CSC 138 Wang Study Sheet

Please feel free to make additions / corrections as needed. Using another color helps differentiate the answers from the given information. If you are not sure about an answer, leave a comment on it or add a note next to it.

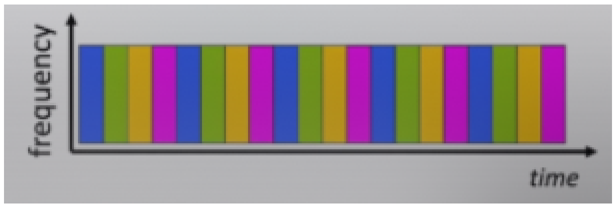
Chapter 1

- What is internet protocol?
    - Protocols control the sending and receiving of messages between network entities. Protocols define how the messages are sent, including the format, order, and actions taken upon transmission or receipt.

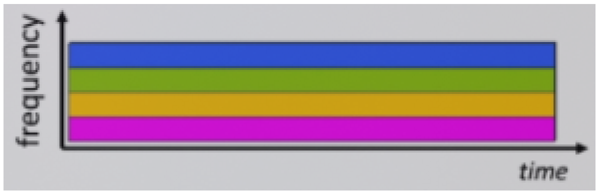
- Network Edge
    - The network edge consists of hosts; clients and servers that receive and store the information.

    - Access networks, physical media
        - Wireless local networks ~100ft
        - 802.1 (wifi): 11, 54, 450 transmission rate
        - Wide area (cellular networks) $G
        - Enterprise networks & Data Centers
    - Host sending function
        - This takes the application message and breaks it into smaller packets of L bits.
        - Transmission Delay = L/R (transmission rate)
- Network Core
    - Packet-switching: Store and forward; end-end delay; queuing and loss;
        - Store and forward: The entire packet must arrive at the router before it can be transmitted on to the next link.
        - end-end delay: It takes L/R seconds to transmit (push out) L-bit packet into link at R bps
        - Queuing and loss: If the arrival rate (in bps) to the link exceeds transmission rate (bps) of the link for some period of time:
            - Packets will queue, waiting to be transmitted on the output link.
            - Packets can be dropped (lost) if memory (buffer) in the router fills up.
    - Circuit switching: TDM vs FDM; Example.
        - Circuit switching: End-to-end resources allocated, or reserved, for the "call" between source and destination. It works like a telephone switchboard.

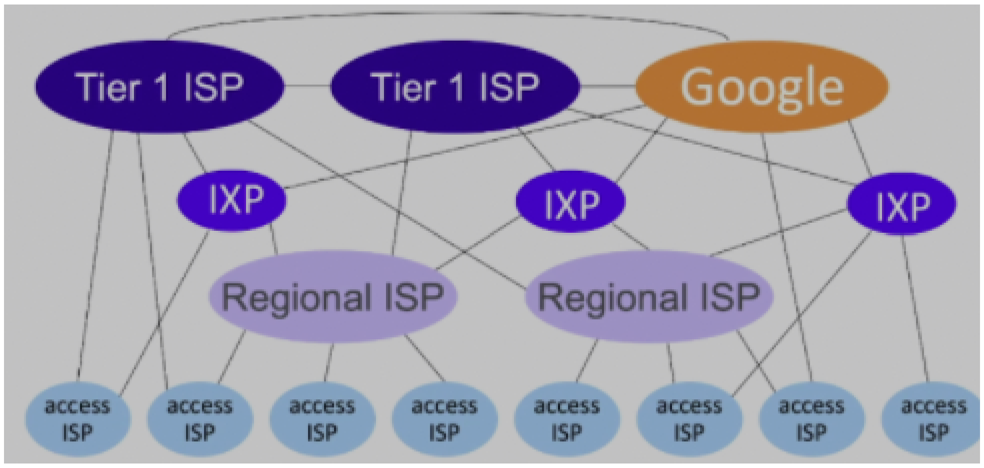
                - TDM: Time Division Multiplexing
                    - Each call is given the maximum rate of the frequency band, but only during the calls timeslot.



                - FDM: Frequency Division Multiplexing
                    - The optical electromagnetic frequencies are divided (narrowed) into frequency bands.
                    - Each call gets its own band, and can only transmit at the max rate of the narrow band.



- Internet structure
    - Also known as a network of networks
    - If we just had every router connected to each other we would need O(N^2) connections.
    - One global ISP could connect customers, but this is a valuable business, so there will be more competitors. The competitor ISPs would also be connected at *Internet Exchange Points* or IXPs.
    - Regional networks also arise to connect access networks to the ISPs.
    - And content providers like Google, Microsoft and Akamai may run their own network, to bring services, content close to the end users.
    - At “Center”: small number id well-connected large networks.
        - Tier-1 commercial ISPs (Level 2, Sprint, AT&T, NTTT) provide national and international coverage.
        - Content Provider networks (Google, Facebook) provide a private network that connects its data centers to the internet, often bypassing tier-1 and regional ISPs.



- Packet delay
    - Four sources of packet delay
        - Nodal processing
        - Queueing delay
        - Transmission delay
        - Propagation delay
- Packet loss
    - If a router is queuing packets because of high congestion, and the queue is full, the packets will be dropped.
- Throughput
    - The rate (bps/time unit) at which bits are being sent from sender to receiver.
        - *Instantaneous:* rate at given point in time
        - *Average:* rate over longer period of time
- Protocol layer
    - Internet protocol stack
        - Application (M): Applications implement services of the transport layer
        - Presentation (ISO/OSI only): Allows applications to interpret the meaning of data (encryption, compression, conventions).
            - Session (ISO/OSI only): Synchronization, checkpointing, recovery of data exchange.
        - Transport (Ht M): Encapsulates the application layer message and integrates the protocol type (UDP, TCP, etc…).
        - Network (Hn HT M): Transfers messages from one host to another using link layer.
        - Link (hl Hn Ht M): Transfers datagram from host to neighboring host using network layer.
        - Physical: Physical layer includes the physical medium of transfer and protocols needed to handle that transfer.

