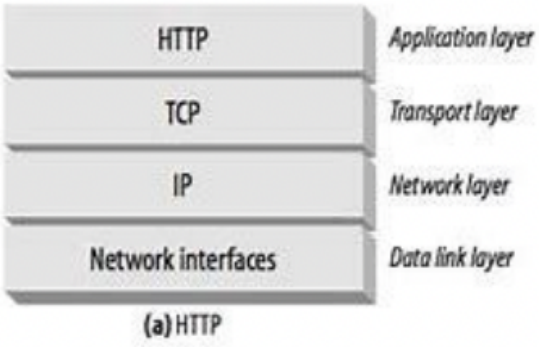
CAS CS 357

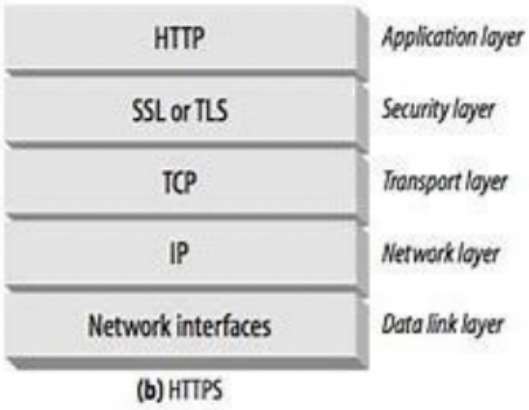
InClass Note 20

1. Layers



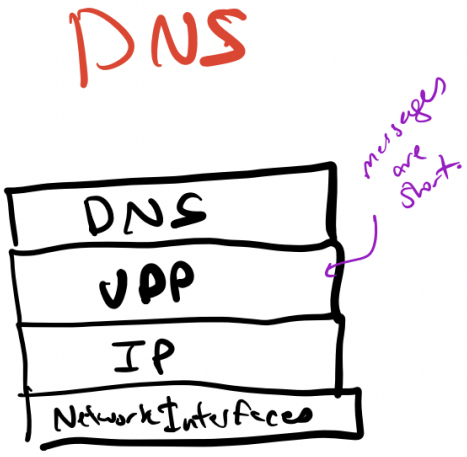
   a. HTTP:

   b. HTTP → what we use to communicate with websites (things such as apis need to arrive correctly in order in one piece without pieces missing) → runs on top of TCP (provides reliable transmission so that we can send messages without parts of them getting lost or dropped)



   c. HTTPS:

   d. HTTPS has an additional layer (TLS) → web page gets encrypted with a symmetric encryption scheme and authenticated with a MAC

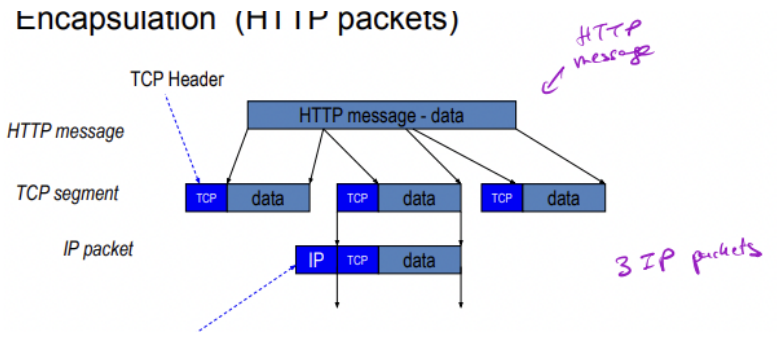


(c) DNS

e. DNS:

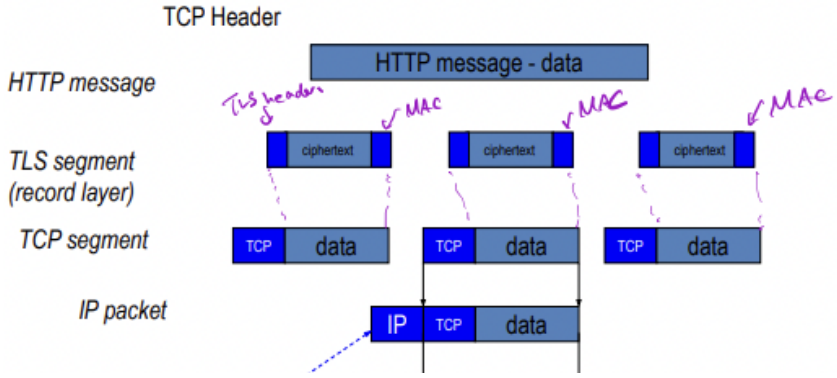
f. DNS (one packet that goes over query): DNS over UDP (runs over UDP because the messages are very short → because no need to set up handshake and do things for just a response (the IP is ???.???.???.???) → it is just one packet, no need to hold authentication and TCP)

2. Encapsulation

a. 

for HTTP

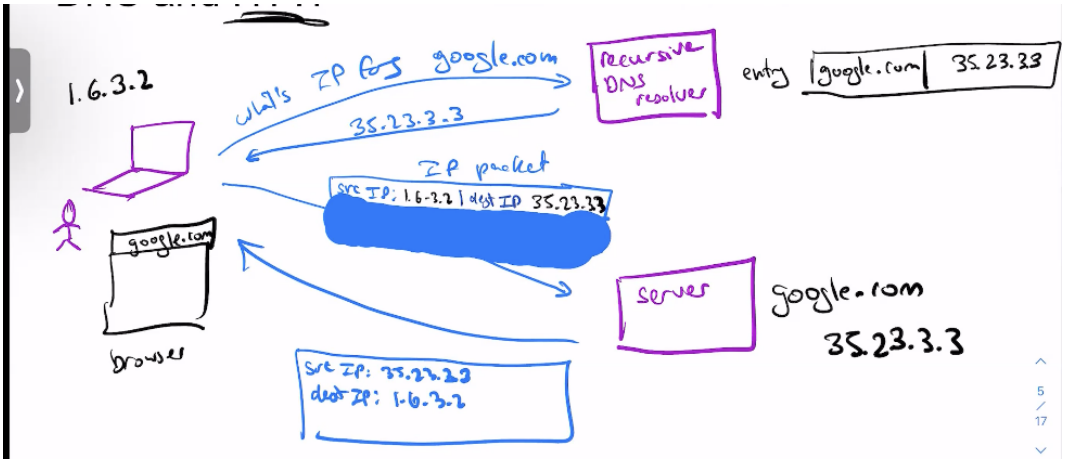
b. HTTP message is converted in to 3 IP packets in the figure above

c. 

for HTTPS

d. For HTTPS, there are TLS headers and MAC with the message that gets put into a TCP packet

3. DNS and HTTP (Assuming that laptop is public IP)

a. 

b. If the user has private IP, there is an additional layer (NAT) that includes the port

c. DNS resolver has an entry that stores the IP address of google → 35.23.3.3 (assuming that google takes HTTP)

d. User (who has IP of 1.6.3.2) types http.google.com in the browser

e. User asks the IP for google.com to the resolver and learns that it is 35.23.3.3

f. User connects with google.com by sending IP packets that includes the following information:

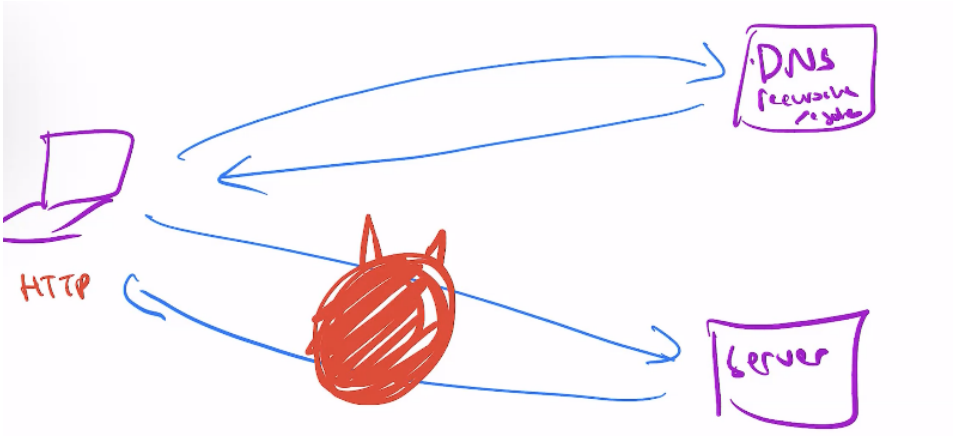
- Source IP: 1.6.3.2

- Destination IP: 35.23.3.3
- Information that we want to send

g. Response comes back to us by google.com sending IP packets including following information:

- Source IP: 35.23.3.3
- Destination IP: 1.6.3.2
- Information that google wants to send

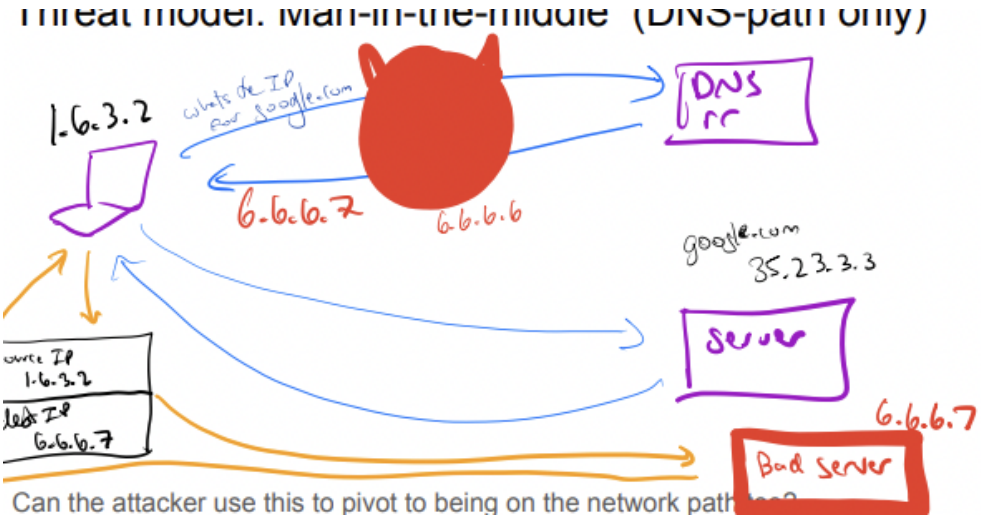
4. DNS and HTTP (Man-in-the-middle)



a.

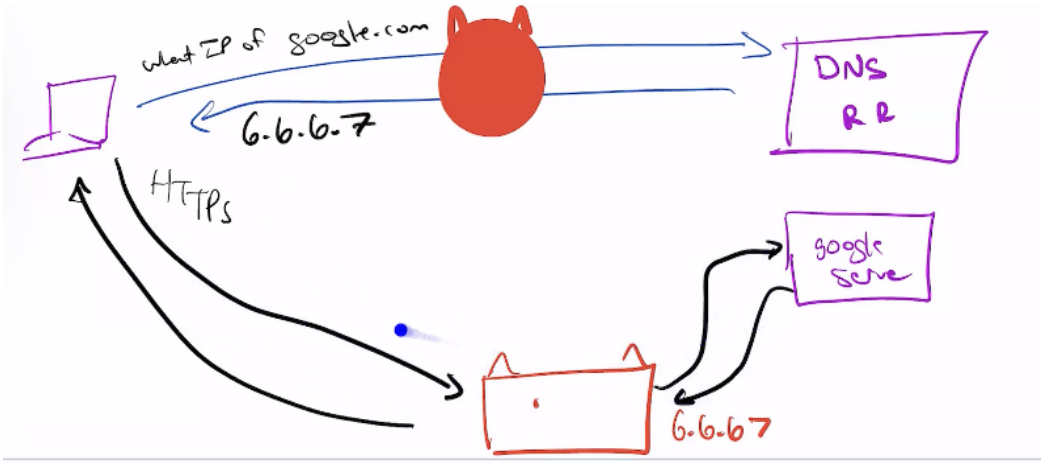
b. If there is man-in-the-middle between the user and the computer, what will the adversary be able to do?

