnet Mask

- *Network/Subnet mask is a string of X 1s followed by 32 - X 0s. It specifies the first X bits as the network address of the IP address. Hosts with the same network address are in the same subnet/ network*

Longest Matching Prefix - At a router, if network address of a packet matches entries of multiple mappings, the one with the longest matching number of bits is selected

DHCP

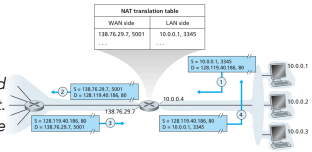


1. *New host first sends a DHCP Discover message using the source IP 0.0.0.0. This IP is valid only for the DHCP message. The broadcast destination address of 255.255.255.255 is used.*
2. *DHCP server sends a DHCP Offer message containing an IP address to be leased with broadcast address 255.255.255.255 as destination (DHCP still does not know who is the requestee)*
3. *Host then responds with DHCP Request acknowledging the leased address by broadcasting.*

Broadcasting allows the DHCP Request message to be sent to all DHCP servers that offers, and rejects the DHCP offer of DHCP servers whose DHCP Server ID doesn't match the one in the broadcasted DHCP Request message.

4. *DHCP server whose request is accepted sends a DHCP ACK.*

Network Address Translation(NAT)



- *Packets sent by hosts within a network have their source IP and Port in the data segment 'translated' into a global IP and Port at the NAT router.*
- *NAT router stores this mapping in the NAT table and uses the global IP and Port as the source IP and Port. Upon receiving a reply to this source IP and Port, the mapping for the reverse IP is used.*

ICMP(Traceroute and Ping)

- *Host first sends a ICMP message with an unlikely port number and TTL = 1 to the next hop router with the intended destination IP.*
- *As the TTL decreases by 1 each hop, when the TTL reaches zero, router discards the ICMP message and sends an ICMP error message(Type 11 code 0) containing the router's IP and name.*
- *Host logs the router IP and name in the error message, increases the TTL by 1 and repeats until the packet reaches the intended recipient. Since the port is unlikely, the recipient discards it and sends an error message(Type 3 code 3) back to the host, signalling the host to terminate.*

Routing Algorithms

Distance Vector Algorithm

- *Each node in the network shares it's distance estimates with each of it's neighbours, which then update their tables and propagate the changes*
- *Slow convergence due to propagation, and may suffer from routing loops*

Link State Algorithm

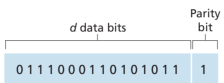
- *All link costs in a network is known to all nodes. This is achieved through each node broadcasting it's link cost to all other nodes.*
- *Faster Convergence than Distance Vector algorithm, more expensive on network*

Poisoned Reverse

- *Solves the problem of counting to infinity if a change in distance vector occurs.*
- *If a node u takes the path to node v through an intermediate node w, it advertises it's cost to v to node w as ∞ - prevents w from taking a path to v through u as it is an intermediate node*
  - *Poisoned reverse only works for nodes directly adjacent to a node and not other intermediate nodes on the path(i.e > 3 nodes in path, does not work)*

Link Layer

Error Detection

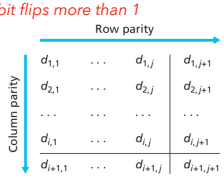


1-D Parity Bit - checks against the number of bits in the data word

Can be even parity (Parity bit + number of 1 bits is even) or odd parity(Parity bit + number of 1s is odd)

Able to detect, but not correct a single bit flip. Unable to detect bit flips more than 1

2-D Parity Bit - Similar to 1-D parity check, but uses a 2-D matrix formed by splitting the data word into rows



Bit flip detected in the row and column - Able to detect and correct any single bit flip, and detect any 2 bit flips but not correct them

Bottom right bit used for detection of bit errors in checksum bit

Cyclic Redundancy Check(CRC)

Computes a checksum of the data using the formula: $R = remainder(\frac{D \cdot 2^r}{G})$

Where D is the data bits, R is the number bits of the generator code and the G is the generator code. The division operation uses XOR (no carry)

Link layer ED handles corruption, but not reordering nor loss of data

Multiple Access Protocols

Channel Partitioning

Frequency Division (FDMA) - channels divided into frequency bands, each sender allocated a specific band that he can send in - prone to frequency jamming

Time Division (TDMA) - Senders each allocated specific time intervals in which they can send their data.