- Encapsulation
    - The process of taking a data unit from a higher layer and then adding information to create a new data unit, this new protocol data unit at another layer is called encapsulation.
- Network security
    - DoS
      Purposeful overloading of a server with requests (usually through multiple machines controlled by one attacker) to prevent legitimate access.

    - Packet sniffing
      Intercept and copy packets via an application. Unencrypted info can be

exposed easily.

- IP spoofing
  Creation of packets with a modified source address to hide the identity of the sender or impersonate another sender.

Chapter 2

- Application models
  - Client-server
    Clients do not directly talk to each other; mediated by a server. Managing connection is far simpler than p2p.

  - Peer-to-peer (P2P)
    Direct connections between endpoints (clients). Peers request and fulfill services, and each new peer brings more capacity. Managing connections is far more complex.

- Socket
  Metaphorical doorway on each side of the connection between the application and transport layers.

- Transport protocols
  - TCP
    - Reliable transport
    - Flow control; sender will not overwhelm receiver
    - Congestion control; sender can be throttled
    - No timing, security, or minimum throughput
    - Setup is required to establish connection
  - UDP
    - Unreliable transport
    - No flow control, congestion control, timing, minimum throughput, security, or setup
- Web and HTTP
  - HTTP protocol
    Application layer of the internet. Uses client / server model. Uses TCP. Stores no info about past requests (stateless).

  - Persistent and non-persistent HTTP
    - Non-persistent HTTP sends one object over TCP connection at a time. Downloading multiple objects requires multiple connections
    - Persistent HTTP can send multiple objects over a single TCP connection.

- RTT
  Time for a small packet to travel from client to server and back

- HTTP response time
  1 RTT (TCP connection) + 1 RTT (HTTP request and response) + file transmission time = 2RTT + file transmission time

- HTTP response message (see example)

  - 200 OK = success
  - 301 Moved Permanently = requested object moved and location specified in location field
  - 400 Bad Request = request not understood
  - 404 Not Found = requested resource not found
  - 505 HTTP Version Not Supported

- Cookies
  For keeping track of data / states in stateless situations. Cookies are kept on the user's host and managed by the browser.

- Web caches
  - Satisfy client requests without involving origin server. This uses a proxy server as a local Web cache. You might find these on campus to decrease the access link utilization.
- Conditional GET
  Don't send objects if the browser cache has an up to date version already. If an object is not modified since the last cache, returns 304 Not Modified.

- Electronic mail
  - Three major components
    - User agent: Client that reads, edits messages
    - Mail server: Server that stores messages, sends mail between servers with SMTP
    - SMTP (Simple Mail Transfer Protocol)

  - SMTP
    - Simple Mail Transfer Protocol
    - Used to push outgoing emails.

  - Mail access protocols (IMAP, HTTP)
    - IMAP used to retrieve emails
    - HTTP user interfaces are provided by services such as gmail for ease of use. HTTP is layered on top of IMAP and SMTP protocols.
  - SMTP vs HTTP

            - Both have encapsulated objects, use ASCII commands and responses, and response status codes.
            - HTTP pulls data, does not necessarily use a persistent connection
            - SMTP pushes data, uses persistent connections, message in 7-bit ASCII, and CRLF determines end of message
- DNS
    - Definition: Domain Name Service
    - Services: Convert domain to IP address, address aliasing
    - Different Tiers of servers
        - Root DNS Servers
        - Top Level Domain DNS Servers (.com, .edu, .org, etc…)
        - Authoritative DNS Servers (yahoo, amazon, pbs, etc...)
    - DNS resolution example
        - Client wants www.amazon.com
        - Query root servers to find .com DNS server
        - Query .com DNS server to get amazon.com DNS server
        - Query amazon.com DNS server to get IP address of www.amazon.com
    - DNS records (resource records (RR))
- P2P 🍐2 🍐
    - Example about file distribution using CS vs P2P
    - BitTorrent
- Video Streaming
    - DASH: server and client
        - Dynamic Adaptive Streaming HTTP
        - Server:
            - Divides video file into multiple chunks
            - Each chunk encoded at multiple different rates
            - Different rate encodings stored in different files
            - Files replicated in various CDN nodes
            - **Manifest File:** Provides URLs for different chunks
        - Client:
            - Periodically estimates server-to-client bandwidth
            - Consults manifest, requests one chunk at a time
                - Chooses maximum coding rate sustainable given current bandwidth
                - Can choose different coding rates at different points in time(depending on available bandwidth at time), and form different servers
            - Client gets to determine:
                - **When** to request a chunk (so that buffer starvation, or overflow does not occur)
                - **What encoding rate** to request (higher quality when more bandwidth available

