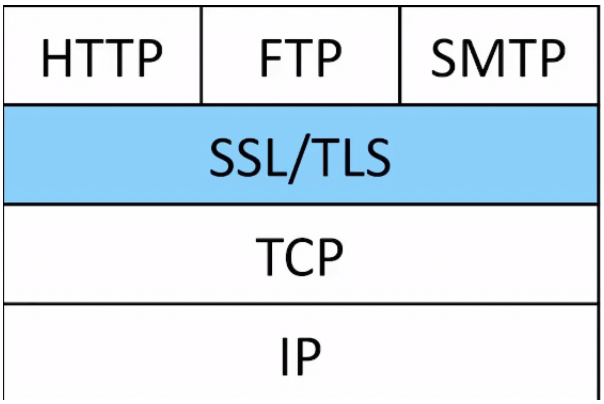
CAS CS 357

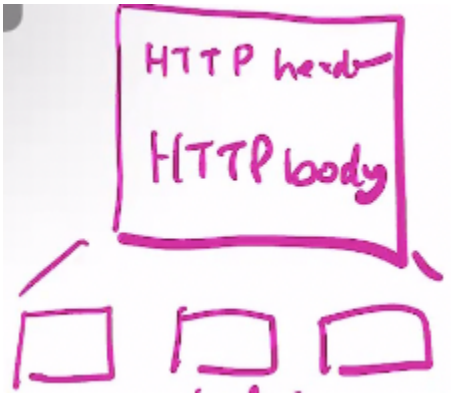
InClass Note 18

1. General Review

a. 

b. TLS → certificates, encryption, authentication, signature

c. TLS sits on top of two protocols: TCP and IP

d. 

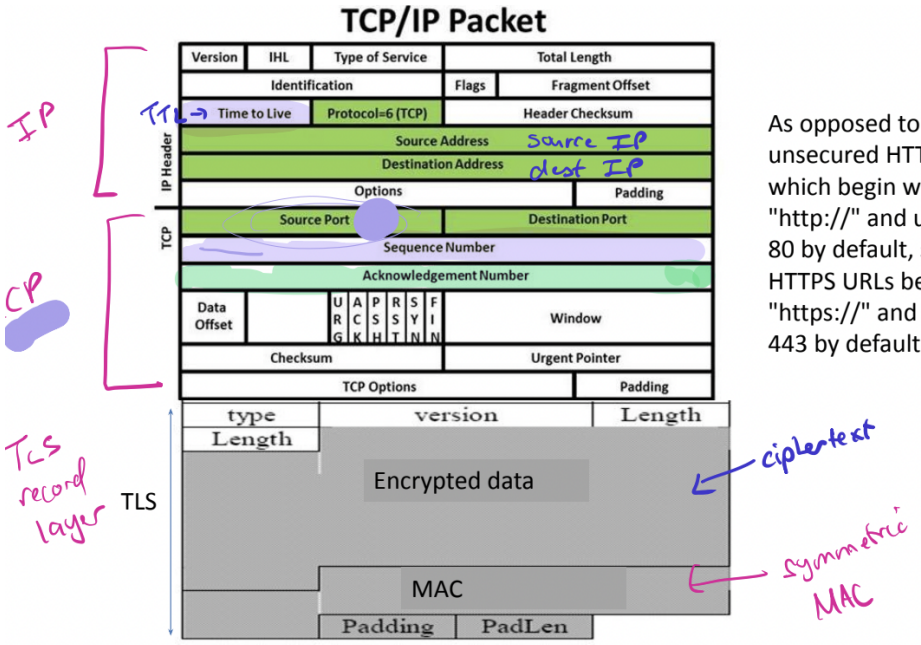
e. 

→ IP packet

f. The HTTP header, body (which includes cookies, hyperlinks, URLs) gets broken into pieces → encrypted and authenticated over TLS → each of the encrypted

pieces of information gets IP header attached, then TCP header, and actual encrypted and authenticated information

2. TLS packet format

a. 