- Completely alter communication → can be bad because the communication can be access to email or bank instructing to send the money to the adversary (already studied)

c. 

d. If there is man-in-the-middle between the user and the DNS resolver, what will the adversary be able to do?

- Assuming the adversary has the IP address of 6.6.6.6
- Assuming that no adversary exists between the user and server
- User asks the IP address for google.com → adversary does not tell the real IP address of google but can send IP addresses of malicious websites such as phishing websites (for example, adversary can provide 6.6.6.7 as the IP address which is a malicious website) → user forms packets that have destination IP as 6.6.6.7 → user goes into the bad server without notifying that the user is at the bad server (the browser tells you that you are in google.com → the URL bar states that the user is in google.com → very dangerous since it says that the user is in google.com but actually is in a malicious website)
- If this was HTTPS, the TLS handshake would prevent the adversary from doing so → see certificate error, site is offline or show some kind of error

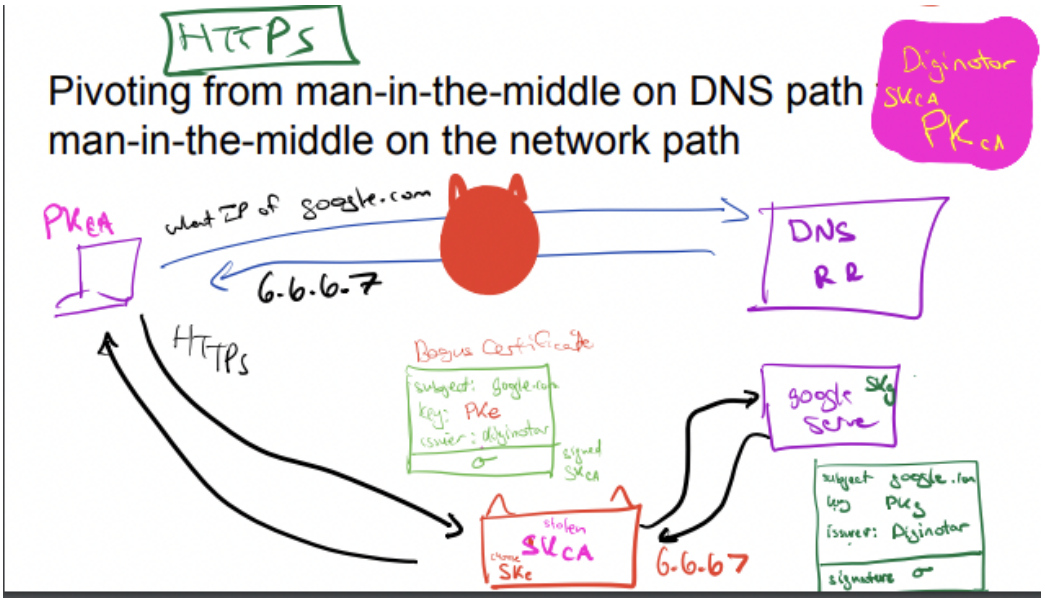
e. 

f. Pivoting from man-in-the-middle on DNS path to man-in-the-middle on the network path

- Assume that there is an adversary that sends the IP address of 6.6.6.7 as a response to the IP address of google.com
- User passes packets to the adversary server and the adversary server can connect with google.com with the information the user sent (no matter whether the user is using HTTP or HTTPS) to connect to google.com
- If we are using HTTPS, the adversary may not be able to read/alter communications but we are still sending traffic through the adversary but the adversary can still know the actions that users are taking (know the servers that the user is visiting)
- DNS attacks are serious because they can cause a rerouting of traffic
- All the traffic is routed to the adversary (even if the routing does not cause TLS to be broken, the adversary can learn the sites you are visiting (has the map of the websites/servers you are visiting)

