

# CSL317 Computer Networks Assignment – Wireshark

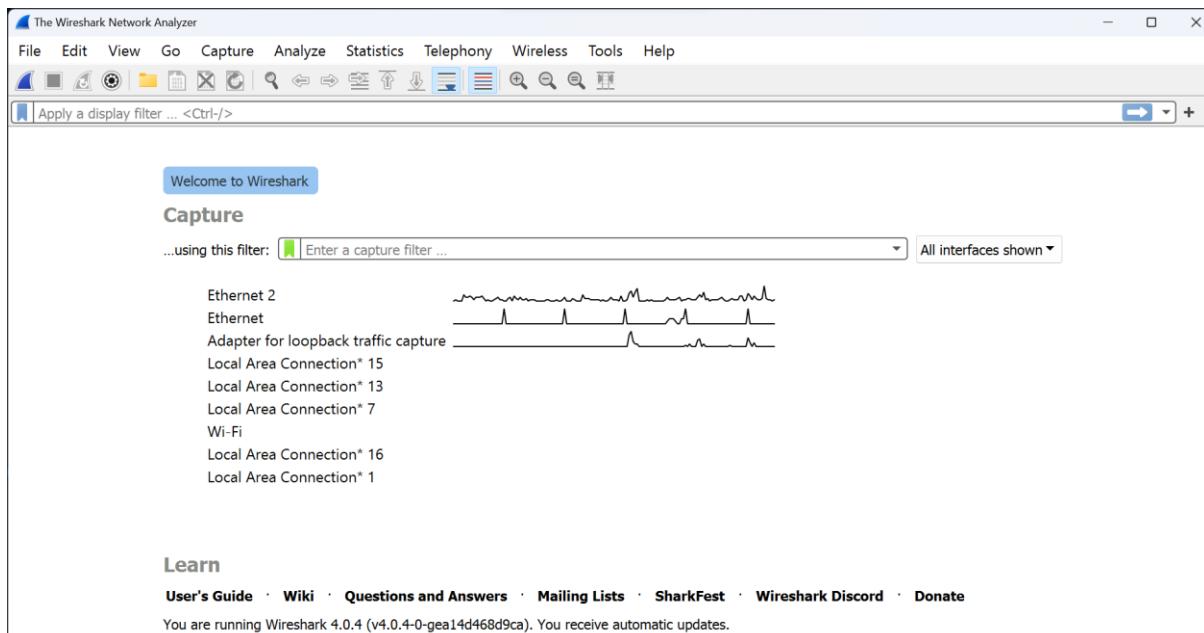
**Name:** Dweeja Reddy Devi

**Enrollment Number:** BT20CSE032

## HTTP Assignment

Begin Wireshark packet capture and enter the following to your browser "<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html>" Your browser should display the very simple, one-line HTML file. Now Stop Wireshark packet capture. By looking at the information in the HTTP GET and response messages, answer the following questions:

Wireshark Interface showing the networks available (both wired and wireless):



1. Is your browser running HTTP version 1.0 or 1.1?

My browser is running the HTTP version 1.1 which can be seen from the GET message of HTTP protocol

Ethernet 2

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
406	4.271286	172.27.45.128	128.119.245.12	HTTP	535	GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
503	4.530942	128.119.245.12	172.27.45.128	HTTP	540	HTTP/1.1 200 OK (text/html)
552	4.945244	172.27.45.128	128.119.245.12	HTTP	481	GET /favicon.ico HTTP/1.1
559	5.205120	128.119.245.12	172.27.45.128	HTTP	538	HTTP/1.1 404 Not Found (text/html)

```
> Frame 406: 535 bytes on wire (4280 bits), 535 bytes captured (4280 bits) on interface \Device\NPF_{...}
> Ethernet II, Src: Shenzhen_21:36:e2 (c8:4d:44:21:36:e2), Dst: HewlettP_18:db:6a (cc:3e:5-...)
> Internet Protocol Version 4, Src: 172.27.45.128, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 51075, Dst Port: 80, Seq: 1, Ack: 1, Len: 481
  Hypertext Transfer Protocol
    GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
      Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
      Accept-Encoding: gzip, deflate\r\n
      Accept-Language: en-US,en;q=0.9,la;q=0.8\r\n
      Connection: keep-alive\r\n
      Host: gaia.cs.umass.edu\r\n
      
```

0000	cc 3e 5f 18 db 6a c8 4d	44 21
0010	02 09 5a ff 40 00 80 06	00 00
0020	f5 0c c7 83 00 50 f6 fe	c7 e1
0030	02 01 51 1b 00 00 47 45	54 20
0040	68 61 72 6b 2d 6c 61 62	73 2f
0050	69 72 65 73 68 61 72 6b	2d 66
0060	74 6d 6c 20 48 54 54 50	2f 31
0070	63 65 70 74 3a 20 74 65	78 74
0080	61 70 70 6c 69 63 61 74	69 6f
0090	6c 2b 78 6d 6c 2c 61 70	70 6c
00a0	6e 2f 78 6d 6c 3b 71 3d	30 2e
00b0	65 2f 61 76 69 66 2c 69	6d 61
00c0	70 2c 69 6d 61 67 65 2f	61 70

## 2.What languages (if any) does your browser indicate that it can accept to the server?

Accept-Languages are: **en-US, en, q=0.9, la; q=0.8**

```
> Frame 406: 535 bytes on wire (4280 bits), 535 bytes captured (4280 bits) on interface \Device\NPF_{...}
> Ethernet II, Src: Shenzhen_21:36:e2 (c8:4d:44:21:36:e2), Dst: HewlettP_18:db:6a (cc:3e:5-...)
> Internet Protocol Version 4, Src: 172.27.45.128, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 51075, Dst Port: 80, Seq: 1, Ack: 1, Len: 481
  Hypertext Transfer Protocol
    GET /wireshark-labs/HTTP-wireshark-file1.html HTTP/1.1\r\n
      Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*
      Accept-Encoding: gzip, deflate\r\n
      Accept-Language: en-US,en;q=0.9,la;q=0.8\r\n
      Connection: keep-alive\r\n
      Host: gaia.cs.umass.edu\r\n
      
```

0000	cc 3e 5f 18 db 6a c8 4d	44 21
0010	02 09 5a ff 40 00 80 06	00 00
0020	f5 0c c7 83 00 50 f6 fe	c7 e1
0030	02 01 51 1b 00 00 47 45	54 20
0040	68 61 72 6b 2d 6c 61 62	73 2f
0050	69 72 65 73 68 61 72 6b	2d 66
0060	74 6d 6c 20 48 54 54 50	2f 31
0070	63 65 70 74 3a 20 74 65	78 74
0080	61 70 70 6c 69 63 61 74	69 6f
0090	6c 2b 78 6d 6c 2c 61 70	70 6c
00a0	6e 2f 78 6d 6c 3b 71 3d	30 2e
00b0	65 2f 61 76 69 66 2c 69	6d 61
00c0	70 2c 69 6d 61 67 65 2f	61 70

## 3.What is the IP address of your computer?

My Computer IP address: **172.27.45.128**

http

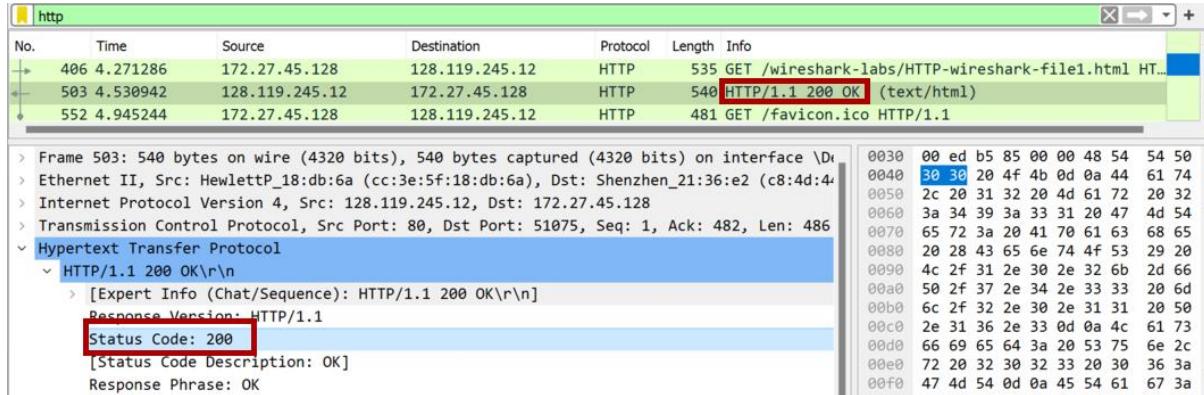
No.	Time	Source	Destination	Protocol	Length	Info
406	4.271286	172.27.45.128	128.119.245.12	HTTP	535	GET /wireshark-labs/HTTP-wireshark-file1.html HT...
503	4.530942	128.119.245.12	172.27.45.128	HTTP	540	HTTP/1.1 200 OK (text/html)
552	4.945244	172.27.45.128	128.119.245.12	HTTP	481	GET /favicon.ico HTTP/1.1

```
> Frame 406: 535 bytes on wire (4280 bits), 535 bytes captured (4280 bits) on interface \Device\NPF_{...}
> Ethernet II, Src: Shenzhen_21:36:e2 (c8:4d:44:21:36:e2), Dst: HewlettP_18:db:6a (cc:3e:5-...)
  Internet Protocol Version 4, Src: 172.27.45.128, Dst: 128.119.245.12
    0100 .... = Version: 4
    ... 0101 = Header Length: 20 bytes (5)
    > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
      Total Length: 521
      Identification: 0x5aff (23295)
    > 010. .... = Flags: 0x2, Don't fragment
      ...0 0000 0000 0000 = Fragment Offset: 0
      Time to Live: 128
      Protocol: TCP (6)
      Header Checksum: 0x0000 [validation disabled]
        [Header checksum status: Unverified]
        Source Address: 172.27.45.128
        Destination Address: 128.119.245.12
    > Transmission Control Protocol, Src Port: 51075, Dst Port: 80, Seq: 1, Ack: 1, Len: 481
  Hypertext Transfer Protocol
```