As opposed to unsecured HTTP URLs which begin with "http://" and use port 80 by default, secure HTTPS URLs begin with "https://" and use port 443 by default.

b. TLS record layer includes MAC(symmetric key signature, symmetric MAC, tag on the message)

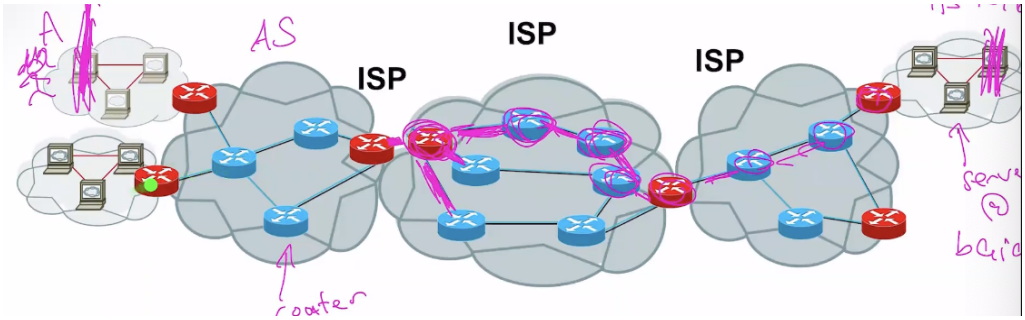
c. MAC provides authenticity and integrity

d. IP includes the source IP, destination IP (ultimate destination)

e. Time to Live field in IP Header → if there exists a loop where the packet goes circle forever, the packet gets dropped after certain number of hops written in the TTL field

f. TCP includes sequence number, acknowledgement number

3. General IP

a. 

b. IP is about guiding the packet from one router to another

c. TCP is about how the client can talk to the server and make sure that all these packets that are traveling over the path are actually arriving in the correct order (not getting dropped and in the right order)

4. IP Protocol Functions (summary)

   a. Routing

      - IP host knows location of router (gateway)
      - IP gateway must know route to other networks

   b. Fragmentation and reassembly

      - If max-packet-size less than the user-data-size (don't use anymore)

   c. Error reporting

      - ICMP packet to source if packet is dropped

   d. TTL field: decremented after every hop

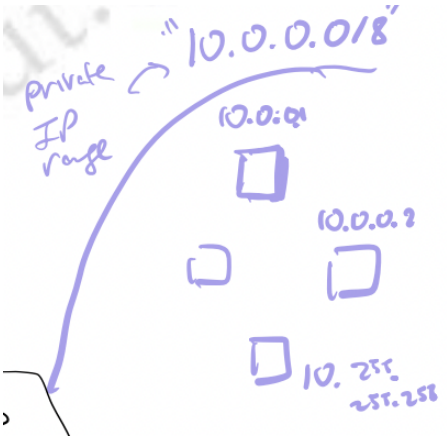
      - Packet dropped f TTL = 0. Prevents infinite loops
      - TTL hop drops by one every time it visits a router (drop packets that circulate forever)

5. Private IPs + NATS (Network Address Translation)

   a. How many IPv4 addresses → 2^32 = 4 billion

b. Ran out of IPs a few years ago (since there are more people than the IP addresses available) → solved by using private IPs

c. Private IPs → IP addresses that are reused (not unique, can be used by lots of networks)

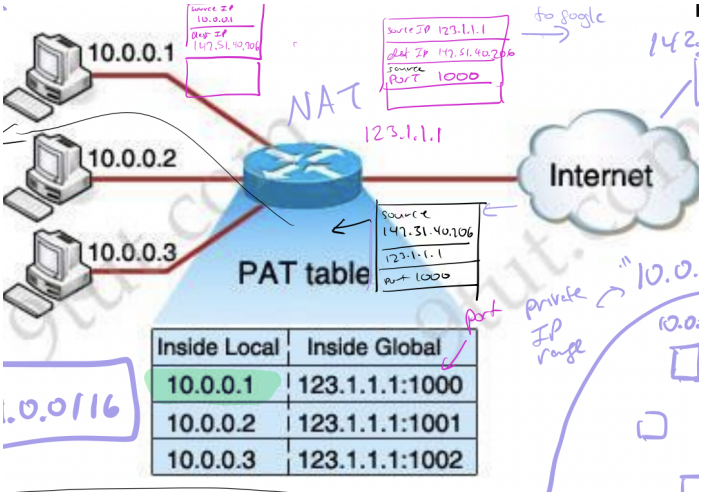
d. 

e. Above figure is a system → the boxes in the system are using IP addresses within the private IP range (10.0.0.0/8 → inside includes 10.0.0.1, 10.255.255.255) → address used internally that can be reused across all networks

f. NAT (Network address translation) → if computers inside a system need to communicate with each other, they will communicate across the 10. IPs

g. If you need to go out to the internet, NAT takes place

h. In other words, the ten dot numbering allows us to number devices inside an internal network so that they can talk to each other. But when they need to talk to the outside world we need to translate them to something that the outside world can understand → NAT takes place

i. 