            - **Where** to request chunk (can request from URL server that is “close” to client or has high available bandwidth
    - Content distribution networks
        - Store and serve multiple copies of videos at multiple geographically distributed sites.
            - **Enter Deep:** Push CDN servers deep into many access networks.
                - Close to users
                - Akamai: 240,000 servers deployed in > 120 countries (2015)
            - **Bring Home**: Small number (10’s) of large clusters in POPs near access nets
                - Used by Limelight
- Socket programing
    - UDP and TCP
    - Application example

```
#UDPServer.py
from socket import *

serverPort = 12000
serverSocket = socket(AF_INET, SOCK_DGRAM)
serverSocket.bind(('', serverPort))
print("The server is ready to receive")
while True:
    message, clientAddress = serverSocket.recvfrom(2048)
    modifiedMessege = message.decode().upper()
    serverSocket.sendto(modifiedMessege.encode(), clientAddress)

#UDPClient.py
from socket import *
serverName = 'localhost'
serverPort = 12000
clientSocket = socket(AF_INET, SOCK_DGRAM)
message = raw_input('Input lowercase sentence:')
clientSocket.sendto(message.encode(), serverName, serverPort)

modifiedMessage, serverAddress = clientSocket.recvfrom(2048)
print (modifiedMessage.decode())
clientSocket.close()
```

Chapter 3

- Transport-layer services
    - Allows for direct communication between host applications
    - Implemented in sender and receiver endpoints but not routers
    - Sender breaks messages into segments passed to network layer
    - Receiver reassembles segments into messages and passes to application layer
    - TCP and UDP available to internet applications
- Multiplexing and demultiplexing
    - Multiplexing: take multiple application layer messages and passing them to a single IP
        - Multiple sources condensed into one stream
    - Demultiplexing: take segment from IP and split it and pass messages to application layer (like splitting)
        - Host receives IP datagram
            - Each datagram has source IP address, destination IP addr.
            - Each datagram carries one transport-layer segment
            - Each segment has source, destination port number
            - Host uses IP addrs. and port # to direct segment to socket
- UDP
    - Unreliable unordered transport
    - No flow control, congestion control, timing, minimum throughput, security, or setup
- Internet checksum
    - Used to detect errors (flipped bits)
    - Is a weak protection, errors can still slip by
- Rdt1.0 to 3.0
    - Reliable Data Transfer protocol
    - Implemented in application to add reliability to UDP
    - Not as reliable / functional as a real protocol
        - 1.0
            - Reliable channel with no packet loss or bit errors
            - Separate FSMs for sender and receiver
        - 2.0
            - Channel may flip bits, checksum used to detect this
            - Error recovery involves ACKs and NAKs
                - ACK: Packet received OK
                - NAK: Packet received with errors
                - Retransmit if NAK
            - Stop and wait method: sender sends one packet then waits for response, repeat

            - Flawed, ACKs and NAKs can be corrupted and leave the sender stuck without knowledge of what happened at receiving end
        - 2.1
            - Sequence number added at sender (0,1)
            - Receiver checks if packet is duplicate via expected sequence number
            - Receiver still can’t know if ACK / NAK received okay
        - 2.2
            - NAK-less functionality, last okay ACK is sent instead
            - Duplicate ACK at sender triggers retransmission of current packet
        - 3.0
            - Channel can flip bits and also lose packets
            - Sender uses a timeout to wait a reasonable amount of time for ACK and retransmits if none received
            - Transmission delay = L / R where L is packet size and R is link rate
            - RTT high due to stop and wait, performance sucks
- Pipelined protocols
    - Increase performance by sending multiple packets at a time
    - Requires large range of sequence numbers
    - Go-Back N
        - Sender has an N-sized window of consecutive transmitted unACKed packets
        - When a packet is ACKed, move the window up one
        - Timeout for oldest packet results in re-transmission of all packets in window
        - Receiver always ACKs highest in-order sequence numbered packet that is received without error
        - If an out of order packet is received, re-ACKs the packet with highest in-order sequence number
        - Can be buffered or not
    - Selective repeat
        - All packets within window are individually ACKed by receiver and tracked for timeout by sender
        - Timed out packets are retransmitted individually
        - Received out of order sometimes, but buffered for eventual in-order delivery to application
- TCP overview
    - Reliable transport
    - Flow control; sender will not overwhelm receiver
    - Congestion control; sender can be throttled
    - No timing, security, or minimum throughput
    - Setup is required to establish connection

- Seq and ack
    - Sequence numbers used to determine if packets are received in the "expected" order
    - ACKs used to notify sender that their packets were received correctly, or, if not received, to retransmit depending on protocols
- RTT
    - Used to determine time intervals
- Retransmission
    - When sender receives 3 duplicate ACKs for the same packet, resend the unACKed segment with smallest sequence number
    - It is likely the unACKed segment was lost, so don't wait for timeout
- Flow control
    - In TCP, receiver uses RWND value to guarantee buffer does not overflow
    - Receiver controls the flow of data from sender by advertising free space via RWND field in TCP headers
    - Sender limits amount of unACKed in-flight data to the received RWND value
- 3-way handshake
    - Prevents possible issues with a 2-way handshake (half open connection, accepting duplicate data)
    - Sender finds receiver, receiver responds, sender sends initialization

- TCP congestion control
    - AIMD
        - Additive Increase Multiplicative Decrease
        - Increase send rate by 1 segment size every RTT until loss detected
        - Cut sending rate in half at every loss event
    - RENO and Tahoe
        - RENO: Cut send rate in half on triple duplicate ACK
        - Tahoe: Cut send rate to 1 MSS when loss by timeout
    - ECN
        - Explicit Congestion Notification
        - Congestion indicators are sent to destination
        - Destination flags ACK segment to notify sender of congestion

Chapter 4 and 5

- Forwarding and routing
    - Forwarding: process through a single interchange (from one router's input link to the appropriate output link)
    - Routing: planning of the whole trip from source to destination (utilizing algorithms)
- Data plane and control plane
    - Data plane uses forwarding

