



哈尔滨工业大学
Harbin Institute of Technology

计算机网络 课程实验报告

实验名称	利用 Wireshark 进行协议分析					
姓名	林燕燕		院系	人工智能		
班级	1903601		学号	1190200501		
任课教师	李全龙		指导教师	李全龙		
实验地点	格物 207		实验时间	2020.11.20		
实验课表现	出勤、表现得分 (10)		实验报告 得分(40)		实验总分	
	操作结果得分(50)					
教师评语						



实验目的:

熟悉并掌握 Wireshark 的基本操作, 了解网络协议实体间进行交互以及报文交换的情况。

实验内容:

- 1) 学习 Wireshark 的使用
- 2) 利用 Wireshark 分析 HTTP 协议
- 3) 利用 Wireshark 分析 TCP 协议
- 4) 利用 Wireshark 分析 IP 协议
- 5) 利用 Wireshark 分析 Ethernet 数据帧

选做内容:

- a) 利用 Wireshark 分析 DNS 协议
- b) 利用 Wireshark 分析 UDP 协议
- c) 利用 Wireshark 分析 ARP 协议

实验过程:

(一) Wireshark 的使用

选择无线网卡进行分组捕获, 并访问<http://www.hit.edu.cn>。在完整的页面加载完成后, 结束分组捕获。在这一段时间Wireshark捕获了本机所有利用该无线网卡与其他网络实体进行交换的报文。

(二) HTTP分析

1) HTTP GET/response交互

在Wireshark显示过滤部分输入“HTTP”即仅显示捕获的HTTP报文, 开始捕获后访问<http://news.hit.edu.cn>, 在加载完全部页面后停止分组捕获。

2) HTTP 条件 GET/response交互

将浏览器内的所有缓存清空, 启动Wireshark分组捕获, 访问<http://news.hit.edu.cn>, 在加载完全部页面后, 重新刷新页面; 在刷新页面后, 停止Wireshark分组捕获。

(三) TCP分析

1) 访问<http://gaia.cs.umass.edu/wireshark-labs/alice.txt>, 获得alice.txt文件。再打开

<https://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html>, 选择好本地alice.txt文件的位置, 开始Wireshark分组捕获后, 点击“Upload alice.txt file”按钮; 在文件上传完毕后, 停止Wireshark分组捕获。

2) 在筛选规则中选择“tcp”部分, 进行分析。

(四) IP分析

使用pingplotter进行实验, 启动Wireshark开始分组捕获, 首先发送一系列56字节的包; 再发送一系列2000字节的包; 再发送一系列3500字节的包, 然后停止Wireshark捕获。

(五) 抓取ARP数据包

1) 利用arp查看本机的ARP缓存表

2) 开始Wireshark分组捕获, 在命令行中输入: `ping 172.20.45.95`

3) ping通之后停止Wireshark捕获

(六) 抓取UDP数据包

启动Wireshark分组捕获, 利用QQ给好友发送消息, 消息发送结束后, 停止分组捕获。

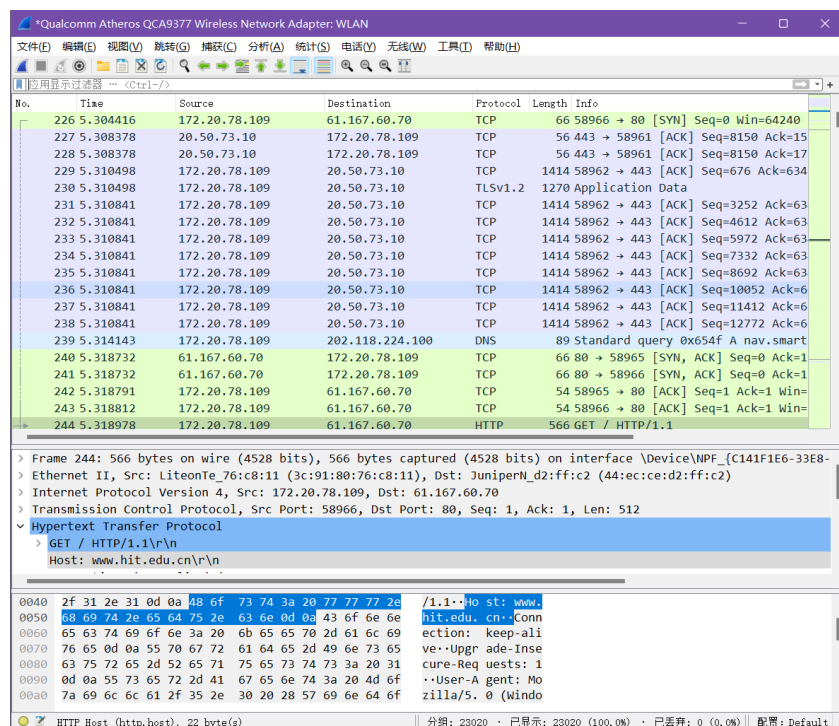
(七) 利用Wireshark进行DNS协议分析

首先清空dns缓存, 在浏览器中访问<https://www.baidu.com>, 进行Wireshark抓包。

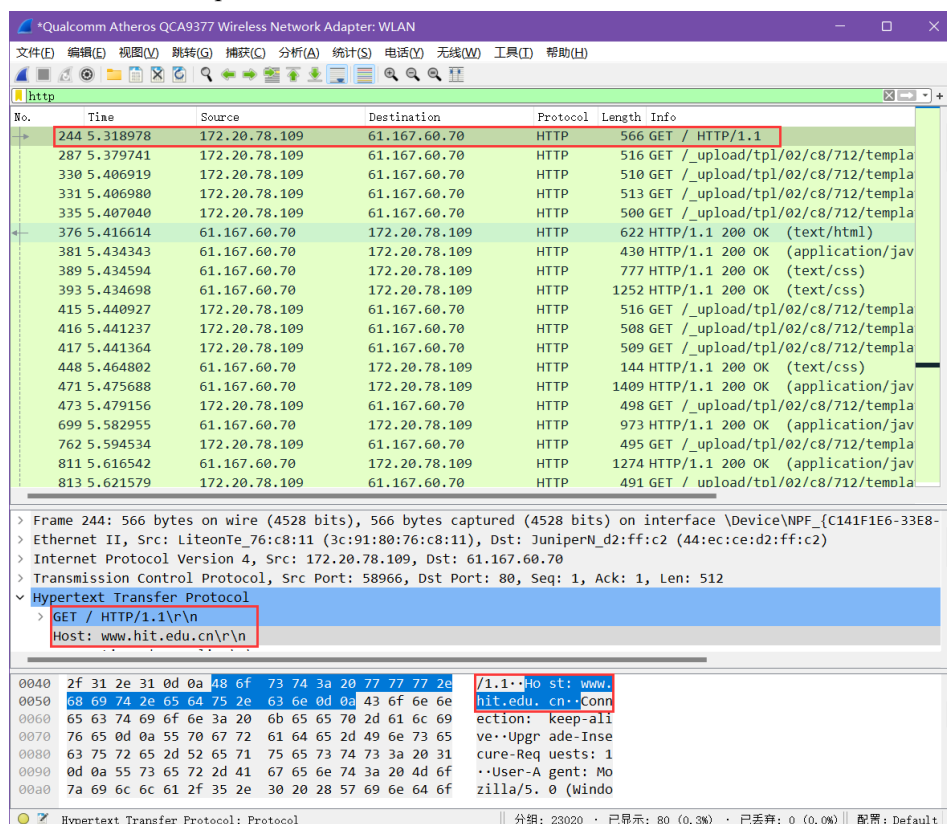
实验结果:

(一) Wireshark 的使用

访问<http://www.hit.edu.cn>。在完整的页面加载完成后，结束分组捕获。



在显示筛选规则中输入“http”:



筛选后可看到第一条HTTP报文为计算机发向 www.hit.edu.cn 服务器的 HTTP GET 报文。

(二) HTTP分析

1) HTTP GET/response交互

在Wireshark显示过滤部分输入“HTTP”即仅显示捕获的HTTP报文，开始捕获后访问<http://news.hit.edu.cn>，在加载完全部页面后停止分组捕获。

Wireshark packet capture showing HTTP GET request and response. The packet list shows a GET request from 172.20.78.109 to 202.118.254.136. The packet details show the request line 'GET / HTTP/1.1' and the response line '200 OK (text/html)'. The packet bytes show the raw data of the request and response.