0010	02 09 5a ff 40 00 80 06	00 00
0020	f5 0c c7 83 00 50 f6 fe	c7 e1
0030	02 01 51 1b 00 00 47 45	54 20
0040	68 61 72 6b 2d 6c 61 62	73 2f
0050	69 72 65 73 68 61 72 6b	2d 66
0060	74 6d 6c 20 48 54 54 50	2f 31
0070	63 65 70 74 3a 20 74 65	78 74
0080	61 70 70 6c 69 63 61 74	69 6f
0090	6c 2b 78 6d 6c 2c 61 70	70 6c
00a0	6e 2f 78 6d 6c 3b 71 3d	30 2e
00b0	65 2f 61 76 69 66 2c 69	6d 61
00c0	70 2c 69 6d 61 67 65 2f	61 70
00d0	3b 71 3d 3e 2e 38 2e 61	70 70
00e0	6f 6e 2f 73 69 67 6e 65	64 2d
00f0	67 65 3b 76 3d 62 33 3b	71 3d
0100	63 63 65 70 74 2d 45 6e	63 6f
0110	67 7a 69 70 2c 20 64 65	66 6c
0120	63 63 65 70 74 2d 4c 61	6e 67
0130	65 6e 2d 55 53 2c 65 6e	3b 71
0140	61 3b 71 3d 30 2e 38 0d	0a 43
0150	69 6f 6e 3a 20 6b 65 65	70 2d

#### 4. What is the status code returned from the server to your browser?

The status code returned is **200 OK** indicates that request has been succeeded i.e resource is fetched and is transmitted in the message body.

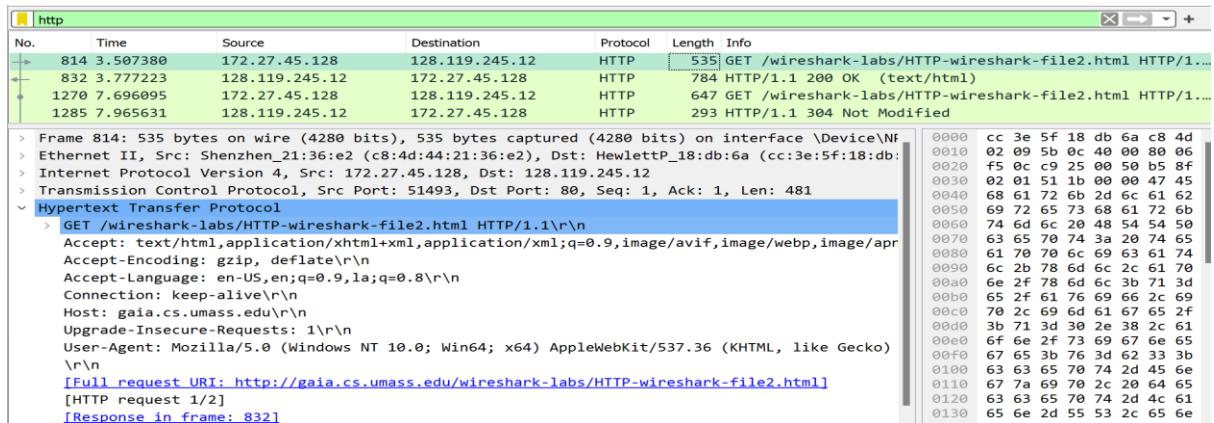


Start up the Wireshark packet sniffer. Enter the following URL into your browser(make sure your browser's cache is empty)

<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html>. Your browser should display a very simple five-line HTML file. Quickly enter the same URL into your browser again (or simply select the refresh button on your browser).Stop Wireshark packet capture.

#### 5. Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an “IF-MODIFIED-SINCE” line in the HTTP GET?

There is **no if-modified-since** line in the first HTTP GET request because the file was requested on the browser for the first time not yet modified



After refreshing, there will be a if-modified-since line because chrome will ping the server for each file regardless of whether or not they are already cached.

No.	Time	Source	Destination	Protocol	Length	Info
814	3.507380	172.27.45.128	128.119.245.12	HTTP	535	GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
832	3.777223	128.119.245.12	172.27.45.128	HTTP	784	HTTP/1.1 200 OK (text/html)
1270	7.696095	172.27.45.128	128.119.245.12	HTTP	647	GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1
1285	7.965631	128.119.245.12	172.27.45.128	HTTP	293	HTTP/1.1 304 Not Modified

```

> Frame 1270: 647 bytes on wire (5176 bits), 647 bytes captured (5176 bits) on interface \Device\NPF_{...}
> Ethernet II, Src: Shenzhen_21:36:e2 (c8:4d:44:21:36:e2), Dst: HewlettP_18:db:6a (cc:3e:5f:18:db:6a)
> Internet Protocol Version 4, Src: 172.27.45.128, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 51493, Dst Port: 80, Seq: 482, Ack: 731, Len: 593
> Hypertext Transfer Protocol
>   GET /wireshark-labs/HTTP-wireshark-file2.html HTTP/1.1\r\n
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apr
Accept-Encoding: gzip, deflate\r\n
Accept-Language: en-US,en;q=0.9,la;q=0.8\r\n
Cache-Control: max-age=0\r\n
Connection: keep-alive\r\n
Host: gaia.cs.umass.edu\r\n
If-Modified-Since: Sun, 12 Mar 2023 06:59:01 GMT\r\n
If-None-Match: "173-5f6ae87804663"\r\n
Upgrade-Insecure-Requests: 1\r\n
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)\r\n
\r\n

```

6. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?

The contents of the file are returned for the first time.

No.	Time	Source	Destination	Protocol	Length	Info
1	3.507380	172.27.45.128	128.119.245.12	HTTP	535	[Request URI: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html] File Data: 371 bytes

```

[Request URI: http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html]
File Data: 371 bytes
< Line-based text data: text/html (10 lines)
  \n
  <html>\n
  \n
  Congratulations again! Now you've downloaded the file lab2-2.html. <br>\n
  This file's last modification date will not change. <p>\n
  Thus if you download this multiple times on your browser, a complete copy <br>\n
  will only be sent once by the server due to the inclusion of the IN-MODIFIED-SINCE<br>\n
  field in your browser's HTTP GET request to the server.\n
  \n
  </html>\n

```

After refreshing, there will be no data returned as earlier response got cached.

No.	Time	Source	Destination	Protocol	Length	Info
1	7.965631	128.119.245.12	172.27.45.128	HTTP	293	HTTP/1.1 304 Not Modified

```

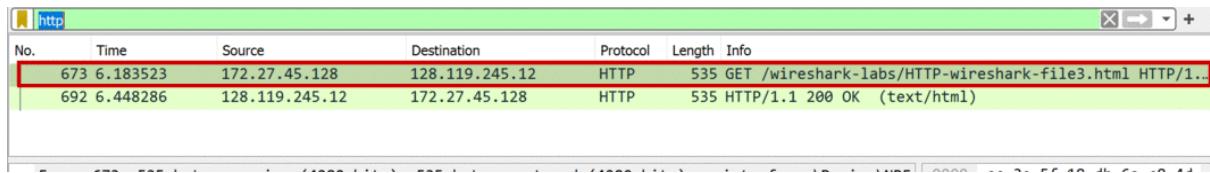
> Ethernet II, Src: HewlettP_18:db:6a (cc:3e:5f:18:db:6a), Dst: Shenzhen_21:36:e2 (c8:4d:44:21:36:e2)
> Internet Protocol Version 4, Src: 128.119.245.12, Dst: 172.27.45.128
> Transmission Control Protocol, Src Port: 80, Dst Port: 51493, Seq: 731, Ack: 1075, Len: 239
> Hypertext Transfer Protocol
>   HTTP/1.1 304 Not Modified\r\n
Date: Sun, 12 Mar 2023 11:14:09 GMT\r\n
Server: Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips PHP/7.4.33 mod_perl/2.0.11 Perl/v5.16.3\r\n
Connection: Keep-Alive\r\n
Keep-Alive: timeout=5, max=99\r\n
ETag: "173-5f6ae87804663"\r\n
\r\n
[HTTP response 2/2]
[Time since request: 0.269536000 seconds]
[Prev request in frame: 814]
[Prev response in frame: 832]
[Request in frame: 1270]

```

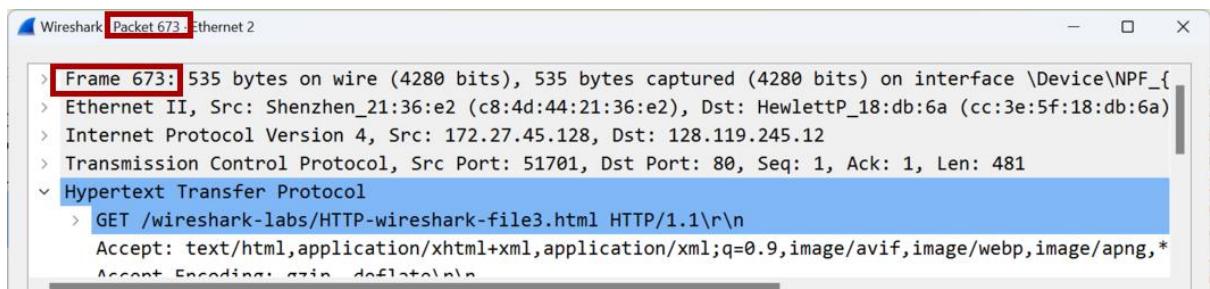
Start up the Wireshark packet sniffer. Enter the following URL into your browser(make sure your browser's cache is empty)  
<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file3.html>. Your browser should display the lengthy US Bill of Rights. Stop Wireshark packet capture, and enter “http” in the display-filter-specification window, so that only captured HTTP messages will be displayed.