        - Control plane uses routing
- What's inside a router
    - Router checks a datagram header to get destination IP
    - Longest prefix matching
        - Uses to forward to correct output interface
        - In order to forward, 3 switching fabrics are used
            1. Memory
            2. Bus
            3. Interconnection network
    - Switching fabrics
        - Transfers packets from input link to appropriate output link
        - Memory, bus, interconnection network
- IP datagram
    - IP addressing
        - 32 bit identifier for each host or router
    - Subnets
        - Isolated networks detached from hosts and routers
    - DHCP
        - Gives the IP address for a device
        - Can also give:
            - Address of first-hop router
            - Name and IP address of DNS server
            - Network mask
    - NAT
        - Network Address Translation
        - All devices in a local network share the same IP address (from the viewpoint of the rest of the internet), but have differing port assignments
    - IPv6 Format
        - 32 bit IPv4 addresses not enough to serve all devices
        - Speeds up processing / forwarding
        - Enables differing network-layer treatment of "flows"
        - No checksum, no fragmentation / reassembly, no options
    - Tunneling
        - Ex. if an IPv6 router sends data to an IPv4 router, tunneling is used to encapsulate the data
- ICMP
    - Internet Control Message Protocol used by hosts and routers to communicate network information including error reporting and echo requests / replies.
    - ICMP messages carried in IP datagrams
    - Messages contain type, code, and first 8 bits of datagram causing error
- Routing protocols
    - Determine good paths through routers from sender to receiver
    - Classification

        - Global vs decentralized:
            - Global: all routers have complete topology
            - Decentralized: routers only initially know link costs to neighbors, but information is exchanged iteratively when needed
        - Static vs dynamic:
            - Static: routes change slowly
            - Dynamic: Routes change quickly and link costs must be updated periodically
    - Link-state routing
        - Centralized Dijkstra's algorithm computes least cost paths from source node to all other nodes
        - $O(n^2)$
    - Distance vector
        - Bellman-Ford equation used to calculate distance vector cost estimates from neighbors
        - Each node periodically sends its own distance vector estimate to neighbors
        - Good news (low costs) travel quickly while bad news (high costs) travel slowly
    - Intra-AS routing (RIP, OSPF, IS-IS, IGRP)
        - Routing within AS network
        - RIP (Routing Information Protocol)
            - DVs exchanged every 30 seconds
            - No longer widely used
        - OSPF (Open Shortest Path First)
            - Link-state routing
            - IS-IS protocol essentially the same
        - IGRP (Interior Gateway Routing Protocol)
            - DV based
            - Formerly CISCO proprietary until 2013
    - Inter-AS routing (BGP)
        - Routing in AS'es
        - BGP (Border Gateway Protocol)
            - Subnets advertise existence and destinations it can reach

Chapter 6

- Link Layer
    - NIC
        - Network Interface Card implements link and physical layer
        - Can be a chip or dedicated hardware
    - Error Detection (CRC)
        - Cyclic Redundancy Check

- Multiple Access protocols
    - TDMA and FDMA
        - Time Division Multiple Access
            - Access to channel divided into rounds
            - Each station gets fixed length slot in each round
            - Unused slots go idle
        - Frequency Division Multiple Access
            - Channel divided into frequency bands
            - Each station assigned fixed frequency
            - Unused channels go idle
    - Slotted ALOHA, Pure ALOHA, CSMA/CD
        - Slotted ALOHA: Synchronized nodes transmit in equally sized frames. If a collision is detected, nodes retransmit with probability p (randomized) until success. 37% efficiency.
        - Pure ALOHA: No slots, no synchronization, collision chance increased, efficiency 18%.
        - CSMA/CD: Carrier Sense Multiple Access with collision detection, defers transmission if channel is busy, collisions easier to detect in wired than wireless.
- MAC addresses and ARP
    - MAC address vs IP address
        - IP address has 32-bits
        - MAC address has 48-bits
        - MAC address is portable, IP address is not
        - MAC address is unique(?), IP address is dynamic
    - To send a frame from one interface another, you need the destination MAC address
        - This can be obtained from an ARP table
- Ethernet (switch)
    - The host is not aware that the switch exists
- Putting all together

Chapter 8

- Network Security
    - Principles of cryptography
        - Public key - Public encryption key is known to all, private decryption key is known only to receiver
        - Symmetric key - Sender and receiver know a shared secret key, but they must agree on this key somehow.
        - How to generate public/private key using RSA:
            - Pick two large prime numbers p and q
            - Compute n = pq
            - Compute z = (p-1)(q-1)

            - Choose e < n such that e has no common factors with z, both e and z are “relatively prime”
            - Choose d such that ed - 1 is exactly divisible by z (IE ed mod z = 1)
            - Public key is (n,e)
            - Private key is (n,d)
    - Authentication, message integrity
        - Message integrity
            - Use hash function for longer messages to compress them (message digest)
            - Use a private key to encrypt the message
    - Securing e-mail
        - 3 keys used to ensure message integrity, authentication, and confidentiality: sender’s private key, receiver’s public key, and new symmetric key

Short Answer Examples

Please give the names of the seven layers of ISO/OSI reference model.
Application, presentation, session, transport, network, link, physical.

What is the main reason that packet loss may occur in a computer network?
The packet queue becomes too long and the router has no memory to hold the incoming packet so it gets lost.(AKA: Buffer Overflow)

smtp

True/False Example

Packet switching is dedicated, while circuit switching is not.
False, circuit switching is dedicated, while packet switching is not.

Multiple Choice Example

Which of the following about SMTP is false?
(a) It enables a mail server to send email messages to another mail server
(b) User agent cannot use it to retrieve email
(c) It typically uses a connectionless protocol such as UDP
(d) Unlike FTP it uses a single port number

SMTP uses TCP as all packets are important and reliable transport is important.