No.	Time	Source	Destination	Protocol	Length	Info
44	1.087298	172.20.78.109	202.118.254.136	HTTP	446	GET / HTTP/1.1
97	1.183201	202.118.254.136	172.20.78.109	HTTP	1394	HTTP/1.1 200 OK (text/html)

Frame 44: 446 bytes on wire (3568 bits), 446 bytes captured (3568 bits) on interface \Device\NPF_{C141F1E6-33E8-4...}

Ethernet II, Src: LiteonTe_76:c8:11 (3c:91:80:76:c8:11), Dst: JuniperN_d2:ff:c2 (44:ec:ce:d2:ff:c2)

Internet Protocol Version 4, Src: 172.20.78.109, Dst: 202.118.254.136

Transmission Control Protocol, Src Port: 60581, Dst Port: 80, Seq: 1, Ack: 1, Len: 392

Hypertext Transfer Protocol

GET / HTTP/1.1\r\n

Host: news.hit.edu.cn\r\n

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:94.0) Gecko/20100101 Firefox/94.0\r\n

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8\r\n

Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2\r\n

Accept-Encoding: gzip, deflate\r\n

Connection: keep-alive\r\n

- a) 你的浏览器运行的是 HTTP1.0，还是 HTTP1.1？你所访问的服务器所运行 HTTP 协议的版本号是多少？

本地浏览器运行的协议是HTTP1.1，服务器运行的协议是HTTP1.1

446 GET / HTTP/1.1 1394 HTTP/1.1 200 OK (text/html)

- b) 你的浏览器向服务器指出它能接收何种语言版本的对象？

可接收的语言版本为zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2

Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2\r\n

- c) 你的计算机的 IP 地址是多少？服务器 <http://news.hit.edu.cn> 的 IP 地址是多少？

计算机IP地址为 172.20.78.109，服务器的IP地址为202.118.254.136

Source	Destination
172.20.78.109	202.118.254.136

- d) 从服务器向你的浏览器返回的状态代码是多少？

返回的状态码为200

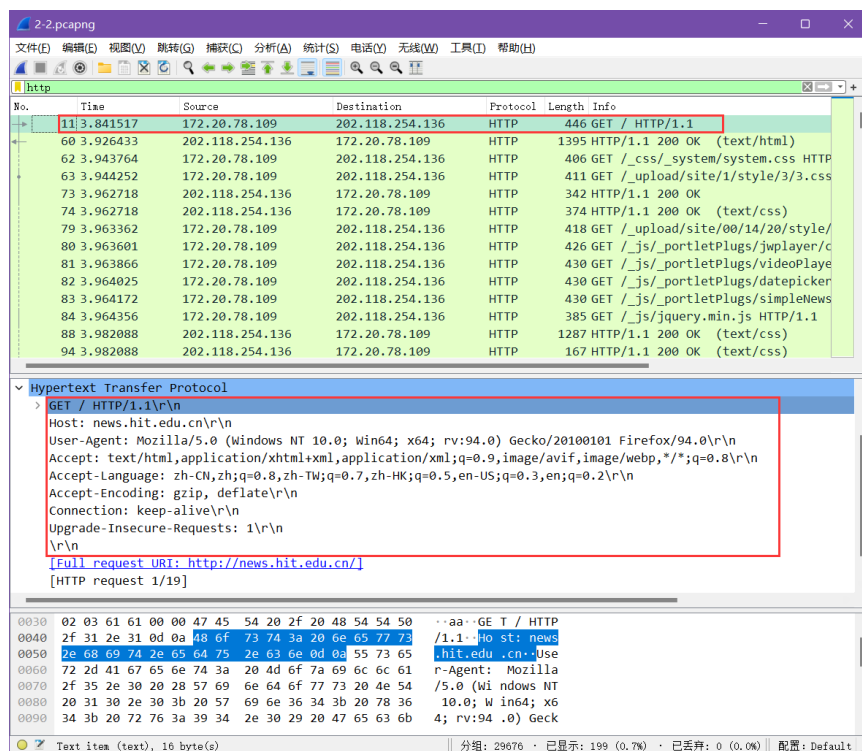
1394 HTTP/1.1 200 OK (text/html)

2) HTTP 条件 GET/response交互

将浏览器内的所有缓存清空，启动Wireshark分组捕获，访问<http://news.hit.edu.cn>，在加载完全部页面后，重新刷新页面；在刷新页面后，停止Wireshark分组捕获。

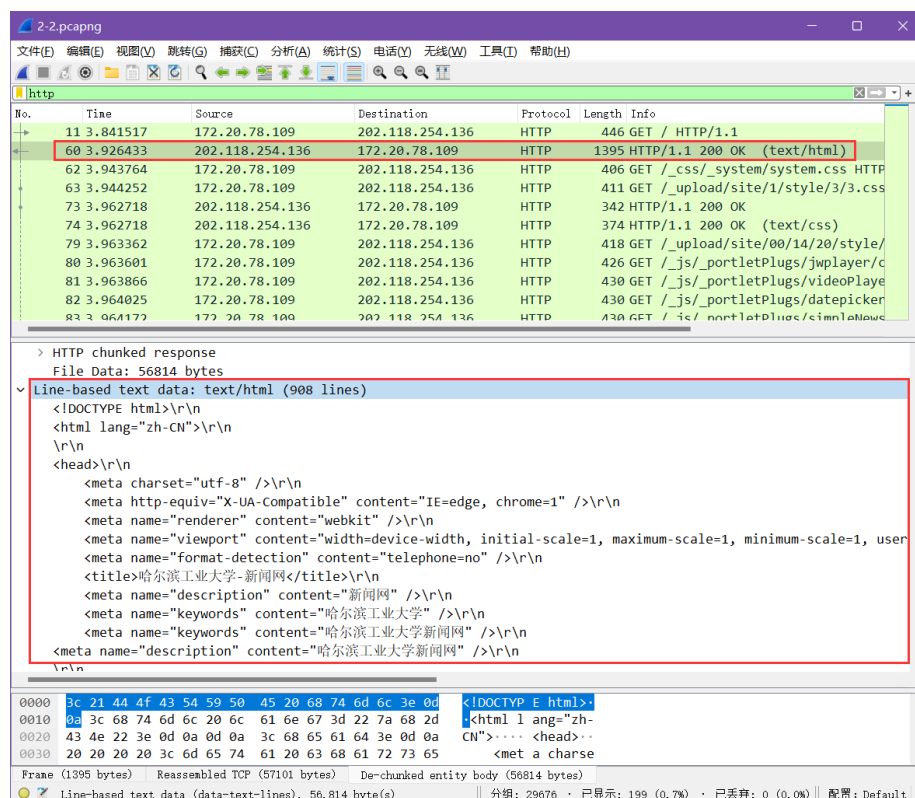
- a) 分析你的浏览器向服务器发出的第一个 HTTP GET 请求的内容，在该请求报文中，是否有一行是：IF-MODIFIED-SINCE？