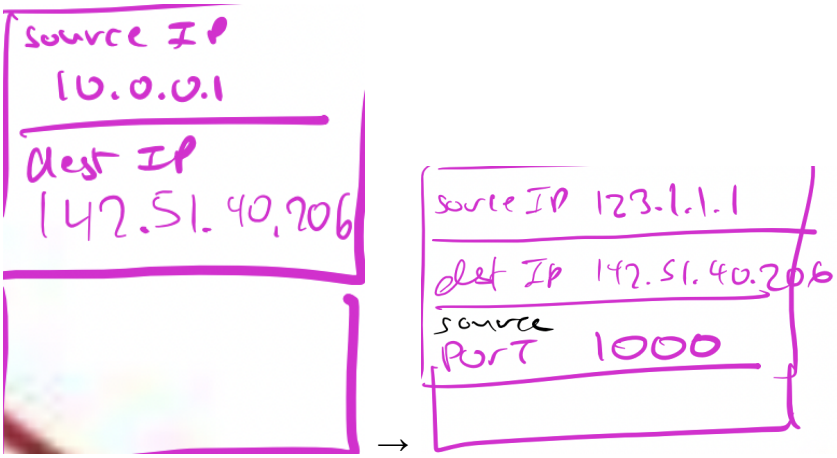
j. IP address of Google: 142.51.40.206 (public), the user is using a private IP

k. How to speak from private IP to public IP:

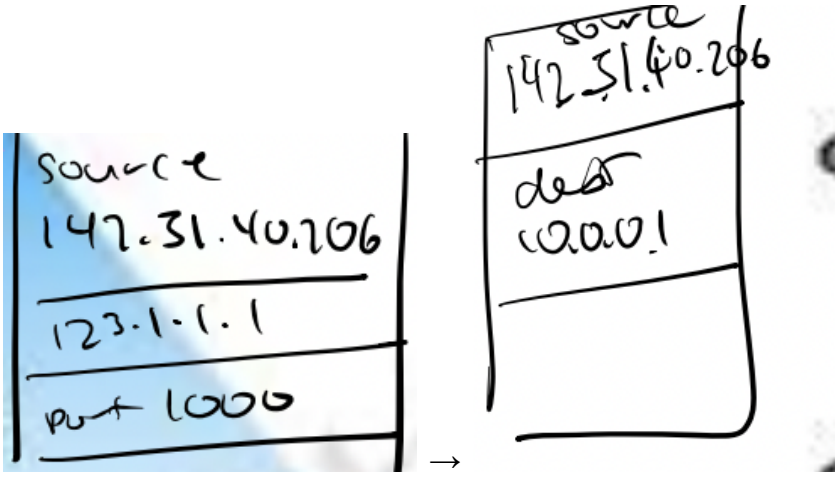
- Initiate the request to Google and get response (form a packet (IP hader, TCP, and TLS))
- Source IP is 10.0.0.1, destination IP is Google's IP
- Response back: NAT includes a table (123.1.1.1 is the one single public IP and the three computers share the same IP address), but each private address includes a port (1000,1001,1002), there exists a port field that saves each individual private IP address
- For example, 10.0.0.1 is translated into the shared IP address and the port 1000

l. Profound thing: your private computer can reach the public IP of Google since it is public but the public cannot connect to your private device since the IP is private (hiding computers from the Internet)

m. 10.0.0.1 sending to NAT to Google (the NAT has IP address of 123.1.1.1)

n. 

o. Google to NAT to 10.0.0.1

p. 

q. Expose a server to the internet means that we put server on public IP

r. So few IPs → make them dynamic → if I connect Google today and tomorrow, I'll probably connect on different IPs → load balancing (different servers exist and if one server is congested while other one is not, I;ll be sent to the server that is not congested) → do not think IPs being bound to like a human being or server, not to a specific computer