7. How many HTTP GET request messages did your browser send? Which packet number in the trace contains the GET message for the Bill or Rights?

Browser sent 1 HTTP GET request

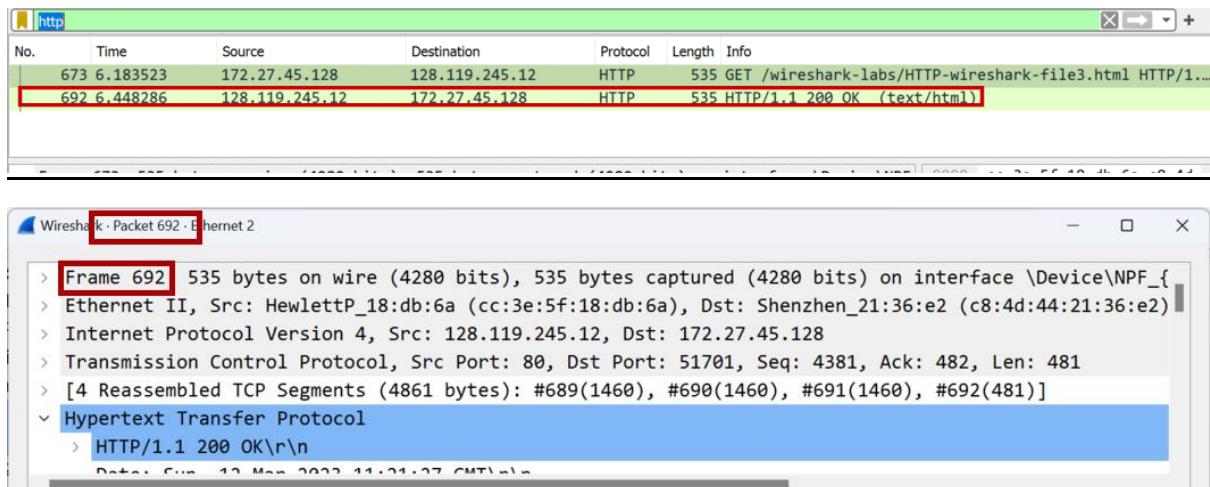


Packet number **673** contains the GET message for the Bill or Rights



8. Which packet number in the trace contains the status code and phrase associated with the response to the HTTP GET request?

Packet number **692** in trace contains the status code.



Start up the Wireshark packet sniffer. Enter the following URL into your browser (make sure your browser's cache is empty)

<http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file4.html> Your browser should display a short HTML file with two images. Stop Wireshark packet capture

9. How many HTTP GET request messages did your browser send? To which Internet addresses were these GET requests sent?

Browser sent **3** HTTP GET requests to the IP address **128.119.245.12**

No.	Time	Source	Destination	Protocol	Length	Info
332	3.798293	172.27.45.128	128.119.245.12	HTTP	535	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1...
342	4.066604	128.119.245.12	172.27.45.128	HTTP	1355	HTTP/1.1 200 OK (text/html)
346	4.174856	172.27.45.128	128.119.245.12	HTTP	481	GET /pearson.png HTTP/1.1
412	4.443074	128.119.245.12	172.27.45.128	HTTP	745	HTTP/1.1 200 OK (PNG)
500	4.773965	172.27.45.128	178.79.137.164	HTTP	448	GET /8E_cover_small.jpg HTTP/1.1
532	4.928364	178.79.137.164	172.27.45.128	HTTP	225	HTTP/1.1 301 Moved Permanently

10. Can you tell whether your browser downloaded the two images serially, or whether they were downloaded from the two websites in parallel? Explain.

The images were downloaded **serially** and it can be explained by their packet arrival times. Both the times are different, hence cannot be parallel.

No.	Time	Source	Destination	Protocol	Length	Info
332	3.798293	172.27.45.128	128.119.245.12	HTTP	535	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1...
342	4.066604	128.119.245.12	172.27.45.128	HTTP	1355	HTTP/1.1 200 OK (text/html)
346	4.174856	172.27.45.128	128.119.245.12	HTTP	481	GET /pearson.png HTTP/1.1
412	4.443074	128.119.245.12	172.27.45.128	HTTP	745	HTTP/1.1 200 OK (PNG)
500	4.773965	172.27.45.128	178.79.137.164	HTTP	448	GET /8E_cover_small.jpg HTTP/1.1
532	4.928364	178.79.137.164	172.27.45.128	HTTP	225	HTTP/1.1 301 Moved Permanently

No.	Time	Source	Destination	Protocol	Length	Info
332	3.798293	172.27.45.128	128.119.245.12	HTTP	535	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1...
342	4.066604	128.119.245.12	172.27.45.128	HTTP	1355	HTTP/1.1 200 OK (text/html)
346	4.174856	172.27.45.128	128.119.245.12	HTTP	481	GET /pearson.png HTTP/1.1
412	4.443074	128.119.245.12	172.27.45.128	HTTP	745	HTTP/1.1 200 OK (PNG)
500	4.773965	172.27.45.128	178.79.137.164	HTTP	448	GET /8E_cover_small.jpg HTTP/1.1
532	4.928364	178.79.137.164	172.27.45.128	HTTP	225	HTTP/1.1 301 Moved Permanently

No.	Time	Source	Destination	Protocol	Length	Info
332	3.798293	172.27.45.128	128.119.245.12	HTTP	535	GET /wireshark-labs/HTTP-wireshark-file4.html HTTP/1...
342	4.066604	128.119.245.12	172.27.45.128	HTTP	1355	HTTP/1.1 200 OK (text/html)
346	4.174856	172.27.45.128	128.119.245.12	HTTP	481	GET /pearson.png HTTP/1.1
412	4.443074	128.119.245.12	172.27.45.128	HTTP	745	HTTP/1.1 200 OK (PNG)
500	4.773965	172.27.45.128	178.79.137.164	HTTP	448	GET /8E_cover_small.jpg HTTP/1.1
532	4.928364	178.79.137.164	172.27.45.128	HTTP	225	HTTP/1.1 301 Moved Permanently

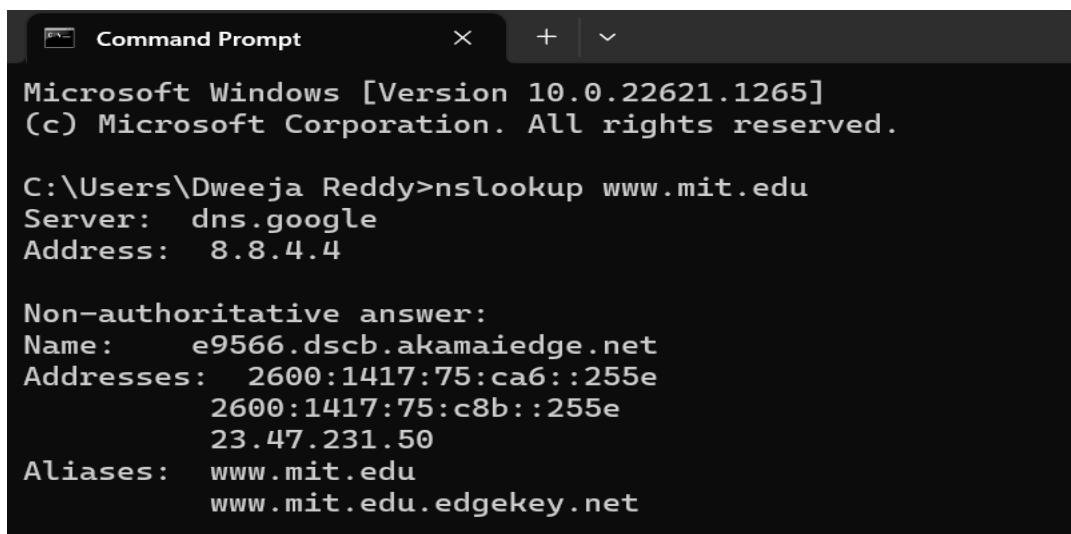
## DNS Assignment

**nslookup:** nslookup tool allows the host running the tool to query any specified DNS server for a DNS record. The queried DNS server can be a root DNS server, a top - level domain DNS server, an authoritative DNS server, or an intermediate DNS server. To accomplish this task, nslookup sends a DNS query to the specified DNS server, receives a DNS reply from that same DNS server, and displays the result.

1.Run ‘nslookup www.mit.edu’ on your command prompt and what will be the name and IP address of the DNS server that provides the answer?