没有IF-MODIFIED-SINCE头部

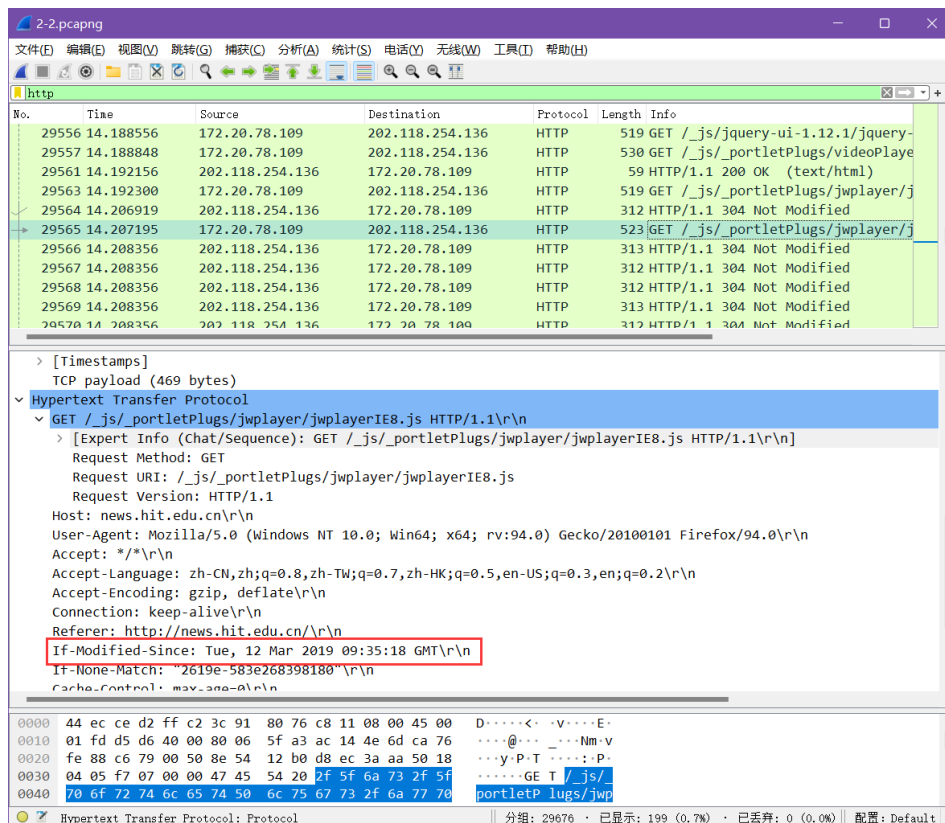


- b) 分析服务器响应报文的内容，服务器是否明确返回了文件的内容？如何获知？

服务器明确返回了文件的内容，在服务器响应报文中多了Line-based text data字段。

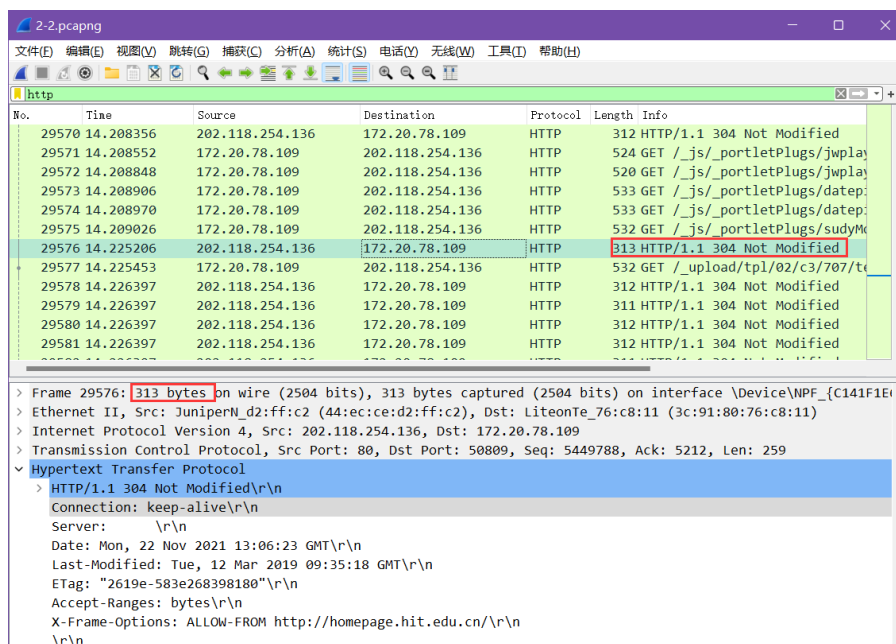


- c) 分析你的浏览器向服务器发出的较晚的“HTTP GET”请求，在该请求报文中是否有一行是：IF-MODIFIED-SINCE？如果有，在该首部行后面跟着的信息是什么？
- 有IF-MODIFIED-SINCE字段，后面跟着的信息是上一次缓存文件的修改时间。



- d) 服务器对较晚的 HTTP GET 请求的响应中的 HTTP 状态代码是多少？服务器是否明确返回了文件的内容？请解释。

状态码为304，服务器没有明确返回文件的内容，消息长度只有313bytes，缓存的内容没有更新，直接使用缓存中的储存的信息。



(三) TCP分析

- 1) 访问<http://gaia.cs.umass.edu/wireshark-labs/alice.txt>，获得alice.txt文件。再打开<https://gaia.cs.umass.edu/wireshark-labs/TCP-wireshark-file1.html>，选择好本地alice.txt文件的位置，开始Wireshark分组捕获后，点击”Upload alice.txt file”按钮；在文件上传完毕后，停止Wireshark分组捕获。
- 2) 在筛选规则中选择 “tcp” 部分，进行分析。
 - a) 向 gaia.cs.umass.edu 服务器传送文件的客户端主机的 IP 地址和TCP 端口号是多少？
客户端主机的 IP 地址是172.20.53.162，TCP 端口号是63190

Time	Source	Destination	Protocol	Length	Info
29 1.209174	2409:8c3c:900:10e:3...	2001:250:fe01:130:b...	TCP	78	443 → 52476 [SYN, ACK] Seq=0 Ack=1
30 1.626619	172.20.53.162	128.119.245.12	TCP	66	63190 → 80 [SYN] Seq=0 Win=64240 L
31 1.876810	172.20.53.162	128.119.245.12	TCP	66	63191 → 80 [SYN] Seq=0 Win=64240 L
32 1.958016	128.119.245.12	172.20.53.162	TCP	66	80 → 63190 [SYN, ACK] Seq=0 Ack=1
33 1.958090	172.20.53.162	128.119.245.12	TCP	54	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
34 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
35 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1361 Ack=1 Wi

Transmission Control Protocol, Src Port: 63190, Dst Port: 80, Seq: 0, Len: 0
Source Port: 63190
Destination Port: 80

- b) Gaia.cs.umass.edu 服务器的 IP 地址是多少？对这一连接，它用来发送和接收 TCP 报文的端口号是多少？
服务器的 IP 地址是128.119.245.12，用来发送和接收 TCP 报文的端口号都是80

Time	Source	Destination	Protocol	Length	Info
29 1.209174	2409:8c3c:900:10e:3...	2001:250:fe01:130:b...	TCP	78	443 → 52476 [SYN, ACK] Seq=0 Ack=1
30 1.626619	172.20.53.162	128.119.245.12	TCP	66	63190 → 80 [SYN] Seq=0 Win=64240 L
31 1.876810	172.20.53.162	128.119.245.12	TCP	66	63191 → 80 [SYN] Seq=0 Win=64240 L
32 1.958016	128.119.245.12	172.20.53.162	TCP	66	80 → 63190 [SYN, ACK] Seq=0 Ack=1
33 1.958090	172.20.53.162	128.119.245.12	TCP	54	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
34 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
35 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1361 Ack=1 Wi

Transmission Control Protocol, Src Port: 63190, Dst Port: 80, Seq: 0, Len: 0
Source Port: 63190
Destination Port: 80

- c) 客户服务器之间用于初始化 TCP 连接的 TCP SYN 报文段的序号（sequence number）是多少？在该报文段中，是用什么来标示该报文段是 SYN 报文段的？

用于初始化 TCP 连接的 TCP SYN 报文段的序号是0，在该报文段中将SYN标志位置为1来标示该报文段是 SYN 报文段的

Time	Source	Destination	Protocol	Length	Info
30 1.626619	172.20.53.162	128.119.245.12	TCP	66	63190 → 80 [SYN] Seq=0 Win=64240 L
31 1.876810	172.20.53.162	128.119.245.12	TCP	66	63191 → 80 [SYN] Seq=0 Win=64240 L
32 1.958016	128.119.245.12	172.20.53.162	TCP	66	80 → 63190 [SYN, ACK] Seq=0 Ack=1
33 1.958090	172.20.53.162	128.119.245.12	TCP	54	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
34 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
35 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1361 Ack=1 Wi
36 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=2721 Ack=1 Wi
37 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=4081 Ack=1 Wi

Transmission Control Protocol, Src Port: 63190, Dst Port: 80, Seq: 0, Len: 0
Source Port: 63190
Destination Port: 80
[Stream index: 6]
[TCP Segment Len: 0]
Sequence Number: 0 (relative sequence number)
Sequence Number (raw): 561754985
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 0
Acknowledgment number (raw): 0
1000 = Header Length: 32 bytes (8)
Flags: 0x002 (SYN)
000. = Reserved: Not set
...0 = Nonce: Not set
...0 = Congestion Window Reduced (CWR): 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

- d) 服务器向客户端发送的 SYNACK 报文段序号是多少？该报文段中，Acknowledgement 字段的值是多少？Gaia.cs.umass.edu 服务器是如何决定此值的？在该报文段中，是用什么来标示该报文段是SYNACK 报文段的？

服务器向客户端发送的 SYNACK 报文段序号是0，Acknowledgement 字段的值是1，服务器通过SYN请求报文段的seq序号加1确定acknowledgement字段，在该报文段中，是用 flags部分的ack和SYN标志位置为1表示该报文段为SYNACK报文段。