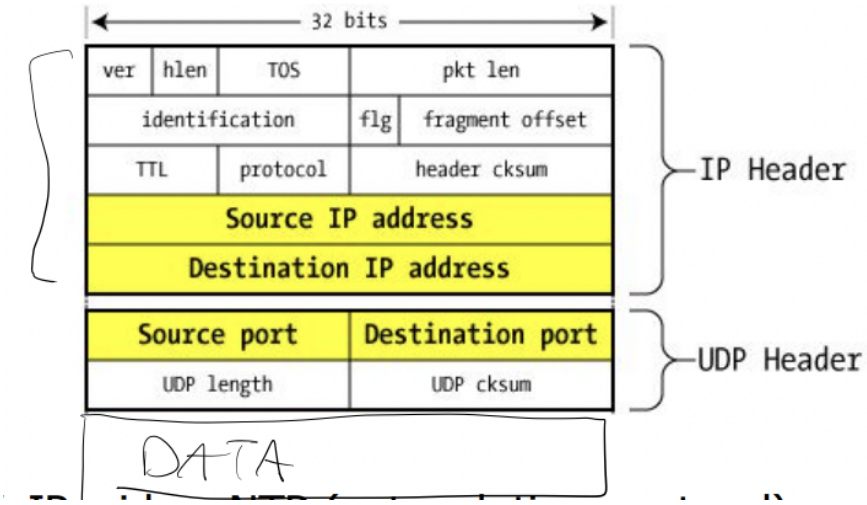
s. Static IP → IP assigned to a specific computer (routers have static IP), things like computers, printers, raptors, devices, host servers do not have static IP

6. Difference in applications

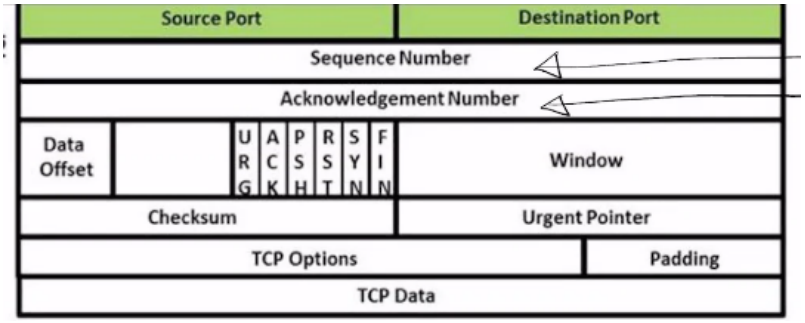
a. Applications that prioritize latency (you can allow few things to drop in the middle but do not allow waiting) → UDP (don't care whether some packets are lost, don't care about the order)
    - DNS
    - Voice Call
    - Streaming video/audio
    - Zoom video calls
    - Network time protocol(NTP)

b. Applications that prioritize no packet loss (you can wait few minutes for it to completely be loaded but you cannot allow things to get lost) → TCP (guarantee that you don't lose any packets, everytime you send a packet to somebody, somebody has to acknowledge it → if don't acknowledgement, resend packets every time)
    - HTTP (viewing a webpage over HTTP)
    - Podcast download

7. User datagram protocol (UDP)

a. 

b. Used for VoIP, video, NTP (Network time protocol) and anything else where latency matters more than reliability
c. UDP length: how long is the packet
d. UDP cksum: MAC (message authentication code), but there is no key and is not secured against adversaries (not cryptographic) → only reason is in case there is some kind of corruption in the path (check that there is no corruption in data)
e. Zoom: when it leaves UDP, it gets sent to zoom and zoom processes it and does all zoom algorithms to make sure that we get a good quality call
f. UDP: protocols themselves handle all reconstruction and don't expect the network to do it for them

8. Transmission Control Protocol (TCP)
a. 

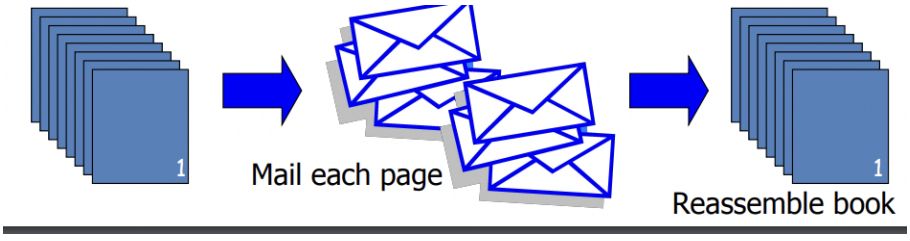
b. Includes acknowledgement number and sequence number
c. Sequence number: ordering of the packets so that they arrive in order
d. Acknowledgement number: acknowledgement number indicates that the receiver received every packet (there exists an acknowledgement number and sequence number in every packet) → for example, there exists packet 1,2,3 (sequence number). If the receiver receives packet 1,2, the receiver sends acknowledgement number for those packets but if the receiver does not receive packet 3, the receiver

does not send acknowledgement number for packet 3 → the sender resends packet 3 until the receiver receives the packet and sends acknowledgement number