- Adversary can filter sites → send the information to non-existing IPs and force the packets to be dropped)

5. DNS and HTTPS

a. 

b. If the user connects with the google server using HTTPS (there is certificate from google → subject is google.com, key is PK_s, issuer is Diginotor(in this example), signature is o)

c. The user's browser knows the secret key and the public key of the certificate

d. The google server knows the secret key of the certificate of google.com

e. If for some reason, Diginotor gets compromised → The adversary steals/hacks the certificate authority and therefore learns the secret key

f. The adversary can therefore create a secret key on his/her own (SK_{evil})

g. Adversary creates a certificate that contains subject as google.com, public key of evil, issuer as diginotor, and signed (signed by public key of certificate) → fake certificate (going to validate because it is signed by the certificate's key) and we do not know the difference

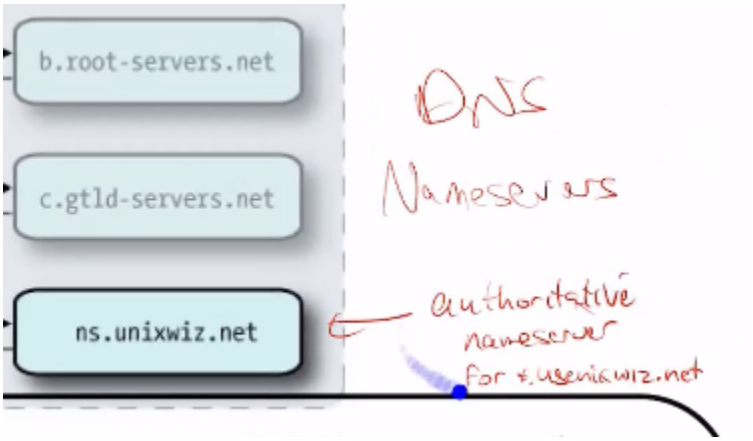
h. Therefore, when users speak TLS to the server thinking that you are speaking to google (domain has been owned and modified to 6.6.6.7(you think you are visiting google) and even the certificate has been owned since the secret key has been leaked and stolen to the adversary)

i. Adversary can read all the TLS traffic due to the setup of a TLS session with the adversary and the user is communicating with the dangerous server as if the user thinks it is google

j. The adversary can also create a TLS to the google server with the information that is sent to to the adversary by the user (can modify/add the information/communication received from user to the server by decrypting the information from the user using the secret key created by the adversary, and alter the information, encrypt it, and send it to the server by the adversary) → EXAM

k. What is scary about this attack → it is not just associated with google but is associated with every website in the Internet

6. DNS basic picture

a. 

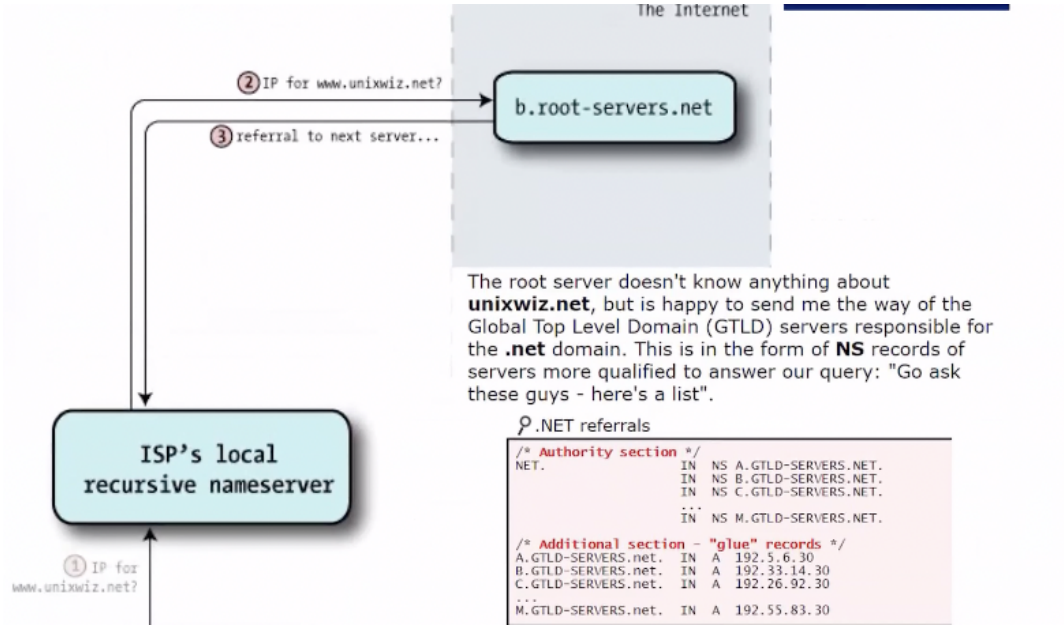
b. Ns.unixwiz.net → authoritative nameserver for the URL