Time	Source	Destination	Protocol	Length	Info
30 1.626619	172.20.53.162	128.119.245.12	TCP	66	63190 → 80 [SYN] Seq=0 Win=64240 L
31 1.876810	172.20.53.162	128.119.245.12	TCP	66	63191 → 80 [SYN] Seq=0 Win=64240 L
32 1.958016	128.119.245.12	172.20.53.162	TCP	66	80 → 63190 [SYN, ACK] Seq=0 Ack=1
33 1.958090	172.20.53.162	128.119.245.12	TCP	54	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
34 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
35 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1361 Ack=1 Wi
36 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=2721 Ack=1 Wi
37 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=4081 Ack=1 Wi

Transmission Control Protocol, Src Port: 80, Dst Port: 63190, Seq: 0, Ack: 1, Len: 0 Source Port: 80 Destination Port: 63190 [Stream index: 6] [TCP Segment Len: 0] Sequence Number: 0 (relative sequence number) Sequence Number (raw): 2799867782 [Next Sequence Number: 1 (relative sequence number)] Acknowledgment Number: 1 (relative ack number) Acknowledgment number (raw): 561754986 1000 = Header Length: 32 bytes (8) Flags: 0x012 (SYN, ACK) 000. = Reserved: Not set ...0 = Nonce: Not set 0... = Congestion Window Reduced (CWR): 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
--

- e) 你能从捕获的数据包中分析出 tcp 三次握手过程吗？

以上两图反映了tcp连接的三次握手中的前两次，分别是客户机向服务器端发送SYN请求报文，以及服务器向客户机回复SYNACK报文；下图为客户机向服务器回复ack报文段：

Time	Source	Destination	Protocol	Length	Info
30 1.626619	172.20.53.162	128.119.245.12	TCP	66	63190 → 80 [SYN] Seq=0 Win=64240 L
31 1.876810	172.20.53.162	128.119.245.12	TCP	66	63191 → 80 [SYN] Seq=0 Win=64240 L
32 1.958016	128.119.245.12	172.20.53.162	TCP	66	80 → 63190 [SYN, ACK] Seq=0 Ack=1
33 1.958090	172.20.53.162	128.119.245.12	TCP	54	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
34 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1 Ack=1 Win=1
35 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1361 Ack=1 Wi
36 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=2721 Ack=1 Wi
37 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=4081 Ack=1 Wi

Transmission Control Protocol, Src Port: 63190, Dst Port: 80, Seq: 1, Ack: 1, Len: 0 Source Port: 63190 Destination Port: 80 [Stream index: 6] [TCP Segment Len: 0] Sequence Number: 1 (relative sequence number) Sequence Number (raw): 561754986 [Next Sequence Number: 1 (relative sequence number)] Acknowledgment Number: 1 (relative ack number) Acknowledgment number (raw): 2799867783 0101 = Header Length: 20 bytes (5) Flags: 0x010 (ACK) 000. = Reserved: Not set ...0 = Nonce: Not set 0... = Congestion Window Reduced (CWR): Not set 0... = ECN-Echo: Not set 0... = Urgent: Not set 1... = Acknowledgment: Set 0... = Push: Not set 0... = Reset: Not set 0... = Syn: Not set

可以根据回复的ack报文段中，ack端的内容为1（恰好为SYNACK报文段序号0加1所得），可知这是对SYN ACK报文段的回复，即第三次握手

f) 包含 HTTP POST 命令的 TCP 报文段的序号是多少？

包含 HTTP POST 命令的 TCP 报文段的序号是 1

Time	Source	Destination	Protocol	Length	Info
30	1.626619	172.20.53.162	128.119.245.12	TCP	66 63190 → 80 [SYN] Seq=0 Win=64240 L
31	1.876810	172.20.53.162	128.119.245.12	TCP	66 63191 → 80 [SYN] Seq=0 Win=64240 L
32	1.958016	128.119.245.12	172.20.53.162	TCP	66 80 → 63190 [SYN, ACK] Seq=0 Ack=1
33	1.958090	172.20.53.162	128.119.245.12	TCP	54 63190 → 80 [ACK] Seq=1 Ack=1 Win=1
34	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=1 Ack=1 Win=1
35	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=1361 Ack=1 Wi
36	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=2721 Ack=1 Wi
37	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=4081 Ack=1 Wi

Destination Port: 80
[Stream index: 6]
[TCP Segment Len: 1360]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 561754986
[Next Sequence Number: 1361 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 2799867783
0101 = Header Length: 20 bytes (5)
Flags: 0x010 (ACK)
Window: 515

0030	02 03 21 36 00 00 50 4f	53 54 20 2f 77 69 72 65	..16..PO St /wire
0040	73 68 61 72 6b 2d 6c 61	62 73 2f 6c 61 62 33 2d	shark-lab/bs/lab3-
0050	31 2d 72 65 70 6c 79 2e	68 74 6d 20 48 54 54 50	i-reply.htm HTTP
0060	2f 31 2e 31 0d 0a 48 6f	73 74 3a 20 67 61 69 61	/1.1-Host: gaia
0070	2e 63 73 2e 75 6d 61 73	73 2e 65 64 75 0d 0a 55	ccs.umass.edu:80
0080	73 65 72 2d 41 67 65 6e	74 3a 20 4d 6f 7a 69 6c	ser-Agent: Mozilla

g) 如果将包含 HTTP POST 命令的 TCP 报文段看作是 TCP 连接上的第一个报文段，那么该 TCP 连接上的第六个报文段的序号是多少？是何时发送的？该报文段所对应的 ACK 是何时接收的？

该 TCP 连接上的第六个报文段的序号是 6801；是第一帧发送后 0.331851s 发送的；该报文段所对应的 ACK 是第一帧发送后 0.664942s 接收的

Time	Source	Destination	Protocol	Length	Info
30	1.626619	172.20.53.162	128.119.245.12	TCP	66 63190 → 80 [SYN] Seq=0 Win=64240 Le
31	1.876810	172.20.53.162	128.119.245.12	TCP	66 63191 → 80 [SYN] Seq=0 Win=64240 Le
32	1.958016	128.119.245.12	172.20.53.162	TCP	66 80 → 63190 [SYN, ACK] Seq=0 Ack=1 w
33	1.958090	172.20.53.162	128.119.245.12	TCP	54 63190 → 80 [ACK] Seq=1 Ack=1 Win=13
34	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=1 Ack=1 Win=13
35	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=1361 Ack=1 Win
36	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=2721 Ack=1 Win
37	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=4081 Ack=1 Win
38	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=5441 Ack=1 Win
39	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=6801 Ack=1 Win
40	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=8161 Ack=1 Win
41	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=9521 Ack=1 Win
42	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=10881 Ack=1 Wi
43	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=12241 Ack=1 Wi
44	2.179990	128.119.245.12	172.20.53.162	TCP	66 80 → 63191 [SYN, ACK] Seq=0 Ack=1 w
45	2.180071	172.20.53.162	128.119.245.12	TCP	54 63191 → 80 [ACK] Seq=1 Ack=1 Win=13

[window size scaling factor: 256]
Checksum: 0x2e7f [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [SEQ/ACK analysis]
~ [Timestamps]
[Time since first frame in this TCP stream: 0.331851000 seconds]
[Time since previous frame in this TCP stream: 0.000000000 seconds]
TCP payload (1360 bytes)

Time	Source	Destination	Protocol	Length	Info
36	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=2721 Ack=1 Win
37	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=4081 Ack=1 Win
38	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=5441 Ack=1 Win
39	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=6801 Ack=1 Win
40	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=8161 Ack=1 Win
41	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=9521 Ack=1 Win
42	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=10881 Ack=1 Wi
43	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=12241 Ack=1 Wi
44	2.179990	128.119.245.12	172.20.53.162	TCP	66 80 → 63191 [SYN, ACK] Seq=0 Ack=1 w
45	2.180071	172.20.53.162	128.119.245.12	TCP	54 63191 → 80 [ACK] Seq=1 Ack=1 Win=13
46	2.228159	172.20.53.162	112.80.248.251	TCP	54 63133 → 80 [RST, ACK] Seq=1 Ack=1 w
47	2.232633	2402:4e00:1020:100b...	2001:250:fe01:130:d...	TCP	78 443 → 42176 [SYN, ACK] Seq=0 Ack=1
48	2.291561	128.119.245.12	172.20.53.162	TCP	56 80 → 63190 [ACK] Seq=1 Ack=4081 Win
49	2.291561	128.119.245.12	172.20.53.162	TCP	56 80 → 63190 [ACK] Seq=1 Ack=9521 Win
50	2.291561	128.119.245.12	172.20.53.162	TCP	56 80 → 63190 [ACK] Seq=1 Ack=12241 Wi
51	2.291561	128.119.245.12	172.20.53.162	TCP	56 80 → 63190 [ACK] Seq=1 Ack=13601 Wi