**Name: e9566.dscb.akamaiedge.net**

**IP Address: 23.47.231.50**



```
Command Prompt      X + ▾
Microsoft Windows [Version 10.0.22621.1265]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Deeja Reddy>nslookup www.mit.edu
Server: dns.google
Address: 8.8.4.4

Non-authoritative answer:
Name: e9566.dscb.akamaiedge.net
Addresses: 2600:1417:75:ca6::255e
           2600:1417:75:c8b::255e
           23.47.231.50
Aliases: www.mit.edu
         www.mit.edu.edgekey.net
```

2.Run ‘nslookup –type=NS mit.edu’ on your command prompt and what will be the host names of the authoritative DNS for mit.edu.

mit.edu nameserver = asia1.akam.net

mit.edu nameserver = ns1-37.akam.net

mit.edu nameserver = eur5.akam.net

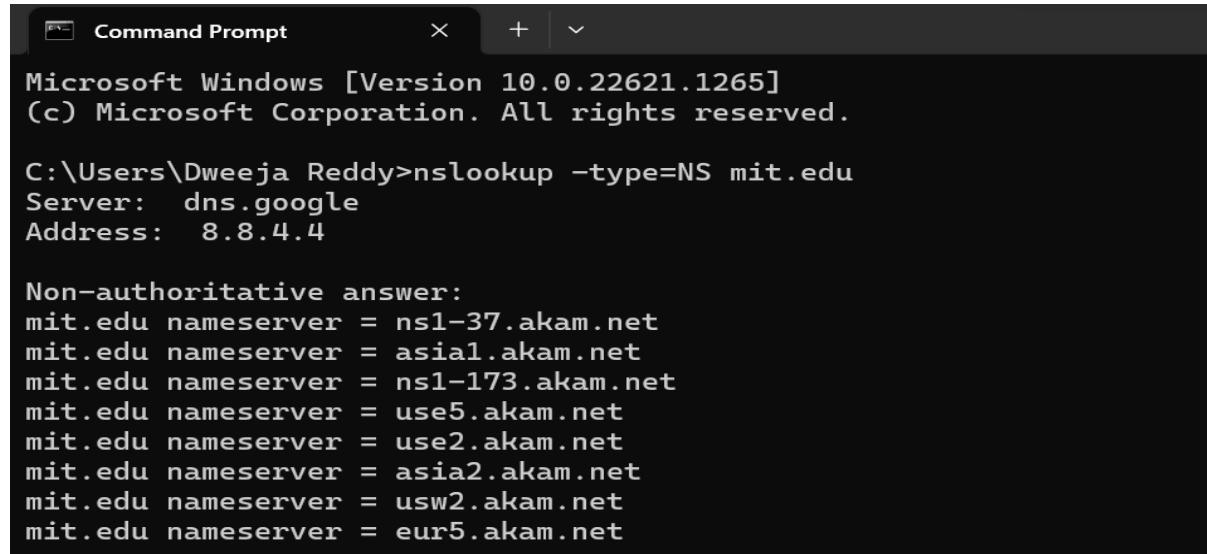
mit.edu nameserver = asia2.akam.net

mit.edu nameserver = ns1-173.akam.net

mit.edu nameserver = use2.akam.net

mit.edu nameserver = use5.akam.net

mit.edu nameserver = usw2.akam.net



```
Command Prompt Microsoft Windows [Version 10.0.22621.1265]
(c) Microsoft Corporation. All rights reserved.

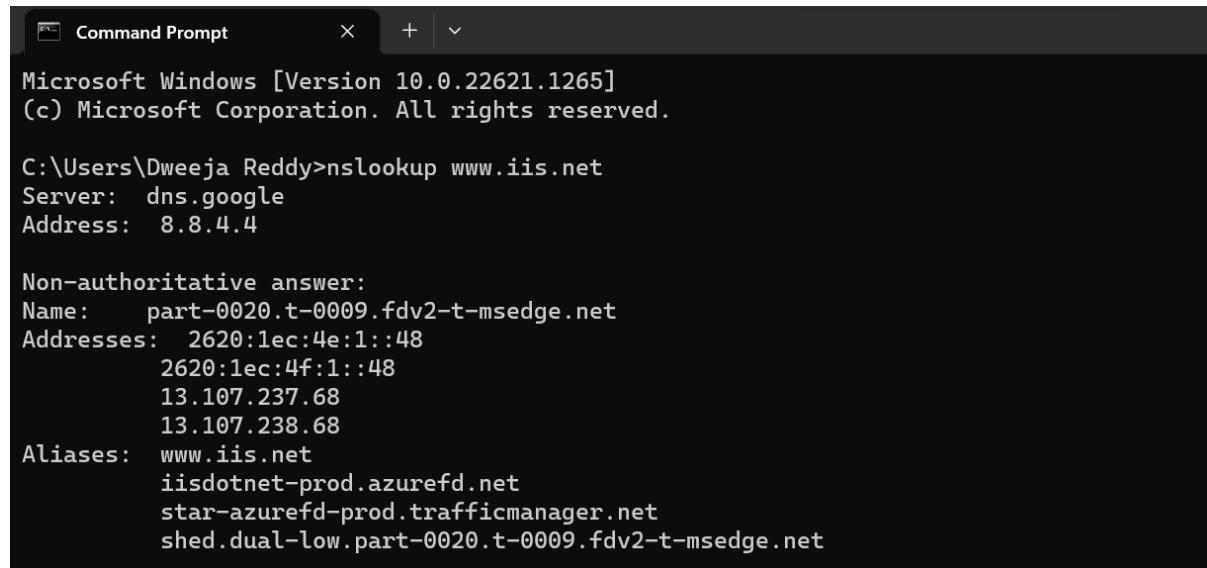
C:\Users\DWEEJA REDDY>nslookup -type=NS mit.edu
Server: dns.google
Address: 8.8.4.4

Non-authoritative answer:
mit.edu nameserver = ns1-37.akam.net
mit.edu nameserver = asia1.akam.net
mit.edu nameserver = ns1-173.akam.net
mit.edu nameserver = use5.akam.net
mit.edu nameserver = use2.akam.net
mit.edu nameserver = asia2.akam.net
mit.edu nameserver = usw2.akam.net
mit.edu nameserver = eur5.akam.net
```

3.Run nslookup to obtain the IP address of a Web server in Asia. What is the IP address of that server?

I took [www.iis.net](http://www.iis.net) as Asian webserver to test nslookup.

**IP address of that server: 13.107.238.68**



```
Command Prompt Microsoft Windows [Version 10.0.22621.1265]
(c) Microsoft Corporation. All rights reserved.

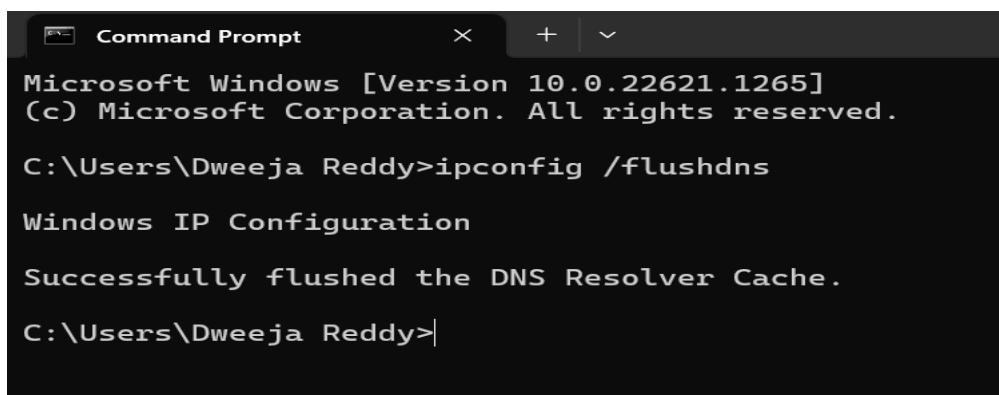
C:\Users\DWEEJA REDDY>nslookup www.iis.net
Server: dns.google
Address: 8.8.4.4

Non-authoritative answer:
Name: part-0020.t-0009.fdv2-t-msedge.net
Addresses: 2620:1ec:4e:1::48
           2620:1ec:4f:1::48
           13.107.237.68
           13.107.238.68
Aliases: www.iis.net
          iisdotnet-prod.azurefd.net
          star-azurefd-prod.trafficmanager.net
          shed.dual-low.part-0020.t-0009.fdv2-t-msedge.net
```

ipconfig: ipconfig (for Windows) and ifconfig (for Linux/Unix) are among the most useful little utilities in your host, especially for debugging network issues. ipconfig can be used to show your current TCP/IP information, including your

address, DNS server addresses, adapter type and so on. For example, you can get all this information about your host simply by entering ‘ipconfig \all’ into the Command Prompt. ‘ipconfig /displaydns’ -->Each entry shows the remaining Time to Live (TTL) in seconds. To clear the cache, enter ‘ipconfig /flushdns’ Flushing the DNS cache clears all entries and reloads the entries from the hosts file. Now that we are familiar with nslookup and ipconfig, firstly now capture the DNS packets that are generated by ordinary Websurfing activity.

- Use ipconfig to empty the DNS cache in your host.
- Open your browser and empty your browser cache.
- Open Wireshark and enter “ip.addr == ” into the filter, where you obtain your\_IP\_address with ipconfig. This filter removes all packets that neither originate nor are destined to your host.
- Start packet capture in Wireshark.
- With your browser, visit the Web page: <http://www.ietf.org>
- Stop packet capture.



