I - WS-Ex-01. II) **Christopher Zúñiga C28730**

a) In the first few packets, the client machine is looking up the common name (cname) of a web site to find its IP address. What is the cname of this web site? Give two IP addresses for this web site.?

The **CNAME (Canonical Name)** of the website in the DNS response (packet 2) is:

- `www.yahoo.akadns.net`

Two IP addresses for this website, also found in the DNS response, are:

- **216.109.117.106**
- **216.109.117.109**

These are among the IP addresses returned as part of the DNS response when querying for `www.yahoo.com`.

b) How many packets/frames does it take to receive the web page (the answer to the first http get request only)?

To determine how many packets/frames it takes to receive the web page for the first HTTP GET request, we can look at the sequence of TCP and HTTP packets in the capture.

1. **Packet 6:** This is the HTTP GET request sent by the client to the server.
2. **Packet 7 to Packet 10**: These packets are part of the HTTP response from the server.

In this case, the server's response begins in **Packet 7** and continues through Packet 10. The server is sending the data in multiple segments, with each TCP packet carrying part of the response.

Therefore, it takes **4 packets (Packets 7 to 10)** to receive the response to the HTTP GET request.

c) Does this web site use gzip to compress its data for sending? Does it write cookies? In order to answer these questions, look under the payload for the reassembled packet that represents the web page. This will be the last packet from question b above. Look to see if it has “Content-Encoding” set to gzip, and to see if it has a “Set-Cookie” to write a cookie.

Yes

d) What is happening in packets 26 and 27? Does every component of a web page have to come from the same server? See the Hint to the left.

Packets 26 & 27 are solving the DNS for `us.js2.yimg.com` which is a domain serving part of the content for the main page. This page comes from various servers and different components.

e) In packet 37 we see another DNS query, this time for us.i1.yimg.com. Why does the client need to ask for this IP address? Didn't we just get this address in packet 26? (This is a trick question; carefully compare the two common names in packet 26 and 37.)

Packet 26 is likely to be loading JavaScript content while Packet 37 is probably loading page content such as images or other resources.

It is common in modern websites to have this kind of pattern where different types of sources are served from different subdomains. Because of that, separated DNS queries are required for each subdomain.

f) In packet 42 we see a HTTP "Get" statement, and in packet 48 a new HTTP "Get" statement. Why didn't the system need another DNS request before the second get statement? Click on packet 42 and look in the middle window. Expand the line titled "Hypertext Transfer Protocol" and read the "Host:" line. Compare that line to the "Host:" line for packet 48.

Packet 42 & 48 are HTTP GET requests to the same host, therefore, a new DNS request is not required. The system had cached the IP address in the first request so it uses the same one in the new fetching.

g) Examine packet 139. It is one segment of a PDU that is reassembled with several other segments in packet 160. Look at packets 141, 142, and 143. Are these three packets also part of packet 160? What happens if a set of packets that are supposed to be reassembled do not arrive in a continuous stream or do not arrive in the proper order?

Packets 141, 142, and 143 are fragments of the reconstructed PDU in packet 160, continuing the sequence of data segments that originated from packet 139. TCP manages this process by gathering and organizing segments from various packets. When packets arrive out of sequence, TCP stores them in a buffer and waits for the missing segments to complete the reassembly of the message. If any packets are lost, TCP requests retransmission from the sender, ensuring the data is reliably transferred. However, this process can introduce delays as TCP waits for all segments to arrive before proceeding.

h) Return to examine frames 141 and 142. Both of these are graphics (GIF files) from the same source IP address. How does the client know which graphic to match up to each get statement? Hint: Click on each and look in the middle window for the heading line that starts with "Transmission Control Protocol". What difference do you see in the heading lines for the two files? Return to the original "Get" statements. Can you see the same difference in the "Get" statements?

The client uses a mix of sequence numbers, content lengths and ports, also a specific URI from the GET requests to know each graphic response and match it to its request.