[Calculated window size: 48256]
[Window size scaling factor: 128]
Checksum: 0x277a [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [SEQ/ACK analysis]
~ [Timestamps]
[Time since first frame in this TCP stream: 0.664942000 seconds]
[Time since previous frame in this TCP stream: 0.000000000 seconds]

h) 前六个 TCP 报文段的长度各是多少？

前六个 TCP 报文段的长度都是1360

34	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=1 Ack=1 Win=131840 Len=1360 [TCP segment of a reassembled PDU]
35	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=1361 Ack=1 Win=131840 Len=1360 [TCP segment of a reassembled PDU]
36	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=2721 Ack=1 Win=131840 Len=1360 [TCP segment of a reassembled PDU]
37	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=4081 Ack=1 Win=131840 Len=1360 [TCP segment of a reassembled PDU]
38	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=5441 Ack=1 Win=131840 Len=1360 [TCP segment of a reassembled PDU]
39	1.958470	172.20.53.162	128.119.245.12	TCP	1414 63190 → 80 [ACK] Seq=6801 Ack=1 Win=131840 Len=1360 [TCP segment of a reassembled PDU]

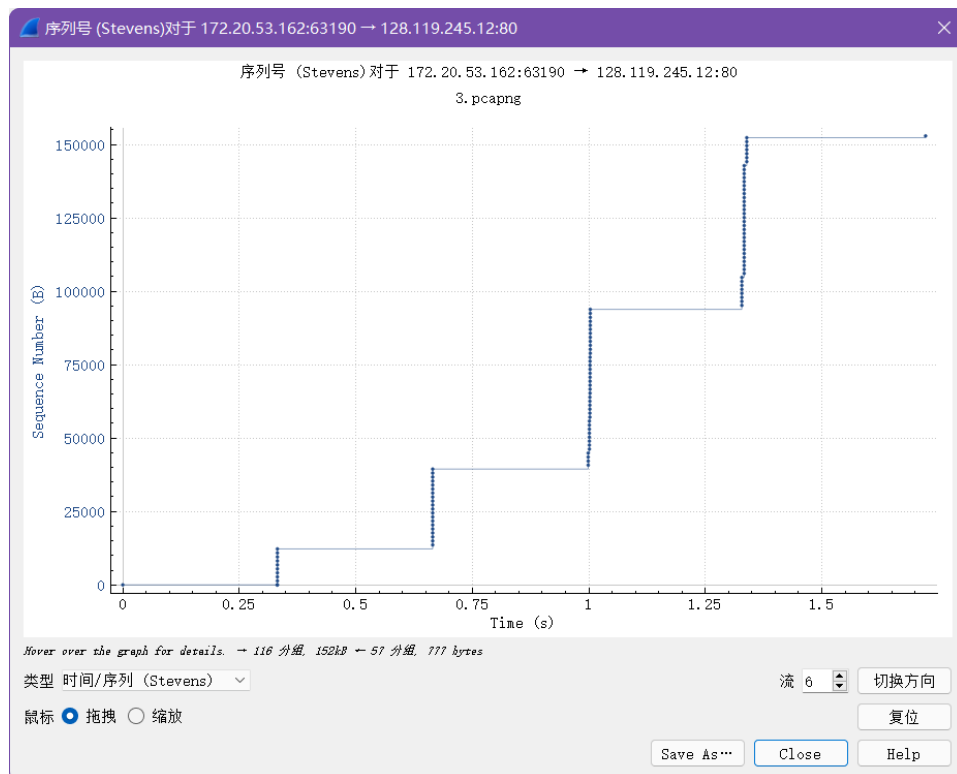
i) 在整个跟踪过程中，接收端公示的最小的可用缓存空间是多少？限制发送端的传输以后，接收端的缓存是否仍然不够用？

在整个的跟踪过程中，接收端公示最小可用的缓存空间为131840，并且始终为131840。所以始终没有出现限制发送端发送的情况。

1414	63190	→	80	[ACK]	Seq=1361 Ack=1 Win=131840 Len=1360
1414	63190	→	80	[ACK]	Seq=2721 Ack=1 Win=131840 Len=1360
1414	63190	→	80	[ACK]	Seq=4081 Ack=1 Win=131840 Len=1360
1414	63190	→	80	[ACK]	Seq=5441 Ack=1 Win=131840 Len=1360
1414	63190	→	80	[ACK]	Seq=6801 Ack=1 Win=131840 Len=1360
1414	63190	→	80	[ACK]	Seq=8161 Ack=1 Win=131840 Len=1360
1414	63190	→	80	[ACK]	Seq=9521 Ack=1 Win=131840 Len=1360
1414	63190	→	80	[ACK]	Seq=10881 Ack=1 Win=131840 Len=1360
1414	63190	→	80	[ACK]	Seq=12241 Ack=1 Win=131840 Len=1360

j) 在跟踪文件中是否有重传的报文段？进行判断的依据是什么？

没有重传的片段，依据为发送端的报文段序号始终在增加，没有出现重复发送某一个序号的报文段的情况，故没有重传的。报文段序号变化如下：



k) TCP 连接的 throughput (bytes transferred per unit time)是多少？请写出你的计算过程
第一个TCP段及最后一个TCP段如下：

Time	Source	Destination	Protocol	Length	Info
30 1.626619	172.20.53.162	128.119.245.12	TCP	66	63190 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
31 1.876810	172.20.53.162	128.119.245.12	TCP	66	63191 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460
32 1.958016	128.119.245.12	172.20.53.162	TCP	66	80 → 63190 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0
33 1.958090	172.20.53.162	128.119.245.12	TCP	54	63190 → 80 [ACK] Seq=1 Ack=1 Win=131840 Len=0
34 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1 Ack=1 Win=131840 Len=1360
35 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=1361 Ack=1 Win=131840 Len=1360
36 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=2721 Ack=1 Win=131840 Len=1360
37 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=4081 Ack=1 Win=131840 Len=1360
38 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=5441 Ack=1 Win=131840 Len=1360
39 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=6801 Ack=1 Win=131840 Len=1360
40 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=8161 Ack=1 Win=131840 Len=1360
41 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=9521 Ack=1 Win=131840 Len=1360
42 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=10881 Ack=1 Win=131840 Len=1360
43 1.958470	172.20.53.162	128.119.245.12	TCP	1414	63190 → 80 [ACK] Seq=12241 Ack=1 Win=131840 Len=1360
44 2.179990	128.119.245.12	172.20.53.162	TCP	66	80 → 63191 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0

[Calculated window size: 131840]
 [Window size scaling factor: 256]
 Checksum: 0x2136 [unverified]
 [Checksum Status: Unverified]
 Urgent Pointer: 0
 > [SEQ/ACK analysis]
 > [Timestamps]
 [Time since first frame in this TCP stream: 0.331851000 seconds]
 [Time since previous frame in this TCP stream: 0.000380000 seconds]
 TCP payload (1360 bytes)
 [Reassembled PDU in frame: 186]
 TCP segment data (1360 bytes)

Time	Source	Destination	Protocol	Length	Info
199 3.284685	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=100641 Win=186112 Len=0
200 3.284685	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=102001 Win=189056 Len=0
201 3.286669	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=106081 Win=197248 Len=0
202 3.290184	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=107441 Win=200192 Len=0
203 3.290184	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=112881 Win=208256 Len=0
204 3.290184	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=119681 Win=207232 Len=0
205 3.292843	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=126481 Win=207232 Len=0
206 3.292843	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=127841 Win=211072 Len=0
207 3.292843	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=130561 Win=210176 Len=0
208 3.295245	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=131921 Win=211072 Len=0
209 3.295245	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=133281 Win=210176 Len=0
210 3.295245	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=136001 Win=208256 Len=0
211 3.295245	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=137361 Win=211072 Len=0
212 3.295245	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=144161 Win=211072 Len=0
213 3.297767	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=148241 Win=211072 Len=0
214 3.301274	128.119.245.12	172.20.53.162	TCP	56	80 → 63190 [ACK] Seq=1 Ack=152927 Win=211072 Len=0
215 3.303420	128.119.245.12	172.20.53.162	HTTP	831	HTTP/1.1 200 OK (text/html)
216 3.341754	34.117.237.239	172.20.53.162	TCP	56	443 → 63070 [ACK] Seq=1 Ack=65 Win=272 Len=0