Long Answer Example

Please fill the following blanks in python code for the implementation of the TCP or UDP server/client,

serverSocket = __________________(AF_INET, __________________)

TCP: serverSocket = socket(AF_INET, SOCK_STREAM)
clientSocket = socket(AF_INET, SOCK_STREAM)

UDP: serverSocket = socket(AF_INET, SOCK_DGRAM)
clientSocket = socket(AF_INET, SOCK_DGRAM)

Long Answer Examples

Suppose users share a 3 Mbps link. Also suppose each user requires 150 kbps when transmitting, b. Find the probability that at any given time, all three users are transmitting simultaneouslyut each user transmits only 10 percent of the time. (See the discussion of packet switching versus circuit switching in Section 1.3.)

a. When circuit switching is used, how many users can be supported?
3 Mbps = 3000 kbps. 3000 kbps / 150 kbps = 20 users max.

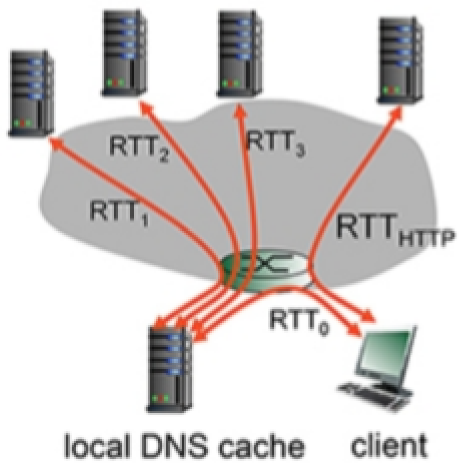
b. For the remainder of this problem, suppose packet switching is used. Find the probability that a given user is transmitting.
The probability that a given user is transmitting is simply 10% or 0.1.

c. Suppose there are 120 users. Find the probability that at any given time exactly *n* users are transmitting simultaneously. (*Hint*: Use the binomial distributio*n.)*
$N\sim Bin(120, 0.1) \Rightarrow P(N = n) = \frac{(120!)}{n!(120-n)} * 0.1^{n}(0.9^{120-n})$

d. Find the probability that there are 21 or more users transmitting simultaneously.
$P(N \geq 21) = 1 - P(n < 21) = 1 - P(n \leq 20) = 1 - \text{binomcdf}(120, 0.1, 20) = 0.00794$

Before doing this question, you might want to review sections 2.2.1 and 2.2.2 on HTTP (in particular the text surrounding Figure 2.7) and the operation of the DNS (in particular the text surrounding Figure 2.19).

Suppose within your web browser you click on a link to obtain a web page. THe IP address for the associated URL is not cached in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that four DNS servers are visited before your host receives the IP address from DNS. The first DNS server visited is the local DNS cache, with an RTT delay of RTT_0 = 4 msecs. The second, third, and fourth DNS servers contacted have RTTs of 2, 31, and 43 msecs respectively. Initially, let's suppose that the web page associated with the link contains exactly one object consisting of a small amount of HTML text. Suppose the RTT between the local host and the web server containing the object is RTT_{HTTP} = 7 msecs.



1. Assuming zero transmission time for the HTML object, how much time elapses from when the client clicks on the link until the client receives the object?
   RTT0+RTT1+RTT2+RTT3+2*RTThttp=4+2+31+43+2*7=94
   One RTThttp for establishing connection and one to send object

2. Now suppose the HTML object references 10 very small objects on the same web server. Neglecting transmission times, how much time elapses from when the client clicks on the link until the base object and all 10 additional objects are

received from the web server (assuming non-persistent HTTP and no parallel connections)?
RTT0+RTT1+RTT2+RTT3+2*RTThttp+2*10*RTThttp=94+2*10*7=234

3. Repeat 2 above but assume that the client is configured to support a maximum of 5 parallel TCP connections with non-persistent HTTP.
RTT0+RTT1+RTT2+RTT3+2*RTThttp+2*RTThttp+2*RTThttp=94'+14+14=122

4. Repeat 2 above but assume that the client is configured to support a maximum of 5 parallel TCP connections with persistent HTTP.
RTT0+RTT1+RTT2+RTT3+2*RTThttp+RTThttp+RTThttp=94+7+7=108

5. What do you notice about the overall delays (taking into account both DNS and HTTP delays) that you computed in cases 2, 3, and 4 above?
That persistent parallel connection is faster than non persistent parallel which is faster than non persistent

In the process of developing a reliable data transfer (rdt) protocol, we may face several issues and have several intermediate versions of this protocol to solve them, such as rdt 1.0, rdt 2.0, rdt 2.1, rdt 2.2, rdt 3.0. Although these versions are not in practical use, their methods become very useful for generating the de-facto protocol, i.e. TCP. In the following questions, please help summarize what methods we can borrow from these early versions.