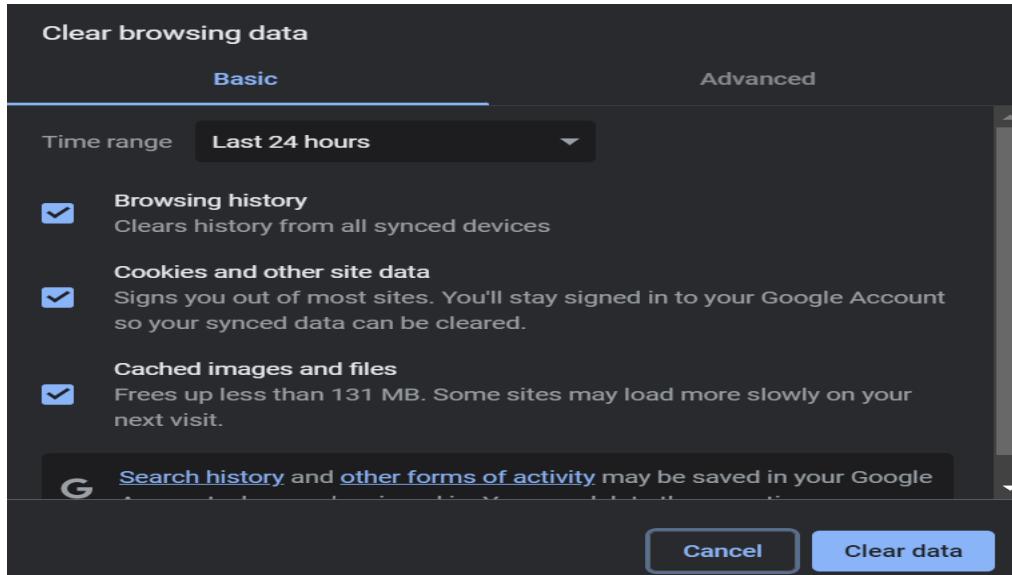
```
Command Prompt
Microsoft Windows [Version 10.0.22621.1265]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Deewija Reddy>ipconfig /flushdns

Windows IP Configuration

Successfully flushed the DNS Resolver Cache.

C:\Users\Deewija Reddy>
```



```

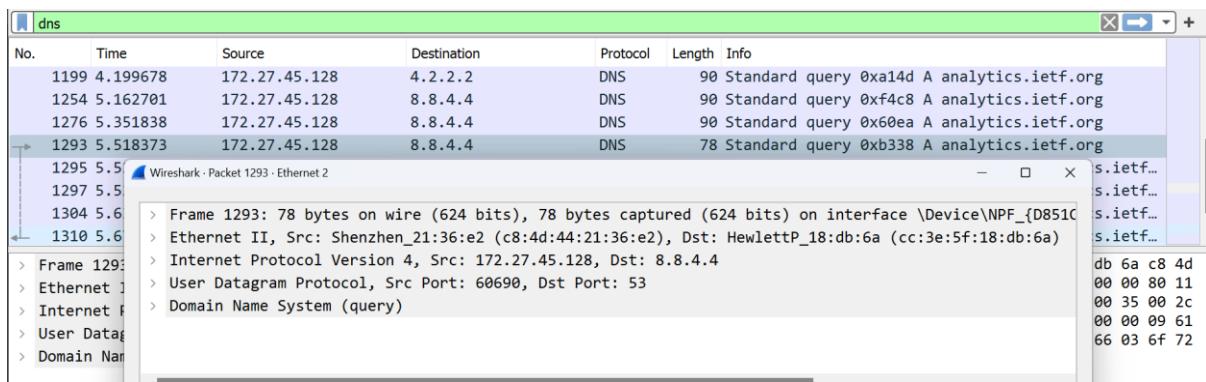
Ethernet adapter Ethernet 2:

Connection-specific DNS Suffix . :
Link-local IPv6 Address . . . . . : fe80::122b:66b2:be99:2c84%56
IPv4 Address. . . . . : 172.27.45.128
Subnet Mask . . . . . : 255.255.240.0
Default Gateway . . . . . : fe80::2a3b:82ff:fe2a:7ab5%56
                                         172.27.32.1

```

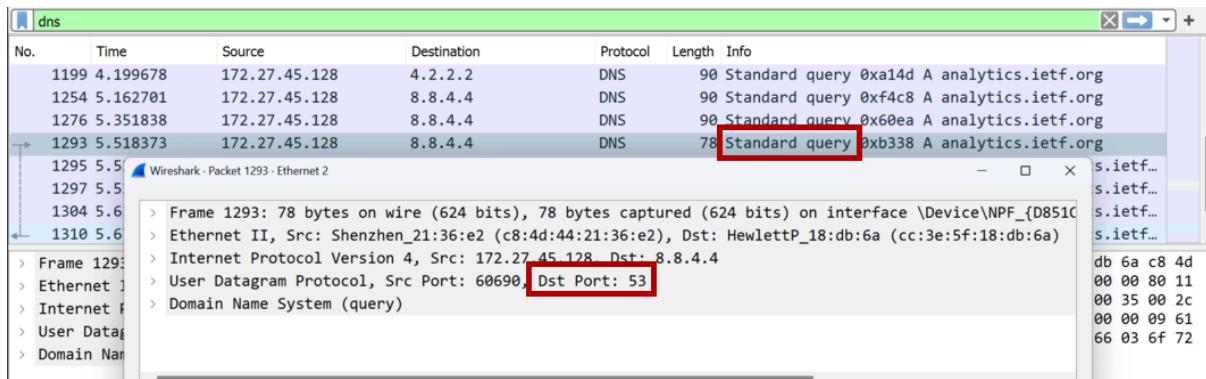
4. Locate the DNS query and response messages. Are they sent over UDP or TCP?

These are sent over UDP



5. What is the destination port for the DNS query message? What is the source port of the DNS response message?

Destination port for DNS query : 53



## Source port for DNS response : 53

No.	Time	Source	Destination	Protocol	Length	Info
1276	5.351838	172.27.45.128	8.8.4.4	DNS	90	Standard query 0x60ea A analytics.ietf.org
1293	5.518373	172.27.45.128	8.8.4.4	DNS	78	Standard query 0xb338 A analytics.ietf.org
1295	5.520438	4.2.2.2	172.27.45.128	DNS	108	Standard query response 0x948f A analytics.ietf...
1297	5.523355	4.2.2.2	172.27.45.128	DNS	108	Standard query response 0xa14d A analytics.ietf...
1304	5.639575	8.8.4.4	172.27.45.128	DNS	108	Standard query response 0x60ea A analytics.ietf...
1310	5.670842	8.8.4.4	172.27.45.128	DNS	94	Standard query response 0xb338 A analytics.ietf...
1311	5.67791					

Wireshark - Packet 1310 - Ethernet 2

```

> Frame 1310: 94 bytes on wire (752 bits), 94 bytes captured (752 bits) on interface \Device\NPF_{D85...
> Ethernet II, Src: HewlettP_18:db:6a (cc:3e:5f:18:db:6a), Dst: Shenzhen_21:36:e2 (c8:4d:44:21:36:e2)
> Internet Protocol Version 4, Src: 8.8.4.4, Dst: 172.27.45.128
> User Datagram Protocol, Src Port: 53, Dst Port: 60690
    Source Port: 53
    Destination Port: 60690
    Length: 60
    Checksum: 0x100F [Calculated]

```

6. To what IP address is the DNS query message sent? Use ipconfig to determine the IP address of your local DNS server. Are these two IP addresses the same?

DNS query message is sent to the IP address: 8.8.4.4 which is also one of the DNS servers IP addresses

No.	Time	Source	Destination	Protocol	Length	Info
300	2.387826	172.27.45.128	4.2.2.2	DNS	84	Standard query 0xea81 A www.ietf.org
325	2.465892	4.2.2.2	172.27.45.128	DNS	163	Standard query response 0xea81 A www.ietf.org C...
371	2.628295	172.27.45.128	4.2.2.2	DNS	84	Standard query 0xb2eb A www.ietf.org
376	2.636700	172.27.45.128	4.2.2.2	DNS	84	Standard query 0x72a5 HTTPS www.ietf.org
386	2.698112	4.2.2.2	172.27.45.128	DNS	163	Standard query response 0xb2eb A www.ietf.org C...
399	2.729892	4.2.2.2	172.27.45.128	DNS	210	Standard query response 0x72a5 HTTPS www.ietf.o...
1065	3.182884	172.27.45.128	4.2.2.2	DNS	90	Standard query 0xfe1c HTTPS analytics.ietf.org
1869	3.186036	172.27.45.128	4.2.2.2	DNS	90	Standard query 0x948f A analytics.ietf.org
1189	4.144218	172.27.45.128	8.8.4.4	DNS	90	Standard query 0x4954 HTTPS analytics.ietf.org
1193	4.198996	8.8.4.4	172.27.45.128	DNS	145	Standard query response 0x4954 HTTPS analytics...
1199	4.199678	172.27.45.128	4.2.2.2	DNS	90	Standard query 0xa14d A analytics.ietf.org
1254	5.162701	172.27.45.128	8.8.4.4	DNS	90	Standard query 0xf4c8 A analytics.ietf.org
1276	5.351838	172.27.45.128	8.8.4.4	DNS	90	Standard query 0x60ea A analytics.ietf.org
1293	5.518373	172.27.45.128	8.8.4.4	DNS	78	Standard query 0xb338 A analytics.ietf.org
1295	5.520438	4.2.2.2	172.27.45.128	DNS	108	Standard query response 0x948f A analytics.ietf...
1297	5.523355	4.2.2.2	172.27.45.128	DNS	108	Standard query response 0xa14d A analytics.ietf...

```

DHCP Server . . . . . : 172.27.32.1
DHCPv6 IAID . . . . . : 214453572
DHCPv6 Client DUID. . . . . : 00-01-00-01-27-94-DC-FA-28-CD-C4-F5-EB-A3
DNS Servers . . . . . : 8.8.4.4
                           4.2.2.2
NetBIOS over Tcpip. . . . . : Enabled

```

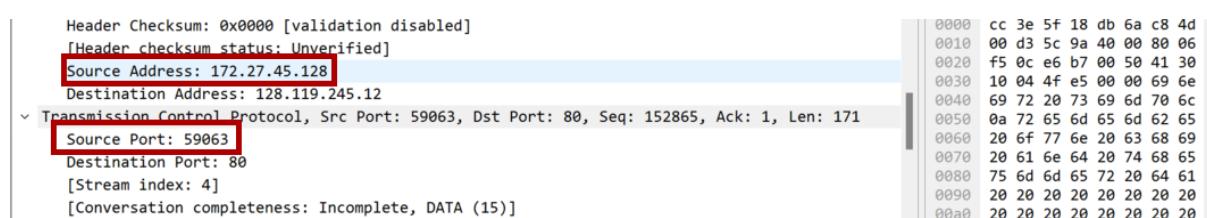
## TCP Assignment

- Start up your web browser. Go the <http://gaia.cs.umass.edu/wiresharklabs/alice.txt> and retrieve an ASCII copy of Alice in Wonderland. Store this file somewhere on your computer.
- Next go to <http://gaia.cs.umass.edu/wireshark-labs/TCP-wiresharkfile1.html>.
- Use the Browse button in this form to enter the name of the file (full path name) on your computer containing Alice in Wonderland (or do so manually). Don't yet press the "Upload alice.txt file" button.
- Now start up Wireshark and begin packet capture (Capture->Start) and then press OK on the Wireshark Packet Capture Options screen (we'll not need to select any options here).
- Returning to your browser, press the "Upload alice.txt file" button to upload the file to the gaia.cs.umass.edu server. Once the file has been uploaded, a short congratulations message will be displayed in your browser window.
- Stop Wireshark packet capture.
- Before analyzing the behavior of the TCP connection in detail, let's take a high level view of the trace. First, filter the packets displayed in the Wireshark window by entering "tcp"

1.What is the IP address and TCP port number used by the client computer (source) that is transferring the file to gaia.cs.umass.edu?