[Calculated window size: 211072]
 [Window size scaling factor: 128]
 Checksum: 0xf251 [unverified]
 [Checksum Status: Unverified]
 Urgent Pointer: 0
 > [SEQ/ACK analysis]
 > [Timestamps]
 [Time since first frame in this TCP stream: 1.674655000 seconds]
 [Time since previous frame in this TCP stream: 0.003507000 seconds]

吞吐量计算如下：

$$\text{Throughput} = \frac{(152927 - 1) * 8 \text{ bits}}{(1.674655 - 0.331851)s} = 0.91 \text{ Mbps}$$

(四) IP分析

使用pingplotter进行实验，启动Wireshark开始分组捕获，首先发送一系列56字节的包；再发送一系列2000字节的包；再发送一系列3500字节的包，然后停止Wireshark捕获。

a) 你主机的IP地址是什么？

本机地址为172.20.2.22

icmp					
No.	Time	Source	Destination	Protocol	Length Info
2	0.168464	10.160.254.106	172.20.2.22	ICMP	70 Time-to-live exceeded (Time to live exceeded)
5	0.428082	172.20.2.22	61.167.60.70	ICMP	70 Echo (ping) request id=0x0001, seq=183
6	0.435060	61.167.60.70	172.20.2.22	ICMP	70 Echo (ping) reply id=0x0001, seq=183
7	0.478569	172.20.2.22	61.167.60.70	ICMP	70 Echo (ping) request id=0x0001, seq=183
8	0.485181	10.0.3.0	172.20.2.22	ICMP	70 Time-to-live exceeded (Time to live exceeded)
9	0.529184	172.20.2.22	61.167.60.70	ICMP	70 Echo (ping) request id=0x0001, seq=183
10	0.535371	192.168.82.1	172.20.2.22	ICMP	70 Time-to-live exceeded (Time to live exceeded)

- 上层协议字段的值是ICMP(1)

```
> Frame 5: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface \Device\NPF_{C141F1E6-33E8-4000-8000-000000000000}
> Ethernet II, Src: IntelNetE_76:c8:11 (3c:91:80:76:c8:11), Dst: JuniperNt_d2:ff:c2 (44:ec:ce:d2:ff:c2)
> Internet Protocol Version 4, Src: 172.20.2.22, Dst: 61.167.60.70
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 56
  Identification: 0xd758 (55128)
> Flags: 0x00
  Fragment Offset: 0
  Time to Live: 255
  Protocol: ICMP (1)
  Header Checksum: 0xbc54 [validation disabled]
```

- 大小的？

IP头有20字节，IP数据包大小为56，则净载为36字节

```

Internet Protocol Version 4, Src: 172.20.2.22, Dst: 61.167.60.70
  0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56

```

- 该IP数据包未分片，观察flag段，没有其余帧且帧偏移为0

```

  ▾ Flags: 0x00
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..0. .... = More fragments: Not set
    Fragment Offset: 0

```

- IP数据报中TTL、checksum和sequence number总是发生改变。

Destination	Protocol	Length	Info
172.20.2.22	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1845/13575, ttl=3 (no res...
172.20.2.22	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1846/13831, ttl=4 (no res...
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1847/14087, ttl=5 (reply...
172.20.2.22	ICMP	70	Echo (ping) reply id=0x0001, seq=1847/14087, ttl=124 (requ...
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1848/14343, ttl=255 (repl...
172.20.2.22	ICMP	70	Echo (ping) reply id=0x0001, seq=1848/14343, ttl=124 (requ...
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1849/14599, ttl=1 (no res...
172.20.2.22	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1850/14855, ttl=2 (no res...
172.20.2.22	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1851/15111, ttl=3 (no res...
172.20.2.22	ICMP	70	Time-to-live exceeded (Time to live exceeded in transit)
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1852/15367, ttl=4 (no res...
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1853/15623, ttl=5 (reply...
172.20.2.22	ICMP	70	Echo (ping) reply id=0x0001, seq=1853/15623, ttl=124 (requ...
61.167.60.70	ICMP	70	Echo (ping) request id=0x0001, seq=1854/15879, ttl=6 (reply...

Total Length: 56
 Identification: 0xd3ed8 (16088)
 > Flags: 0x00
 Fragment Offset: 0
 Time to Live: 124
 Protocol: ICMP (1)
 Header Checksum: 0xd7d5 [validation disabled]
 [Header checksum status: Unverified]
 Source Address: 61.167.60.70
 Destination Address: 172.20.2.22

- f) 哪些字段必须保持常量？哪些字段必须改变？为什么？

必须保持常量的是版本号、首部长度的、区分服务（Differentiated Services Field）以及协议（始终为ICMP）；

必须改变的是TTL、checksum和sequence number，TTL为生存时间，每次转发必然改变，由于TTL的改变，checksum也会改变，sequence number是变化的区分不同的ICMP报文。

- g) 描述你看到的IP数据包Identification字段值的形式。

Identification字段值，一字节一字节逐渐递增：

28	1.424737	172.20.2.22	61.167.60.70	ICMP	28	1.424737	172.20.2.22	61.167.60.70	ICMP
29	1.474258	172.20.2.22	61.167.60.70	ICMP	29	1.474258	172.20.2.22	61.167.60.70	ICMP
30	1.479301	61.167.60.70	172.20.2.22	ICMP	30	1.479301	61.167.60.70	172.20.2.22	ICMP

Total Length: 56	Total Length: 56
Identification: 0xd762 (55138)	Identification: 0xd763 (55139)
Flags: 0x00	Flags: 0x00
Fragment Offset: 0	Fragment Offset: 0

- h) Identification字段和TTL字段的值是什么？

Identification字段的值是0xcd00，TTL字段的值是252

13	0.580271	172.20.2.22	61.167.60.70	ICMP
14	0.593287	10.160.254.106	172.20.2.22	ICMP
16	0.630710	172.20.2.22	61.167.60.70	ICMP
17	0.681149	172.20.2.22	61.167.60.70	ICMP

Total Length: 56
Identification: 0xcd00 (52480)
Flags: 0x00
Fragment Offset: 0
Time to Live: 252
Protocol: ICMP (1)

- i) 最近的路由器（第一跳）返回给你主机的ICMP Time-to-live exceeded消息中这些值是否保持不变？为什么？

Identification段变化，为了区分不同的ICMP time-to-live exceeded消息，但TTL保持不变，均为一次转发。

- j) 当包的大小变为2000字节后，找到第一个ICMP echo request消息，该消息是否被分解成不止一个IP数据报？

当包的大小变为2000字节后，第一个ICMP echo request消息形式如下，可以看到确实被分解成了不止一个数据包。

1116	38.787509	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=d86e)
1117	38.787509	172.20.2.22	61.167.60.70	ICMP	534 Echo (ping) request id=0x0001, seq=2114/16904, ttl=1

- k) 观察第一个IP分片，IP头部的哪些信息表明数据包被进行了分片？IP头部的哪些信息表明数据包是第一个而不是最后一个分片？该分片的长度是多少

flag域中，More fragments位被置为1且长度非1500，表示发生分片；，More fragments位被置为1且offset为0，表示发生分片且该分片不为最后一块；且长度为1500字节

> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 1500
Identification: 0xd86e (55406)
> Flags: 0x20, More fragments
0... = Reserved bit: Not set
.0.. = Don't fragment: Not set
..1. = More fragments: Set
Fragment Offset: 0
> Time to Live: 1
Protocol: ICMP (1)

- l) 当包的大小变为3500字节后，找到第一个ICMP echo request消息，原始数据包被分成了多

少片？

原始数据包被分成3片

2845	87.202167	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol
2846	87.202167	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol
2847	87.202167	172.20.2.22	61.167.60.70	ICMP	554 Echo (ping) request id

m) 这些分片中IP数据报头部哪些字段发生了变化？

IP数据报头部flag字段和checksum字段发生了变化。

2845	87.202167	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol
2846	87.202167	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol
2847	87.202167	172.20.2.22	61.167.60.70	ICMP	554 Echo (ping) request id
2848	87.219021	10.0.3.0	172.20.2.22	ICMP	70 Time-to-live exceeded (
2849	87.252443	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol

Identification: 0xd9cf (55759)

Flags: 0x20, More fragments
 0... = Reserved bit: Not set
 .0.. = Don't fragment: Not set
 ..1. = More fragments: Set
 Fragment Offset: 0