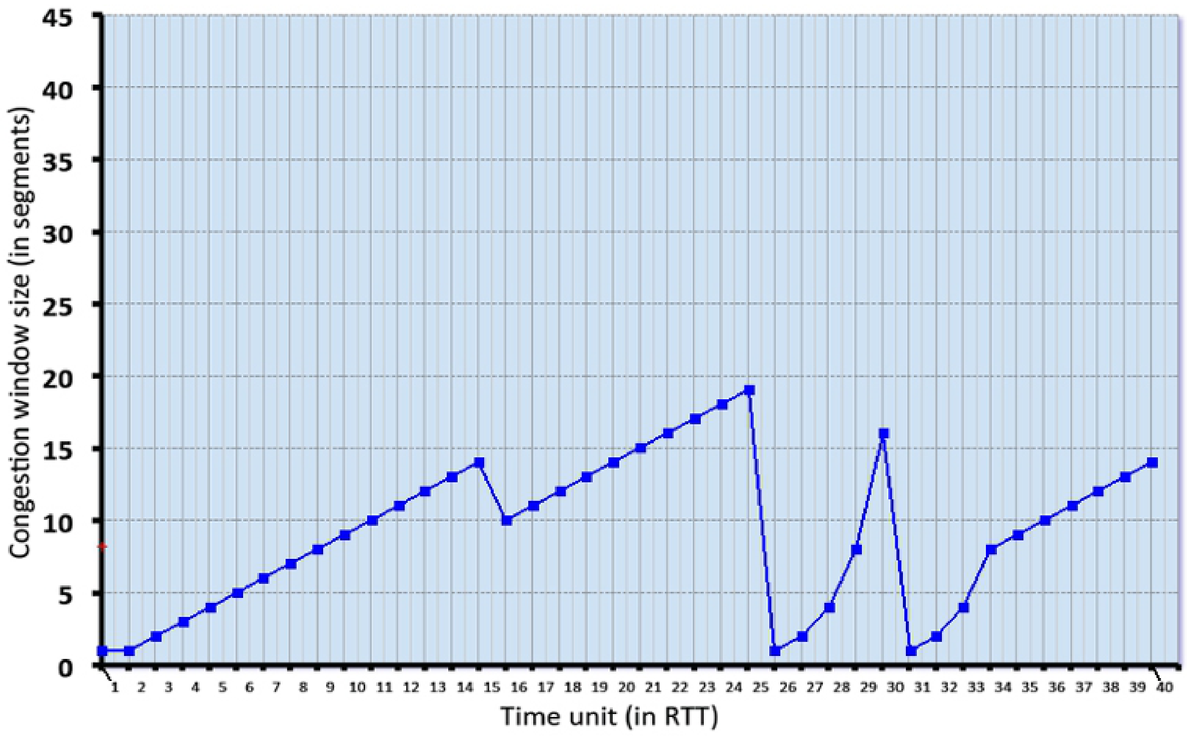
1.0: An underlying channel to send data to and read data from

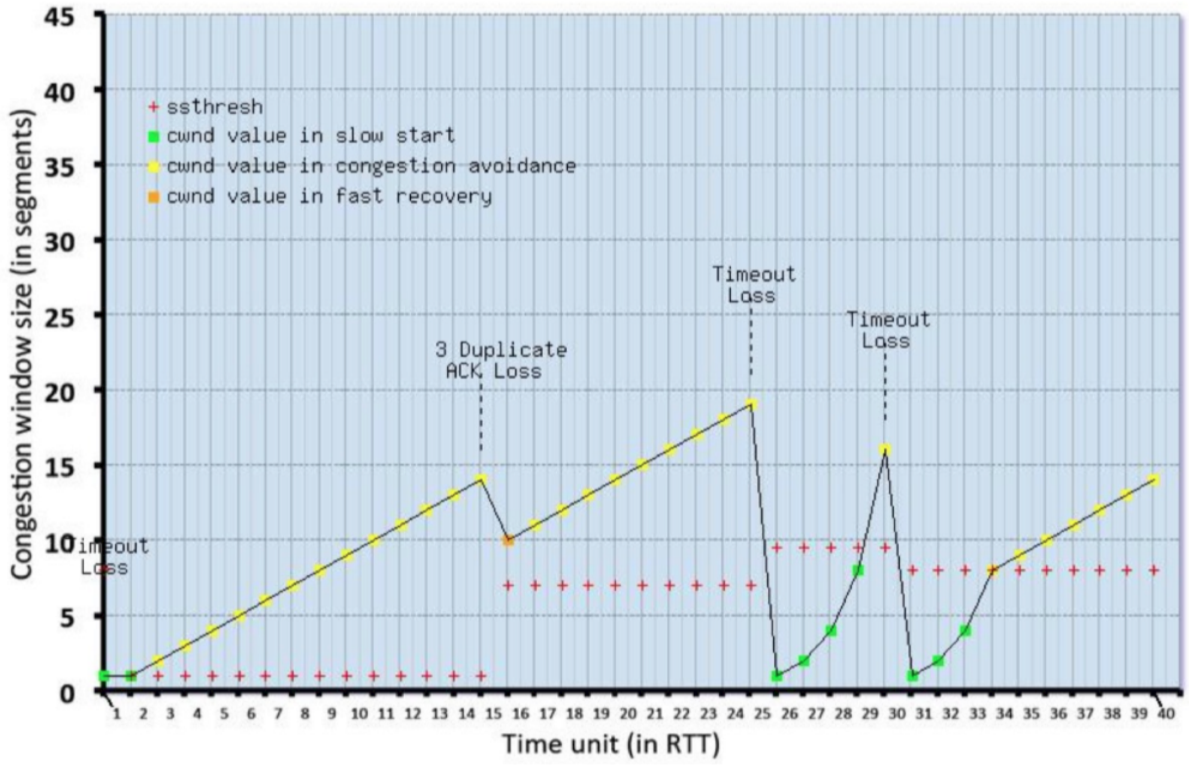
2.0: First use of ACKs to help verify receipt of data

2.1: First use of sequence numbers to check if packet is the one “expected”

2.2: Duplicate ACKs cause retransmission of packets (assumed lost)

3.0: Timeout for waiting for ACKs cause retransmission of packets (assumed lost)





Rules for identifying these events visually:
Timeout loss occurs in the time unit before you see the graph dip to 1 on the Y axis. Even if it is several RTTs worth of sequential 1s. These are all timeouts (as seen at RTT 1 where it is followed by an immediate 1 in RTT 2).