TDMA and FDMA suffers from lack of utilization if there are small number of senders as senders still limited to their own time division/frequency band.

Code Division (CDMA) - Each sender sends data using different codes, and can utilise all time intervals and frequency bands for that code.

Random Access

ALOHA/Slotted ALOHA

- Each sender sends data at full channel capacity independently of other senders.
- If a collision occurs, sender retransmits the frame at each of the next time frames with a random probability *p* until the frame is transmitted successfully
- For slotted ALOHA, time is divided into slots of $\frac{L}{R}$ where L is the frame size and R is the transmission rate of the link - Each slot = the time taken to send a frame.

Carrier Sensing/Collision Detection (CSMA/CD)

Sender listens on the channel before it determines the channel idle to send it's data

If channel not idle, waits for a random period of time before trying again

CD introduces the ability to jam the sending halfway it the sender detects collision during sending process

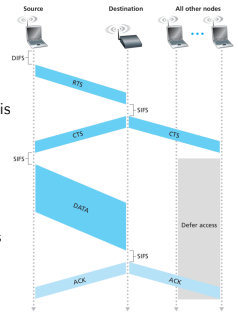
Both CSMA and CD uses Exponential Binary Backoff: a random integer T is picked from the range $[1, 2^n - 1]$ *where n is the number of times sender backs off at a max of up to 10, and* $T \times 512$ *bit time is taken as the wait time*

CSMA/CD suffers from high collisions and slow send rates if there are many senders

Propagation delay along with small packet size may cause data to be sneaked into the channel - need to ensure minimum datagram size

CSMA/CA - Collision avoidance is used rather than collision detection. This is used when the medium is unable to send a jam signal to the send process(i.e wireless networks)

- When sender senses channel idle, it waits for a short period of time known as distributed inter-frame spacing (DIFS)
  - DIFS provides the sender time to ensure the channel is idle
- If channel is busy, sender enters exponential binary backoff, counting down the value only when the channel is idle
- When the counter reaches zero(only happens when channel is idle), the sender sends the frame and waits for an ACK
- If ACK is received, then sender enters exponential binary backoff if it has more frames to send. If it does not receive an ACK, it re-enters exponential binary backoff before attempting to resend the frame
- Short Inter-Frame Spacing(SIFS) is used separate data frames and ACK frames, and also for Clear-To-Send (CTS) and Request-To-Send (RTS) messages to coordinate sending between peers
  - CTS broadcasts tell a specific sender to start sending and all other nodes to cease sending during that period of time, while RTS from the sender informs the AP of the time required for data transmission and ACK.



Taking Turns

Master Node - A master node controls the sending of data by asking senders to send their data if any in a cyclic fashion

Token Ring - Similar to master node, a token object is passed around the senders and only senders with the tokens can send their data.

Single point of failure(token/master node), and master node incurs extra overhead

Security

Message Integrity

Public Key Cryptography - Sender encrypts the data using own private key K_S-, and the receiver decrypts the data using sender's public key K_S+

Symmetric Key Cryptography - Sender and Receiver share the same key which is used to encrypt and decrypt the data.

Faster than Public Key Cryptography, but susceptible to cipher text attack/forging

Common practice to hash symmetric key using public key cryptography and MAC to send symmetric key safely, which will then be used for the session.

Cryptographic Hash - Message is hashed using hash function H and the hash $H(m)$ is sent along with the original message m'. At the receiver's end, $H(m)$ and $H(m')$ compared.