c. The third server is the server that knows the answer to the IP addresses of the websites (authoritative nameserver) → DNS's job is to get the user to the authoritative nameserver so that the user can get the answer

d. How does that work → hierarchical system

e. Recursive resolver goes out to the internet → go to the root and asks who knows the answer to the question and the root (does not know the answer to the question but it knows that if the question goes to .net, maybe the .net knows what to do) tells where to go to answer the question→ go ask the .net server(does not know the answer to the question but knows what server is authoritative for unixwiz.net) and the .net server tells where to go to answer the question→ go ask the authoritative server and learn the answer

7. Root-server

a. 

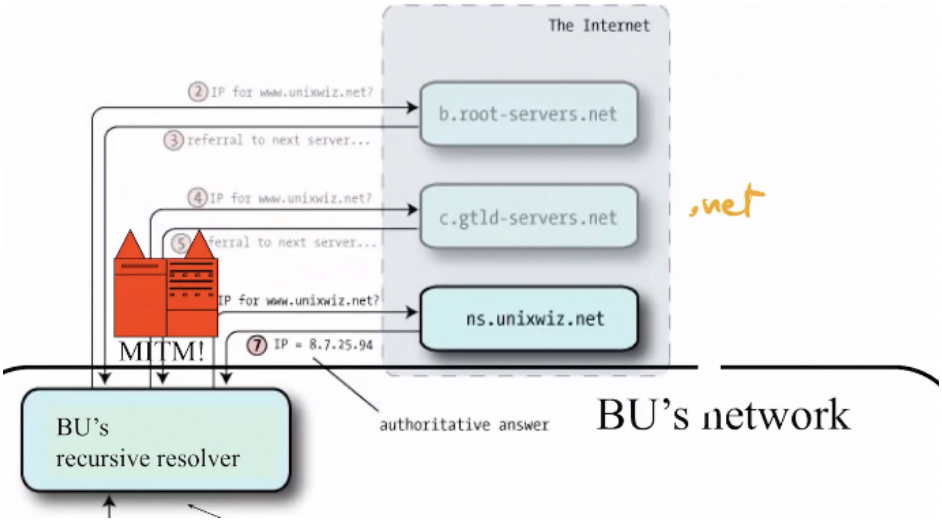
b. The root server doesn't know anything about unixwiz.net but knows that if the user wants to talk to .net, the root server will give the names of the servers that

know the answer → for each one of these servers, the root server tells the user the IP of the server so that the user can form a query → the next thing to do is the user actually sending a packet to that server chosen by the root server by using the IP address → continue this step

c. For example, the root server will return 192.5.6.30 as the IP address that will help answer the question

8. DNS attack

a. DNS → just a UDP packet (no encryption, no authentication)

b. 

c. Man-in-the-middle for .net → can cause lots of damage → can change the referrals in anything in .net → man-in-the-middle can direct you to the wrong referrals within the .net (alter the response)

d. Recursive resolver → cache system(going through all the steps above every time users visit website takes too much time→ reduce the amount of traffic that goes out on the internet) → cache all the referrals from .NET until Time to Live variable expires → DNS cache poisoning attacks can occur (adversary put incorrect information in the cache that is incorrect that can lead to further attacks)