Triple duplicate ACK loss occurs in the time unit before you see the graph dip down anywhere above 1.

Congestion avoidance is in effect anywhere the graph has a constant 1 slope.

Slow start is in effect anywhere the graph has a non-1 slope. These always follow timeout losses.

Ssthresh changes after each timeout event regardless of cause.

Note that technically timeout events occur between the two points where the packets are retransmitted but it is acceptable to just use the first RTT in that pair to distinguish it. For example, there is a timeout between 1 and 2, but we use 1 as the time it occurred in our answer. Just stay consistent.

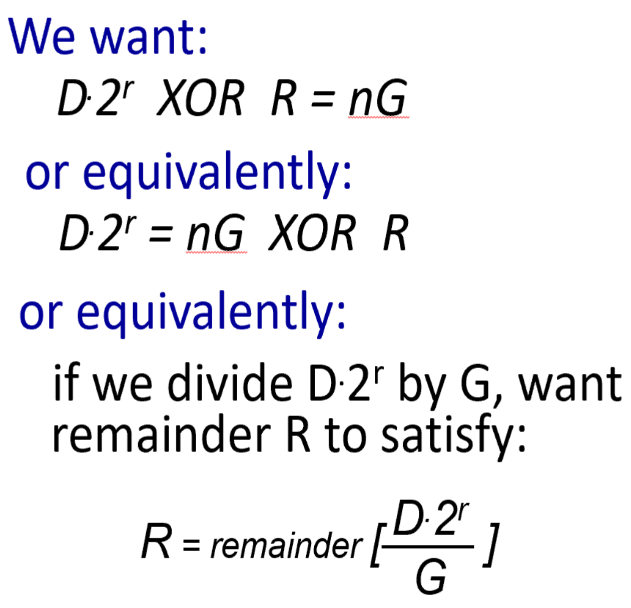
Calculating ssthresh:

- **Initially:** 1 segment size
- **After timeout:** ssthresh = congestion window/2
  newthresh = ss cwnd (at the time of timeout/ACK loss) divided by 2
- **After triple duplicate ACK:** ssthresh = congestion window/2 + 3

Consider the Cyclic Redundancy Check (CRC) algorithm discussed in Section 5.2.3 of the text. Suppose that the 4-bit generator (G) is 1001, that the data payload (D) is 10011101 and that r=3. What are the CRC bits R?

A helpful calculator to understand this algorithm:
https://asecuritysite.com/comms/mod_div
Bit value 1 should be your D*2^r value
Bit value 2 should be G

$D*2^r$=10011101000


We want:

$D \cdot 2^r \;\; XOR \;\; R = nG$

or equivalently:

$D \cdot 2^r = nG \;\; XOR \;\; R$

or equivalently:

if we divide $D \cdot 2^r$ by G, want remainder R to satisfy:

$$R = remainder\left[\frac{D \cdot 2^r}{G}\right]$$

```
         10001100
     ------------
1001)10011101000
     1001
     ----
         1101
         1001
         ----
          1000
          1001
          ----
```

```
             100
   R = 100
```

```
Validating R:
```

goal: choose *r* CRC bits, R, such that <D,R> exactly divisible by G (mod 2)

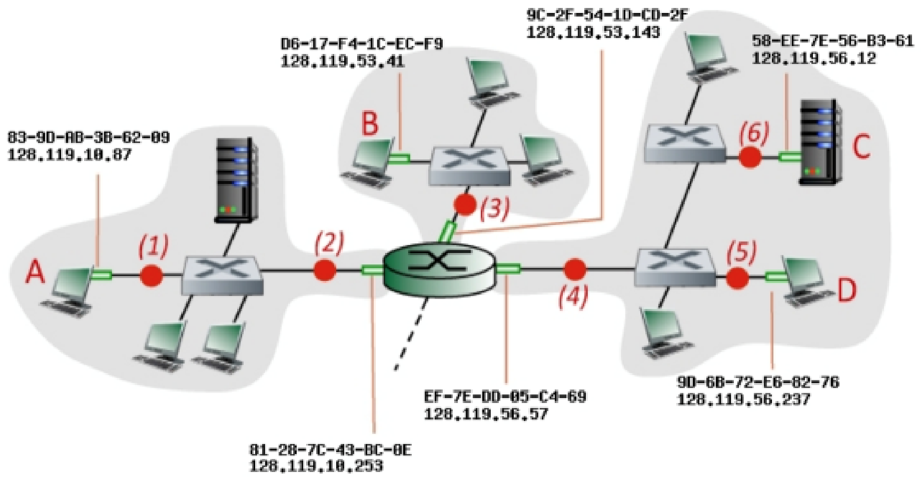
- receiver knows G, divides <D,R> by G. If non-zero remainder: error detected!

```
<D,R> = 10011101100

        10001100
    ------------
1001)10011101100
     1001
     ----
         1101
         1001
         ----
          1001
          1001
          ----
             000
```

Remainder = 0, therefore R is correct

Consider the figure below. The IP and MAC addresses are shown for nodes A, B, C and D, as well as for the router's interfaces.



Consider an IP datagram being sent from node **C** to node **B**. Give the source and destination Ethernet addresses, as well as the source and destination addresses of the IP datagram encapsulated within the Ethernet frame at points *(6)*, *(4)*, and *(3)* in the figure above.