Time to Live: 1

Protocol: ICMP (1)

Header Checksum: 0x923a [validation disabled]

2845	87.202167	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol
2846	87.202167	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol
2847	87.202167	172.20.2.22	61.167.60.70	ICMP	554 Echo (ping) request id
2848	87.219021	10.0.3.0	172.20.2.22	ICMP	70 Time-to-live exceeded (
2849	87.252443	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol

Identification: 0xd9cf (55759)

Flags: 0x20, More fragments
 0... = Reserved bit: Not set
 .0.. = Don't fragment: Not set
 ..1. = More fragments: Set
 Fragment Offset: 1480

Time to Live: 1

Protocol: ICMP (1)

Header Checksum: 0x9181 [validation disabled]

2845	87.202167	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol
2846	87.202167	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol
2847	87.202167	172.20.2.22	61.167.60.70	ICMP	554 Echo (ping) request id
2848	87.219021	10.0.3.0	172.20.2.22	ICMP	70 Time-to-live exceeded (
2849	87.252443	172.20.2.22	61.167.60.70	IPv4	1514 Fragmented IP protocol

Identification: 0xd9cf (55759)

Flags: 0x01
 0... = Reserved bit: Not set
 .0.. = Don't fragment: Not set
 ..0. = More fragments: Not set
 Fragment Offset: 2960

Time to Live: 1

Protocol: ICMP (1)

Header Checksum: 0xb488 [validation disabled]

(五) 抓取ARP数据包

1) 利用arp查看本机的ARP缓存表

2) 开始Wireshark分组捕获，在命令行中输入：ping 172.20.45.95

3) ping通之后停止Wireshark捕获

a) 利用 MS-DOS 命令：arp 或 c:\windows\system32\arp 查看主机上 ARP 缓存的内容。说明 ARP 缓存中每一列的含义是什么？

ARP 缓存中每一列均由其表头决定，如地址等；每一行为某地址对应的硬件地址：


```

PS C:\Users\lyy> arp -a

接口: 192.168.6.1 --- 0xb
Internet 地址      物理地址      类型
192.168.6.254      00-50-56-fd-e7-fc 动态
192.168.6.255      ff-ff-ff-ff-ff-ff 静态
224.0.0.22         01-00-5e-00-00-16 静态
224.0.0.251        01-00-5e-00-00-fb 静态
224.0.0.252        01-00-5e-00-00-fc 静态
239.255.255.250    01-00-5e-7f-ff-fa 静态
255.255.255.255    ff-ff-ff-ff-ff-ff 静态

接口: 192.168.242.1 --- 0x11
Internet 地址      物理地址      类型
192.168.242.254    00-50-56-fa-da-6f 动态
192.168.242.255    ff-ff-ff-ff-ff-ff 静态
224.0.0.22         01-00-5e-00-00-16 静态
224.0.0.251        01-00-5e-00-00-fb 静态
224.0.0.252        01-00-5e-00-00-fc 静态
239.255.255.250    01-00-5e-7f-ff-fa 静态
255.255.255.255    ff-ff-ff-ff-ff-ff 静态

接口: 172.20.160.92 --- 0x13
Internet 地址      物理地址      类型
172.20.0.1         44-ec-ce-d2-ff-c2 动态
172.20.45.95       44-ec-ce-d2-ff-c2 动态
172.20.78.109      44-ec-ce-d2-ff-c2 动态
172.20.127.38      44-ec-ce-d2-ff-c2 动态
172.20.255.255     ff-ff-ff-ff-ff-ff 静态
224.0.0.22         01-00-5e-00-00-16 静态
224.0.0.251        01-00-5e-00-00-fb 静态
224.0.0.252        01-00-5e-00-00-fc 静态
255.255.255.255    ff-ff-ff-ff-ff-ff 静态
    
```

- b) 清除主机上 ARP 缓存的内容,抓取 ping 命令时的数据包。分析数据包,回答下面的问题:
 ARP数据包的格式是怎样的? 由几部分构成, 各个部分所占的字节数是多少?
 ARP数据包格式如下:



No.	Time	Source	Destination	Protocol	Length	Info
39	10.694666	LiteonTe_76:c8:11	JuniperN_d2:ff:c2	ARP	42	Who has 172.20.45.95?
40	10.698899	JuniperN_d2:ff:c2	LiteonTe_76:c8:11	ARP	56	172.20.45.95 is at 44:ec:ce:d2:ff:c2

Address Resolution Protocol (request)

```

Hardware type: Ethernet (1)
Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (1)
Sender MAC address: LiteonTe_76:c8:11 (3c:91:80:76:c8:11)
Sender IP address: 172.20.160.92
Target MAC address: JuniperN_d2:ff:c2 (44:ec:ce:d2:ff:c2)
Target IP address: 172.20.45.95
    
```

- c) 如何判断一个ARP数据是请求包还是应答包?
 只需要判断opcode部分, 当其值为0x0001时为请求, 0x0002时为应答

39	10.694666	LiteonTe_76:c8:11	JuniperN_d2:ff:c2	ARP	42	Who has 172.20.45.95?
40	10.698899	JuniperN_d2:ff:c2	LiteonTe_76:c8:11	ARP	56	172.20.45.95 is at 44:ec:ce:d2:ff:c2

Address Resolution Protocol (request)

```

Hardware type: Ethernet (1)
Protocol type: IPv4 (0x0800)
Hardware size: 6
Protocol size: 4
Opcode: request (1)
Sender MAC address: LiteonTe_76:c8:11 (3c:91:80:76:c8:11)
Sender IP address: 172.20.160.92
Target MAC address: JuniperN_d2:ff:c2 (44:ec:ce:d2:ff:c2)
Target IP address: 172.20.45.95
    
```

39	10.694666	LiteonTe_76:c8:11	JuniperN_d2:ff:c2	ARP	42	Who has 172.20.45.95? Tr
40	10.698899	JuniperN_d2:ff:c2	LiteonTe_76:c8:11	ARP	56	172.20.45.95 is at 44:ec

- Address Resolution Protocol (reply)
 - Hardware type: Ethernet (1)
 - Protocol type: IPv4 (0x0800)
 - Hardware size: 6
 - Protocol size: 4
 - Opcode: reply (2)
 - Sender MAC address: JuniperN_d2:ff:c2 (44:ec:ce:d2:ff:c2)
 - Sender IP address: 172.20.45.95
 - Target MAC address: LiteonTe_76:c8:11 (3c:91:80:76:c8:11)
 - Target IP address: 172.20.160.92

- d) 为什么ARP查询要在广播帧中传送，而ARP响应要在一个有着明确目的局域网地址的帧中传送？

因为不知道具体的ip地址对应的mac地址，需要通过ARP查询得到ip地址对应的mac地址；而由于响应报文能从ARP查询的ip数据包中，拆分出发送查询的主机的ip地址和mac地址，所以不需要再利用广播帧发送信息

(六) 抓取UDP数据包

启动Wireshark分组捕获，利用QQ给好友发送消息，消息发送结束后，停止分组捕获。

- a) 消息是基于UDP的还是TCP的？

消息是基于UDP的

No.	Time	Source	Destination	Protocol	Length	Info
3	0.919590	111.30.159.62	172.20.160.92	OICQ	129	OICQ Protocol
4	0.921754	172.20.160.92	111.30.159.62	UDP	89	4008 → 8000 Len=47
5	0.984891	111.30.159.62	172.20.160.92	UDP	81	8000 → 4008 Len=39
8	1.757127	172.20.160.92	202.118.224.100	DNS	86	Standard query 0x090a A driver
9	1.762907	202.118.224.100	172.20.160.92	DNS	528	Standard query response 0x090a
28	2.199241	111.30.159.62	172.20.160.92	OICQ	129	OICQ Protocol
30	2.698234	111.30.159.62	172.20.160.92	OICQ	433	OICQ Protocol
31	2.699058	172.20.160.92	111.30.159.62	OICQ	97	OICQ Protocol
33	2.712132	111.30.159.62	172.20.160.92	OICQ	433	OICQ Protocol

- Frame 3: 129 bytes on wire (1032 bits), 129 bytes captured (1032 bits) on interface \Device\NPF_{C141F1E6-33E...}
- Ethernet II, Src: JuniperN_d2:ff:c2 (44:ec:ce:d2:ff:c2), Dst: LiteonTe_76:c8:11 (3c:91:80:76:c8:11)
- Internet Protocol Version 4, Src: 111.30.159.62, Dst: 172.20.160.92
- User Datagram Protocol, Src Port: 8000, Dst Port: 4008
 - Source Port: 8000
 - Destination Port: 4008
 - Length: 95
 - Checksum: 0x7b83 [unverified]
 - [Checksum Status: Unverified]
 - [Stream index: 0]
 - [Timestamps]
 - UDP payload (87 bytes)
 - OICQ - IM software, popular in China