Source IP address: 172.27.45.128

Source Port: 59063



2.What is the IP address of gaia.cs.umass.edu? On what port number is it sending and receiving TCP segments for this connection?

Destination IP address: 128.119.245.12

Destination Port: 80

Header Checksum: 0x0000 [validation disabled] [Header checksum status: Unverified]	0000 cc 3e 5f 18 db 6a c8 4d
Source IP: 172.27.45.129	0010 00 d3 5c 9a 40 00 80 06
Destination Address: 128.119.245.12	0020 f5 0c e6 b7 00 50 41 30
Transmission Control Protocol, Src Port: 59063, Dst Port: 80, Seq: 152865, Ack: 1, Len: 171	0030 10 04 4f e5 00 00 69 6e
Source Port: 59063	0040 69 72 20 73 69 6d 70 6c
Destination Port: 80	0050 0a 72 65 6d 65 6d 62 65
[Stream index: 4]	0060 20 6f 77 6e 20 63 68 69
[Conversation completeness: Incomplete, DATA (15)]	0070 20 61 6e 64 20 74 68 65
	0080 75 6d 6d 65 72 20 64 61
	0090 20 20 20 20 20 20 20 20
	00a0 20 20 20 20 20 20 20 20

3.What is the sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu? What is it in the segment that identifies the segment as a SYN segment?

The sequence number of the TCP SYN segment that is used to initiate the TCP connection between the client computer and gaia.cs.umass.edu is 0.

Segment can be identified as a SYN segment if SYN flag is set to 1.

No.	Time	Source	Destination	Protocol	Length	Info	
743	7.000066	128.119.245.12	172.27.45.128	TCP	60	80 → 59063 [ACK] Seq=1 Ack=144105 Win=235776 Len=0	
744	7.003648	150.171.44.254	172.27.45.128	TCP	60	443 → 59070 [ACK] Seq=813 Ack=1022 Win=4193536 Len=0	
747	7.037371	172.27.45.128	150.171.44.254	TLSv1.2	136	Application Data	
748	7.038823	172.27.45.128	23.212.5.41	TCP	66	59072 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=146	
750	7.070570	23.212.5.41	172.27.45.128	TCP	66	443 → 59072 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0	
751	7.070883	172.27.45.128	23.212.5.41	TCP	54	59072 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0	
752	7.071817	172.27.45.128	23.212.5.41	TLSv1.3	626	Client Hello	
753	7.098166	23.212.5.41	172.27.45.128	TCP	60	443 → 59072 [ACK] Seq=1 Ack=573 Win=64128 Len=0	
754	7.098974	23.212.5.41	172.27.45.128	TLSv1.3	318	Server Hello, Change Cipher Spec, Application Data	
755	7.099156	172.27.45.128	23.212.5.41	TCP	54	59072 → 443 [ACK] Seq=573 Ack=265 Win=261632 Len=0	
756	7.100510	172.27.45.128	23.212.5.41	TLSv1.3	134	Change Cipher Spec, Application Data	

Flags: 0x002 (SYN)

- 000. .... . = Reserved: Not set
- ...0 .... . = Accurate ECN: Not set
- .... 0.... . = Congestion Window Reduced: Not set
- .... .0.. . = ECN-Echo: Not set
- .... ..0. . = Urgent: Not set
- .... ...0 . = Acknowledgment: Not set
- .... ....0 . = Push: Not set
- .... .... .0 . = Reset: Not set
- .... .... ..1. . = Syn: Set

0000 cc 3e 5f 18 db 6a c8 4d
0010 00 34 18 ae 40 00 80 06
0020 05 29 e6 c0 01 bb 97 ec
0030 ff ff f6 be 00 00 02 04
0040 04 02

4. Answer the following:

- What is the sequence number of the SYNACK segment sent by gaia.cs.umass.edu to the client computer in reply to the SYN?

Sequence number of the SYNACK segment from gaia.cs.umass.edu to the client computer in reply to the SYN is 0 in this trace

No.	Time	Source	Destination	Protocol	Length	Info
743	7.000066	128.119.245.12	172.27.45.128	TCP	60	80 → 59063 [ACK] Seq=1 Ack=144105 Win=235776 Len=0
744	7.003648	150.171.44.254	172.27.45.128	TCP	60	443 → 59070 [ACK] Seq=813 Ack=1022 Win=4193536 Len=0
747	7.037371	172.27.45.128	150.171.44.254	TLSv1.2	136	Application Data
748	7.038823	172.27.45.128	23.212.5.41	TCP	66	59072 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=146..
750	7.070570	23.212.5.41	172.27.45.128	TCP	66	443 → 59072 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0
751	7.070883	172.27.45.128	23.212.5.41	TCP	54	59072 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
752	7.071817	172.27.45.128	23.212.5.41	TLSv1.3	626	Client Hello
753	7.098166	23.212.5.41	172.27.45.128	TCP	60	443 → 59072 [ACK] Seq=1 Ack=573 Win=64128 Len=0
754	7.098974	23.212.5.41	172.27.45.128	TLSv1.3	318	Server Hello, Change Cipher Spec, Application Data
755	7.099156	172.27.45.128	23.212.5.41	TCP	54	59072 → 443 [ACK] Seq=573 Ack=265 Win=261632 Len=0
756	7.100510	172.27.45.128	23.212.5.41	TLSv1.3	124	Change Cipher Spec, Application Data

b.What is the value of the Acknowledgement field in the SYNACK segment?

The value of the Acknowledgement field in the SYNACK segment is 1.

No.	Time	Source	Destination	Protocol	Length	Info
743	7.000066	128.119.245.12	172.27.45.128	TCP	60	80 → 59063 [ACK] Seq=1 Ack=144105 Win=235776 Len=0
744	7.003648	150.171.44.254	172.27.45.128	TCP	60	443 → 59070 [ACK] Seq=813 Ack=1022 Win=4193536 Len=0
747	7.037371	172.27.45.128	150.171.44.254	TLSv1.2	136	Application Data
748	7.038823	172.27.45.128	23.212.5.41	TCP	66	59072 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=146..
750	7.070570	23.212.5.41	172.27.45.128	TCP	66	443 → 59072 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0
751	7.070883	172.27.45.128	23.212.5.41	TCP	54	59072 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
752	7.071817	172.27.45.128	23.212.5.41	TLSv1.3	626	Client Hello
753	7.098166	23.212.5.41	172.27.45.128	TCP	60	443 → 59072 [ACK] Seq=1 Ack=573 Win=64128 Len=0
754	7.098974	23.212.5.41	172.27.45.128	TLSv1.3	318	Server Hello, Change Cipher Spec, Application Data
755	7.099156	172.27.45.128	23.212.5.41	TCP	54	59072 → 443 [ACK] Seq=573 Ack=265 Win=261632 Len=0
756	7.100510	172.27.45.128	23.212.5.41	TLSv1.3	124	Change Cipher Spec, Application Data

c.How did gaia.cs.umass.edu determine that value?

The value of the ACKnowledgement field in the SYNACK segment is determined by gaia.cs.umass.edu by adding 1 to the initial sequence number of SYN segment from the client computer (i.e. the sequence number of the SYN segment initiated by the client computer is 0.)

d. What is it in the segment that identifies the segment as a SYNACK segment?

No.	Time	Source	Destination	Protocol	Length	Info
743	7.000066	128.119.245.12	172.27.45.128	TCP	60	80 → 59063 [ACK] Seq=1 Ack=144105 Win=235776 Len=0
744	7.003648	150.171.44.254	172.27.45.128	TCP	60	443 → 59070 [ACK] Seq=813 Ack=1022 Win=4193536 Len=0
747	7.037371	172.27.45.128	150.171.44.254	TLSv1.2	136	Application Data
748	7.038823	172.27.45.128	23.212.5.41	TCP	66	59072 → 443 [SYN] Seq=0 Win=65535 Len=0 MSS=146..
750	7.070570	23.212.5.41	172.27.45.128	TCP	66	443 → 59072 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0
751	7.070883	172.27.45.128	23.212.5.41	TCP	54	59072 → 443 [ACK] Seq=1 Ack=1 Win=262144 Len=0
752	7.071817	172.27.45.128	23.212.5.41	TLSv1.3	626	Client Hello
753	7.098166	23.212.5.41	172.27.45.128	TCP	60	443 → 59072 [ACK] Seq=1 Ack=573 Win=64128 Len=0
754	7.098974	23.212.5.41	172.27.45.128	TLSv1.3	318	Server Hello, Change Cipher Spec, Application Data
755	7.099156	172.27.45.128	23.212.5.41	TCP	54	59072 → 443 [ACK] Seq=573 Ack=265 Win=261632 Len=0
756	7.100510	172.27.45.128	23.212.5.41	TLSv1.3	124	Change Cipher Spec, Application Data

000. .... = Reserved: Not set ...0 .... = Accurate ECN: Not set .... 0.... = Congestion Window Reduced: Not set .... .0.... = ECN-Echo: Not set .... 0.... = Urgent: Not set .... .1.... = Acknowledgment: Set .... .... 0.... = Push: Not set .... .... 0.... = Reset: Not set > .... .... 1.... = Syn: Set .... .... 0 = Fin: Not set	0000 c8 4d 44 21 36 e2 cc 3e 0010 00 34 00 00 40 00 3d 06 0020 2d 80 01 bb e6 c0 68 71 0030 fa f0 0c f8 00 00 02 04 0040 03 07
--	--

The SYN flag and Acknowledgement flag in the segment are set to 1 and they indicate that this segment is a SYNACK segment.