At point 6:

- Source Ethernet Address: 58-EE-7E-56-B3-61
- Destination Ethernet Address: EF-7E-DD-05-C4-69
- Source IP Address: 128.119.56.12
- Destination IP Address: 128.119.53.41

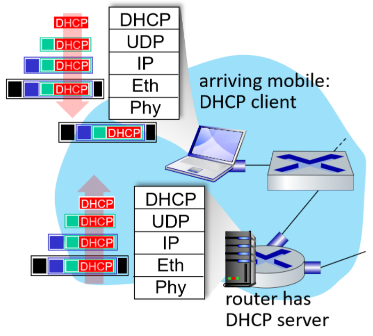
At point 4:

- Source Ethernet Address: 58-EE-7E-56-B3-61
- Destination Ethernet Address: EF-7E-DD-05-C4-69
- Source IP Address: 128.119.56.12
- Destination IP Address: 128.119.53.41

At point 3:

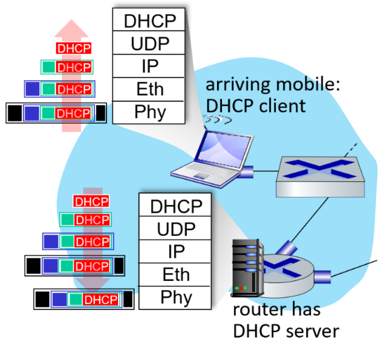
- Source Ethernet Address: 9C-2F-54-1D-CD-2F
- Destination Ethernet Address: D6-17-F4-1C-EC-F9
- Source IP Address: 128.119.56.12
- Destination IP Address: 128.119.53.41

A day in the life: connecting to the Internet



- connecting laptop needs to get its own IP address, addr of first-hop router, addr of DNS server: use DHCP
- DHCP request encapsulated in UDP, encapsulated in IP, encapsulated in 802.3 Ethernet
- Ethernet frame broadcast (dest: FFFFFFFFFFFF) on LAN, received at router running DHCP server
- Ethernet demuxed to IP demuxed, UDP demuxed to DHCP

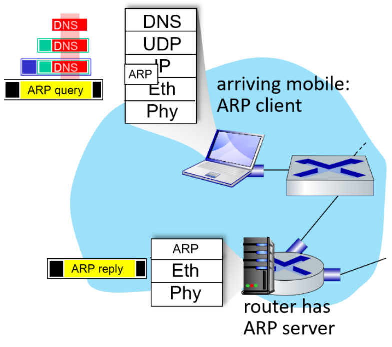
A day in the life: connecting to the Internet



- DHCP server formulates DHCP ACK containing client's IP address, IP address of first-hop router for client, name & IP address of DNS server
- encapsulation at DHCP server, frame forwarded (switch learning) through LAN, demultiplexing at client
- DHCP client receives DHCP ACK reply

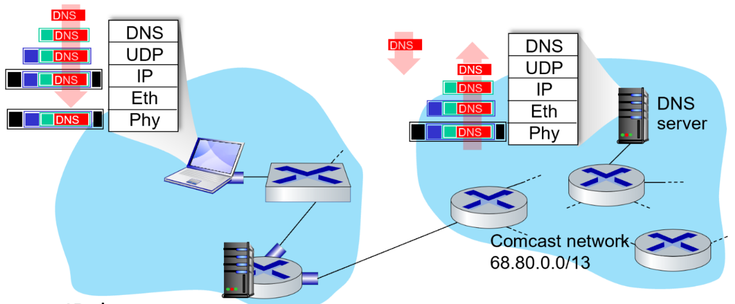
Client now has IP address, knows name & addr of DNS server, IP address of its first-hop router

A day in the life... ARP (before DNS, before HTTP)



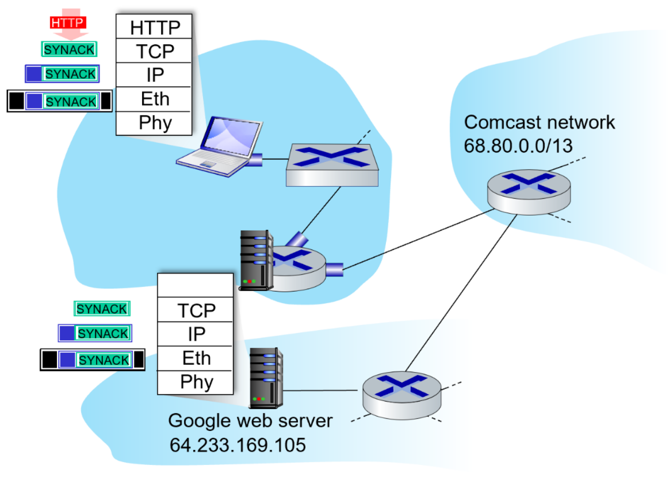
- before sending HTTP request, need IP address of www.google.com: DNS
- DNS query created, encapsulated in UDP, encapsulated in IP, encapsulated in Eth. To send frame to router, need MAC address of router interface: ARP
- ARP query broadcast, received by router, which replies with ARP reply giving MAC address of router interface
- client now knows MAC address of first hop router, so can now send frame containing DNS query

A day in the life… using DNS



- demuxed to DNS
- DNS replies to client with IP address of www.google.com
- IP datagram containing DNS query forwarded via LAN switch from client to 1st hop router
- IP datagram forwarded from campus network into Comcast network, routed (tables created by RIP, OSPF, IS-IS and/or BGP routing protocols) to DNS server

A day in the life…TCP connection carrying HTTP



- to send HTTP request, client first opens TCP socket to web server
- TCP SYN segment (step 1 in TCP 3-way handshake) inter-domain routed to web server
- web server responds with TCP SYNACK (step 2 in TCP 3-way handshake)
- TCP connection established!