Attack can easily create a new message n and hash it creating matching hash $H(n')$

Message Authenciation

Message Authentication Code(MAC)

- Shared secret s paired with message m and the hash $H(m + s)$ is the MAC, where s is shared known only by sender and receiver before transfer
- $(m', H(m + s))$ is sent to the receiver, knowing the shared secret computes $H(m' + s')$ and compares it with $H(m + s)$
- Shared secret s only known between the sender and receiver, hence $H(m + s)$ cannot be forged by attacker

Digital Signatures

- Sender first encrypts his data using his private key K_S-, creating $K_S-(m)$.
- Sender will then send $K_S-(m)$ to the receiver, who decrypts it knowing the sender's public key
- Provides non-repudiation and verification of the other party - Sender cannot deny the signature

Physical Layer

MAC address - 6 byte physical address burned into the network card - unique to each device. Each interface in a network card can have an IP and MAC address - Routers can have multiple address for each of it's interface

Address Resolution Protocol

Used to resolve IP addresses to MAC addresses for sending of data within same subnet

- When data frame needs to be sent, the network card first checks the destination IP is in the same subnet as itself using it's subnet mask
  - If the destination IP is in a different subnet, the sender puts into it's frame the destination IP, but the MAC address of it's default gateway.
- If the destination IP is in the same subnet, the sender tries to resolve the MAC address of the receiver by looking up it's ARP table
  - If the ARP table does not contain the mapping for the destination IP, the sender sends a special message known as a ARP query packet containing the broadcast MAC address FF-FF-FF-FF-FF-FF but with the destination IP
  - Only the host in the subnet with the destination IP will respond to this ARP query, the other hosts will drop this packet as it is not intended for them
- Sender than puts the destination IP and MAC address in the frame and forwards it through the link to the switch

ARP table entries has a TTL - devices previously plugged in will have entries with their MAC addresses expire, allowing for new devices to replace them

Routing between subnets

- When the packet arrives at the router, the router checks the MAC address against it's own, and after verifying unpacks the frame to reveal the IP datagram
- The router looks up the destination IP in the datagram, and matches it to an entry in it's routing table, and identifies which interface to forward it to
- Router then resolves the corresponding MAC address of the destination IP
- The router than encapsulates the IP datagram in a frame, using the MAC address of the interface it is forwarding and the source IP, with the destination IP and destination MAC

Network Switches

Switching Table

Contains a MAC address - interface mapping indicating which device with the specified MAC address is mapped to which interface.

- Upon receiving a packet, the switch identifies the destination MAC address, and looks up it's switch table to find the mapping for the specified MAC address to identify which interface to be forwarded out to
- If the mapping is not found, the switch forwards the packet out all interfaces not including the sender's

Switch table is self learning - if switch receives frame from interface X and interface X does not have an entry, switch creates a new entry with the source MAC of the frame and interface X

Bit Rate

Theoretical maximum bit rate (bits per second) in:

Noiseless Channel(Nyquist Formula) - $2B \times \lg(L)$

Noisy Channel(Shannon Capacity) - $B \times \lg(1 + SNR)$

Where L is the levels of encoding(how many bits used in encoding), B is the bandwidth of the channel in Hz, and SNR is the signal-to-noise ratio of the channel

Baud Rate - Represents the signal per second rate. Baud rate = $\frac{Bit\ Rate}{n}$ where n is the number of bits per signal used

Analog Transmission Encoding - $A \sin(2\pi f t + \phi)$

A - Amplitude of the signal and height of wave, modified by Amplitude Shift Keying(ASK)

f - Frequency of the signal and number of rounds of wave, modified by Frequency Shift Key(FSK)

- ASK & PSK → single bit signal, QPSK → two bit signal

ϕ - Phase of the signal and shifts the wave, modified by Phase Shift Key (PSK)

2^n-QAM(Quadratre Amplitude modulation) - Combines FSK and ASK to derive *n* different combinations of bits.