5. What is the sequence number of the TCP segment containing the HTTP POST command? Note that in order to find the POST command; you'll need to dig into the packet content field at the bottom of the Wireshark window, looking for a segment with a "POST" within its DATA field.

Sequence number of the TCP segment containing the HTTP POST command is 152865

737 6.999435	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=151405 Ack=1 Win=1049600 Len=0
738 6.999435	172.27.45.128	128.119.245.12	HTTP	225 POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 ..
739 7.000066	128.119.245.12	172.27.45.128	TCP	60 80 → 59063 [ACK] Seq=1 Ack=138265 Win=224128 Len=0
740 7.000066	128.119.245.12	172.27.45.128	TCP	60 80 → 59063 [ACK] Seq=1 Ack=139725 Win=227072 Len=0
741 7.000066	128.119.245.12	172.27.45.128	TCP	60 80 → 59063 [ACK] Seq=1 Ack=141185 Win=230016 Len=0
742 7.000066	128.119.245.12	172.27.45.128	TCP	60 80 → 59063 [ACK] Seq=1 Ack=142645 Win=232832 Len=0
743 7.000066	128.119.245.12	172.27.45.128	TCP	60 80 → 59063 [ACK] Seq=1 Ack=144105 Win=235776 Len=0
744 7.000066	128.119.245.12	172.27.45.128	TCP	60 80 → 59063 [ACK] Seq=1 Ack=145526 Win=238720 Len=0
▼ Transmission Control Protocol, Src Port: 59063, Dst Port: 80 Seq: 152865, Ack: 1, Len: 171				
Source Port: 59063 Destination Port: 80 [Stream index: 4] [Conversation completeness: Incomplete, DATA (15)] [TCP Segment Len: 171]				
Sequence Number: 152865 (relative sequence number) Sequence Number (raw): 1093715021 [Next Sequence Number: 153036 (relative sequence number)] Acknowledgment Number: 1 (relative ack number)				
0020 f5 0c e6 b7 00 50 41 30 0030 10 04 4f e5 00 00 69 6e 0040 69 72 20 73 69 6d 70 6c 0050 0a 72 65 6d 65 6d 62 65 0060 20 6f 77 6e 20 63 68 69 0070 20 61 6e 64 20 74 68 65 0080 75 6d 66 65 72 20 64 61 0090 20 20 20 20 20 20 20 20 00a0 20 20 20 20 20 20 20 20 00b0 45 4e 44 0d 0a 2d 2d 2d 00c0 74 46 6f 72 6d 42 6f 75 00d0 46 34 57 6h 6h 75 7d 4d				

6. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection:

a. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)?

The sequence numbers of first six segments in TCP are

1,1025,2485,3945,5405,6865

No.	Time	Source	Destination	Protocol	Length	Info
147 3.446706	172.27.45.128	128.119.245.12	TCP	66 59063 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460..		
233 3.722788	172.27.45.128	128.119.245.12	TCP	54 59063 → 80 [ACK] Seq=1 Ack=1 Win=1049600 Len=0		
236 3.723408	172.27.45.128	128.119.245.12	TCP	1078 59063 → 80 [PSH, ACK] Seq=1 Ack=1 Win=1049600 Len=0		
237 3.724246	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=1225 Ack=1 Win=1049600 Len=0		
238 3.724246	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=2485 Ack=1 Win=1049600 Len=0		
239 3.724246	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=3945 Ack=1 Win=1049600 Len=0		
240 3.724246	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=5405 Ack=1 Win=1049600 Len=0		
241 3.724246	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=6865 Ack=1 Win=1049600 Len=0		
242 3.724246	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=8325 Ack=1 Win=1049600 Len=0		
243 3.724246	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=9785 Ack=1 Win=1049600 Len=0		
244 3.724246	172.27.45.128	128.119.245.12	TCP	1514 59063 → 80 [ACK] Seq=11245 Ack=1 Win=1049600 Len=0		

The sequence number of the segment containing HTTP POST is **152865**

No.	Time	Source	Destination	Protocol	Length	Info	
735	6.999310	128.119.245.12	172.27.45.128	TCP	60	80 → 59063 [ACK] Seq=1 Ack=135345 Win=218240 Len...	
736	6.999310	128.119.245.12	172.27.45.128	TCP	60	80 → 59063 [ACK] Seq=1 Ack=136805 Win=221184 Len...	
737	6.999435	172.27.45.128	128.119.245.12	TCP	1514	59063 → 80 [ACK] Seq=1405 Ack=1 Win=1849600 Len...	
738	6.999435	172.27.45.128	128.119.245.12	HTTP	225	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 ...	
739	7.000066	128.119.245.12	172.27.45.128	TCP	60	80 → 59063 [ACK] Seq=1 Ack=138265 Win=224128 Len...	
740	7.000066	128.119.245.12	172.27.45.128	TCP	60	80 → 59063 [ACK] Seq=1 Ack=139725 Win=227072 Len...	

```

> Frame 738: 225 bytes on wire (1800 bits), 225 bytes captured (1800 bits) on interface \Device\NPF_{...}
> Ethernet II, Src: Shenzhen_21:36:e2 (c8:4d:44:21:36:e2), Dst: HewlettP_18:db:6a (cc:3e:5f:18:db:6a)
> Internet Protocol Version 4, Src: 172.27.45.128, Dst: 128.119.245.12
> Transmission Control Protocol, Src Port: 59063, Dst Port: 80, Seq: 152865 Ack: 1, Len: 171
> [109 Reassembled TCP Segments (153035 bytes): #236(1024), #237(1460), #279(1460), #238(1460), #280(1460)]
> Hypertext Transfer Protocol
> MIME Multipart Media Encapsulation, Type: multipart/form-data, Boundary: "----WebKitFormBoundaryE5...
0000 cc 3e 5f 18 db 6a c8 4d
0010 00 d3 5c 9a 40 00 80 06
0020 f5 0c e6 b7 00 50 41 30
0030 10 04 4f e5 00 00 69 6e
0040 69 72 20 73 69 6d 70 6c
0050 0a 72 65 6d 65 6d 62 65
0060 20 6f 77 6e 20 63 68 69
0070 20 61 6e 64 20 74 68 65
0080 75 6d 6d 65 72 20 64 61

```

Corresponding arrival times are shown below:

230	3.715656	128.119.245.12	172.27.45.128	TCP	66	80 → 59064 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len...	
232	3.722559	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len...	
267	4.028375	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [ACK] Seq=1 Ack=1025 Win=31360 Len=0	
278	4.297893	128.119.245.12	172.27.45.128	TCP	66	[TCP Window Update] 80 → 59063 [ACK] Seq=1 Ack=...	
292	4.569356	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [ACK] Seq=1 Ack=5405 Win=43008 Len=0	
311	4.839782	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [ACK] Seq=1 Ack=6865 Win=45952 Len=0	
312	4.839782	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [ACK] Seq=1 Ack=8325 Win=48768 Len=0	
313	4.839782	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [ACK] Seq=1 Ack=9785 Win=51712 Len=0	
314	4.839782	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [ACK] Seq=1 Ack=11245 Win=54656 Len=...	
325	5.110058	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [ACK] Seq=1 Ack=12705 Win=57600 Len=...	
326	5.110058	128.119.245.12	172.27.45.128	TCP	66	80 → 59063 [ACK] Seq=1 Ack=15675 Win=60511 Len=0	

- b. At what time was each segment sent? And when was the ACK for each segment received?

Packet Number	Sent Time	Received Time	RTT
1	3.723408	4.028375	0.304967
2	3.724246	4.297893	0.573647
3	3.724246	4.569356	0.845110
4	3.724246	4.839782	1.115536
5	3.724246	4.839782	1.115536
6	3.724246	4.839782	1.115536

- c. What is the EstimatedRTT value after the receipt of each ACK? (\*Use the EstimatedRTT equation. Assume that the value of the EstimatedRTT is equal to the measured RTT for the first segment, and then is computed using the EstimatedRTT equation).

EstimatedRTT = 0.875 \* EstimatedRTT + 0.125 \* SampleRTT

Initially, Estimated RTT = 0.304967

Step 1 value  $0.875 \times 0.304967 + 0.125 \times 0.573647 = 0.338552$

Step 2 value  $0.875 \times 0.338552 + 0.125 \times 0.845110 = 0.401872$

Step 3 value  $0.875 \times 0.401872 + 0.125 \times 1.115536 = 0.491080$

Step 4 value  $0.875 \times 0.491080 + 0.125 \times 1.115536 = 0.569137$

Step 5 value  $0.875 \times 0.569137 + 0.125 \times 1.115536 = \mathbf{0.637437}$

d. What is the length of each of the first six TCP segments?

Lengths are **1078,1514,1514,1514,1514,1514**

e. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

The alice.txt on the hard drive is of size **152,136 bytes**

Download time is

$7.269500$  (last TCP segment) -  $3.723408$  (last ACK) =  $3.546092$  second.

Throughput for the TCP connection is computed as  $152,136 / 3.546092$

= **42902.4402** bytes/second.

## UDP Assignment

Start capturing packets in Wireshark and then do something that will cause your host to send and receive several UDP packets. It's also likely that just by doing nothing (except capturing packets via Wireshark) that some UDP packets sent by others will appear in your trace. After stopping packet capture, set your packet filter so that Wireshark only displays the UDP packets sent and received at your host. Pick one of these UDP packets and expand the UDP fields in the details window.