e. How HTTP page is loaded:
   - Use TCP to make sure that all the packets arrive in order
   - Assemble packets back together
   - Get sent up to the browser as an HTTP page
   - Browser processes the page
   - Comes as one nice http package altogether in order (fully formed, perfectly constructed web page that we just sent to the browser)

f. TCP: expect the network to reconstruct things for us

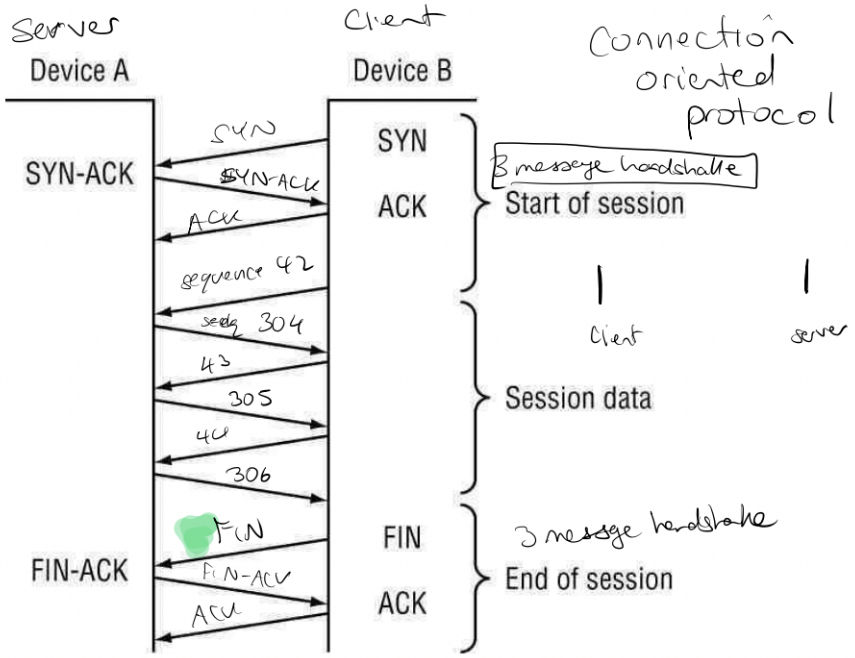
g. Connection-oriented, preserves order
   - Sender
     - ➔ Break data into packets
     - ➔ Attach packet numbers
   - Receiver
     - ➔ Acknowledge receipt; lost packers are resent
     - ➔ Reassemble packets in correct order

h. 

9. General Security Aspect in TCP
   a. Sequence number: do not start with 1 (start counting at random number)

b. If you start the counter from one, bad things can happen → reset attacks, sequence prediction attacks, etc.

10. TCP/IP packet (TCP is connection oriented protocol - client and server identify to each other and then talking just like TLS (get certificate from server and set up keys and then start talking))

a. 

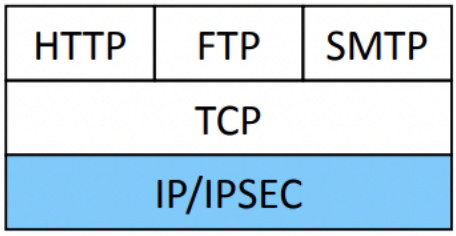
b. Client initiates connection always → because most of the time the client is on a private IP so it cannot be reached anyway

c. Client sends a SYN message → server send a SYN-ACK → client responds with a ACK (3 message handshake → then they communicate)

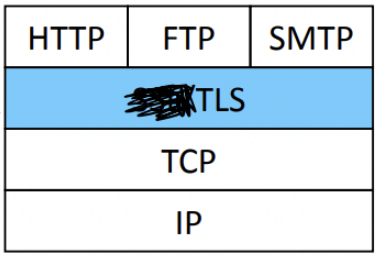
d. When we are finished → close the connection → client sends FIN → server responds with FIN-ACK → client sends ACK (3 message handshake)

e. Anything in between the 3 message handshakes at the bottom and top → sent with sequence numbers (have ordered (or reordered by TCP), reliable (keep sending until the packet arrives))

11. Review



a. At the Network Level



b. At the Transport Level

c. HTTP/SMTP runs over TLS

d. TLS runs over TCP

e. TCP goes first

f.

g. What happens on the outside goes first → TCP (things run over IP, TCP, and TLS)

h. 
client
SYN
SYN AC
ACK
client
hello
server hello
certs
2022-11