Use the packet trace that you have captured and answer the following questions, by opening the Wireshark captured packet file.

1. Select one UDP packet from your trace. From this packet, determine how many fields there are in the UDP header. Name these fields. (Answer these questions directly from what you observe in the packet trace.)

There are four fields – Source Port, Destination Port, Length and checksum

The screenshot shows a Wireshark packet capture window. A specific UDP packet is selected, highlighted with a blue background. The packet details pane shows the following information:

- Frame 398: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface \Device\NPF\_{D8...}
- Ethernet II, Src: HewlettP\_13:41:7c (48:0b:34:13:41:7c), Dst: IPv4mcast\_fb (01:00:5e:00:00:fb)
- Internet Protocol Version 4, Src: 172.27.45.126, Dst: 224.0.0.251
- User Datagram Protocol, Src Port: 5353, Dst Port: 5353
  - Source Port: 5353
  - Destination Port: 5353
  - Length: 48
  - Checksum: 0xd10e [unverified]
    - [Checksum Status: Unverified]
    - [Stream index: 35]
    - [Timestamps]
    - UDP payload (40 bytes)
  - Multicast Domain Name System (query)

Next to the details pane, the hex and ASCII panes are visible, showing the raw byte sequence and the corresponding ASCII characters.

2. By consulting the displayed information in Wireshark's packet content field for this packet, determine the length (in bytes) of each of the UDP header fields.

Each UDP header is of length **2 bytes** which can be understood from the hex values beside.

<pre>&gt; Frame 398: 82 bytes on wire (656 bits), 82 bytes &gt; Ethernet II, Src: HewlettP_13:41:7c (40:b0:34:13:... &gt; Internet Protocol Version 4, Src: 172.27.45.126, ... &lt; User Datagram Protocol, Src Port: 5353, Dst Port:   Source Port: 5353   Destination Port: 5353   Length: 48   Checksum: 0xd10e [unverified]   [Checksum Status: Unverified]   [Stream index: 35]   &gt; [Timestamps]   UDP payload (40 bytes)</pre>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>0000</td><td>01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00</td><td>..^...@ 4-A ..E.</td></tr> <tr><td>0010</td><td>00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00</td><td>.Dt..... ~...</td></tr> <tr><td>0020</td><td>00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01</td><td>....0 .....</td></tr> <tr><td>0030</td><td>00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61</td><td>..... _ googleca</td></tr> <tr><td>0040</td><td>73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c</td><td>st_tcp local...</td></tr> <tr><td>0050</td><td>00 01</td><td>..</td></tr> </table>	0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.	0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...	0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....	0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca	0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...	0050	00 01	..
0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.																	
0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...																	
0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....																	
0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca																	
0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...																	
0050	00 01	..																	
<pre>&gt; Frame 398: 82 bytes on wire (656 bits), 82 bytes &gt; Ethernet II, Src: HewlettP_13:41:7c (40:b0:34:13:... &gt; Internet Protocol Version 4, Src: 172.27.45.126, ... &lt; User Datagram Protocol, Src Port: 5353, Dst Port:   Source Port: 5353   Destination Port: 5353   Length: 48   Checksum: 0xd10e [unverified]   [Checksum Status: Unverified]   [Stream index: 35]   &gt; [Timestamps]   UDP payload (40 bytes)</pre>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>0000</td><td>01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00</td><td>..^...@ 4-A ..E.</td></tr> <tr><td>0010</td><td>00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00</td><td>.Dt..... ~...</td></tr> <tr><td>0020</td><td>00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01</td><td>....0 .....</td></tr> <tr><td>0030</td><td>00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61</td><td>..... _ googleca</td></tr> <tr><td>0040</td><td>73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c</td><td>st_tcp local...</td></tr> <tr><td>0050</td><td>00 01</td><td>..</td></tr> </table>	0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.	0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...	0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....	0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca	0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...	0050	00 01	..
0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.																	
0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...																	
0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....																	
0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca																	
0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...																	
0050	00 01	..																	
<pre>&gt; Frame 398: 82 bytes on wire (656 bits), 82 bytes &gt; Ethernet II, Src: HewlettP_13:41:7c (40:b0:34:13:... &gt; Internet Protocol Version 4, Src: 172.27.45.126, ... &lt; User Datagram Protocol, Src Port: 5353, Dst Port:   Source Port: 5353   Destination Port: 5353   Length: 48   Checksum: 0xd10e [unverified]   [Checksum Status: Unverified]   [Stream index: 35]   &gt; [Timestamps]   UDP payload (40 bytes)</pre>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>0000</td><td>01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00</td><td>..^...@ 4-A ..E.</td></tr> <tr><td>0010</td><td>00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00</td><td>.Dt..... ~...</td></tr> <tr><td>0020</td><td>00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01</td><td>....0 .....</td></tr> <tr><td>0030</td><td>00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61</td><td>..... _ googleca</td></tr> <tr><td>0040</td><td>73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c</td><td>st_tcp local...</td></tr> <tr><td>0050</td><td>00 01</td><td>..</td></tr> </table>	0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.	0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...	0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....	0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca	0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...	0050	00 01	..
0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.																	
0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...																	
0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....																	
0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca																	
0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...																	
0050	00 01	..																	
<pre>&gt; Frame 398: 82 bytes on wire (656 bits), 82 bytes &gt; Ethernet II, Src: HewlettP_13:41:7c (40:b0:34:13:... &gt; Internet Protocol Version 4, Src: 172.27.45.126, ... &lt; User Datagram Protocol, Src Port: 5353, Dst Port:   Source Port: 5353   Destination Port: 5353   Length: 48   Checksum: 0xd10e [unverified]   [Checksum Status: Unverified]   [Stream index: 35]   &gt; [Timestamps]   UDP payload (40 bytes)</pre>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>0000</td><td>01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00</td><td>..^...@ 4-A ..E.</td></tr> <tr><td>0010</td><td>00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00</td><td>.Dt..... ~...</td></tr> <tr><td>0020</td><td>00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01</td><td>....0 .....</td></tr> <tr><td>0030</td><td>00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61</td><td>..... _ googleca</td></tr> <tr><td>0040</td><td>73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c</td><td>st_tcp local...</td></tr> <tr><td>0050</td><td>00 01</td><td>..</td></tr> </table>	0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.	0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...	0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....	0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca	0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...	0050	00 01	..
0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.																	
0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...																	
0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....																	
0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca																	
0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...																	
0050	00 01	..																	

3.The value in the Length field is the length of what? (You can consult the text for this answer). Verify your claim with your captured UDP packet.

<pre>&gt; Frame 398: 82 bytes on wire (656 bits), 82 bytes &gt; Ethernet II, Src: HewlettP_13:41:7c (40:b0:34:13:... &gt; Internet Protocol Version 4, Src: 172.27.45.126, ... &lt; User Datagram Protocol, Src Port: 5353, Dst Port:   Source Port: 5353   Destination Port: 5353   Length: 48   Checksum: 0xd10e [unverified]   [Checksum Status: Unverified]   [Stream index: 35]   &gt; [Timestamps]   UDP payload (40 bytes)</pre>	<table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>0000</td><td>01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00</td><td>..^...@ 4-A ..E.</td></tr> <tr><td>0010</td><td>00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00</td><td>.Dt..... ~...</td></tr> <tr><td>0020</td><td>00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01</td><td>....0 .....</td></tr> <tr><td>0030</td><td>00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61</td><td>..... _ googleca</td></tr> <tr><td>0040</td><td>73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c</td><td>st_tcp local...</td></tr> <tr><td>0050</td><td>00 01</td><td>..</td></tr> </table>	0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.	0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...	0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....	0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca	0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...	0050	00 01	..
0000	01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00	..^...@ 4-A ..E.																	
0010	00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00	.Dt..... ~...																	
0020	00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01	....0 .....																	
0030	00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61	..... _ googleca																	
0040	73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c	st_tcp local...																	
0050	00 01	..																	

Length field has value as 48.

It is the length in bytes for the UDP segment (header + data).

Header will be of  $2 \times 4 = 8$  Bytes.

Hence payload will be of  $48 - 8 = 40$  Bytes which can be seen from above picture.

4. What is the maximum number of bytes that can be included in a UDP payload?

Source port field is 16 bits.

Therefore  $2^{16} - 1$  - header bytes(8) = **65527** can be included in the payload.

5. What is the largest possible source port number?

Largest possible source port number is  **$2^{16} - 1 = 65535$**

6. What is the protocol number for UDP?

Protocol number for UDP is **17** (0x11 in hexadecimal)

0100 .... = Version: 4 .... 0101 = Header Length: 20 bytes (5) > Differentiated Services Field: 0x00 (DSCP: CS0) Total Length: 68 Identification: 0x74fc (29948) > 000. .... = Flags: 0x0 ...0 0000 0000 0000 = Fragment Offset: 0 > Time to Live: 1 <b>Protocol: UDP (17)</b> Header Checksum: 0x8a18 [validation disabled] [Header checksum status: Unverified] Source Address: 172.27.45.126 Destination Address: 224.0.0.251 v User Datagram Protocol, Src Port: 5353, Dst Port:	0000 01 00 5e 00 00 fb 40 b0 34 13 41 7c 08 00 45 00 0010 00 44 74 fc 00 00 01 11 8a 18 ac 1b 2d 7e e0 00 0020 00 fb 14 e9 14 e9 00 30 d1 0e 00 00 00 00 00 01 0030 00 00 00 00 00 00 0b 5f 67 6f 6f 67 6c 65 63 61 0040 73 74 04 5f 74 63 70 05 6c 6f 63 61 6c 00 00 0c 0050 00 01	..^...@. 4-A ..E. .Dt ..... 0 ..... ....._ googleca st_tcp local... ...
---	--	---