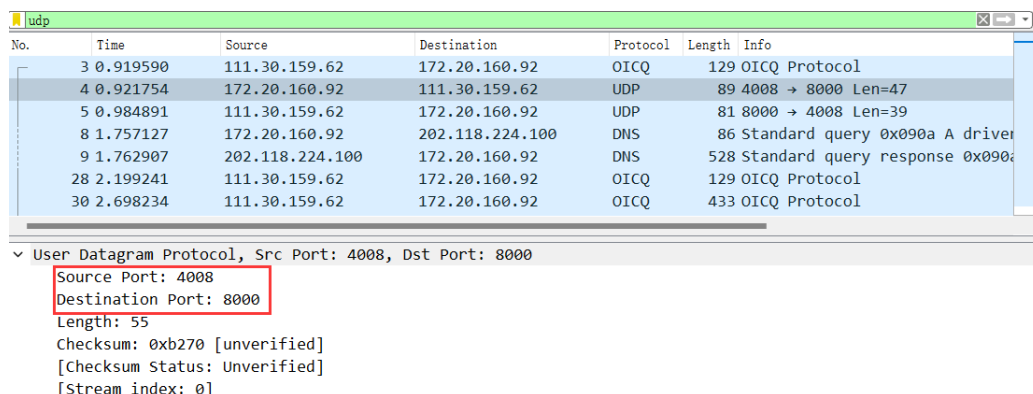
- b) 你的主机ip地址是什么？目的主机ip地址是什么？

主机IP地址为172.20.160.92，目的主机ip地址为111.30.159.62

No.	Time	Source	Destination	Protocol	Length	Info
3	0.919590	111.30.159.62	172.20.160.92	OICQ	129	OICQ Protocol
4	0.921754	172.20.160.92	111.30.159.62	UDP	89	4008 → 8000 Len=47
5	0.984891	111.30.159.62	172.20.160.92	UDP	81	8000 → 4008 Len=39
8	1.757127	172.20.160.92	202.118.224.100	DNS	86	Standard query 0x090a A driver

- Time to Live: 128
- Protocol: UDP (17)
- Header Checksum: 0xdd3f [validation disabled]
- [Header checksum status: Unverified]
- Source Address: 172.20.160.92
- Destination Address: 111.30.159.62
- User Datagram Protocol, Src Port: 4008, Dst Port: 8000
 - Source Port: 4008
 - Destination Port: 8000
 - Length: 55
 - Checksum: 0xb270 [unverified]
 - [Checksum Status: Unverified]
 - [Stream index: 0]

- c) 你的主机发送QQ消息的端口号和QQ服务器的端口号分别是多少？
本人的主机发送QQ消息的端口号是4008；QQ服务器的端口号是8000

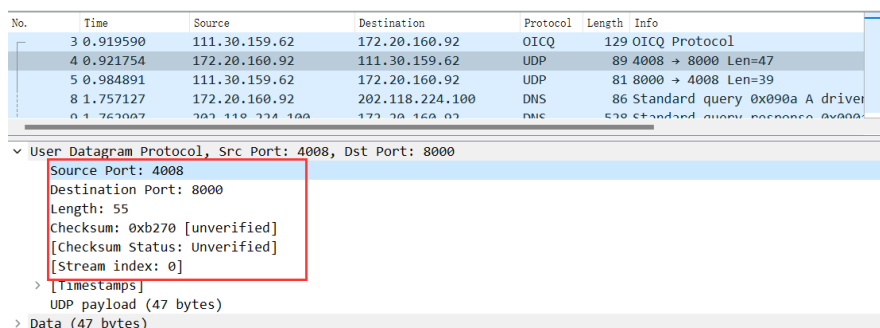


No.	Time	Source	Destination	Protocol	Length	Info
3	0.919590	111.30.159.62	172.20.160.92	OICQ	129	OICQ Protocol
4	0.921754	172.20.160.92	111.30.159.62	UDP	89	4008 → 8000 Len=47
5	0.984891	111.30.159.62	172.20.160.92	UDP	81	8000 → 4008 Len=39
8	1.757127	172.20.160.92	202.118.224.100	DNS	86	Standard query 0x090a A driver
9	1.762907	202.118.224.100	172.20.160.92	DNS	528	Standard query response 0x090a
28	2.199241	111.30.159.62	172.20.160.92	OICQ	129	OICQ Protocol
30	2.698234	111.30.159.62	172.20.160.92	OICQ	433	OICQ Protocol

▼ User Datagram Protocol, Src Port: 4008, Dst Port: 8000

Source Port: 4008
Destination Port: 8000
Length: 55
Checksum: 0xb270 [unverified]
[Checksum Status: Unverified]
[Stream index: 0]

- d) 数据包的格式是什么样的？都包含哪些字段，分别占多少字节？
UDP数据包的格式如下，头部包含源端口号、目的端口号、长度和checksum。各占1个字节



No.	Time	Source	Destination	Protocol	Length	Info
3	0.919590	111.30.159.62	172.20.160.92	OICQ	129	OICQ Protocol
4	0.921754	172.20.160.92	111.30.159.62	UDP	89	4008 → 8000 Len=47
5	0.984891	111.30.159.62	172.20.160.92	UDP	81	8000 → 4008 Len=39
8	1.757127	172.20.160.92	202.118.224.100	DNS	86	Standard query 0x090a A driver
9	1.762907	202.118.224.100	172.20.160.92	DNS	528	Standard query response 0x090a

▼ User Datagram Protocol, Src Port: 4008, Dst Port: 8000

Source Port: 4008
Destination Port: 8000
Length: 55
Checksum: 0xb270 [unverified]
[Checksum Status: Unverified]
[Stream index: 0]

Timestamps
UDP payload (47 bytes)
Data (47 bytes)

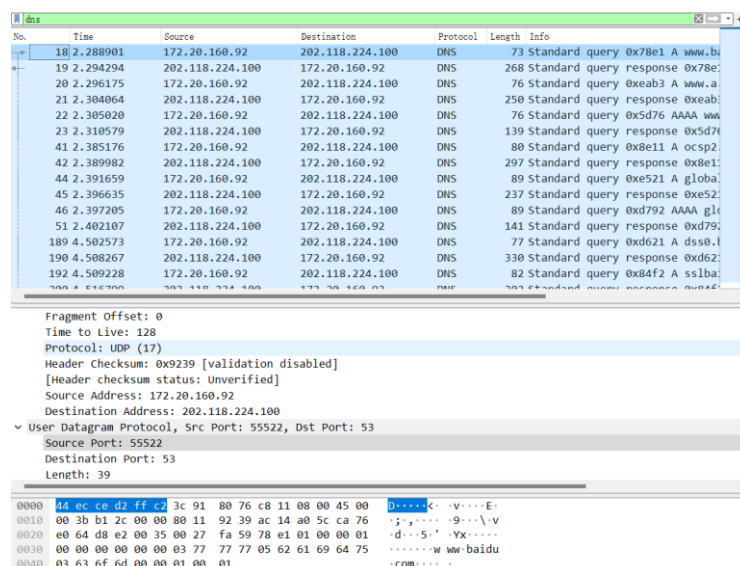
- e) 为什么你发送一个ICQ数据包后，服务器又返回给你的主机一个ICQ数据包？这UDP的不可靠数据传输有什么联系？对比前面的TCP协议分析，你能看出UDP是无连接的吗？

因为UDP是不可靠的数据传输，需要上层协议实现可靠数据传输，因此每次发送ICQ报文后又回复一个ICQ数据包。

UDP是无连接的，可以看到发送数据之前没有连接的建立过程，与TCP不同，因此为无连接数据传输。

(七)利用Wireshark进行DNS协议分析

首先清空dns缓存，在浏览器中访问<https://www.baidu.com>，进行Wireshark抓包。



No.	Time	Source	Destination	Protocol	Length	Info
10	2.288901	172.20.160.92	202.118.224.100	DNS	73	Standard query 0x78e1 A www.b
19	2.294294	202.118.224.100	172.20.160.92	DNS	268	Standard query response 0x78e1
20	2.296175	172.20.160.92	202.118.224.100	DNS	76	Standard query 0xeab3 A www.a
21	2.304064	202.118.224.100	172.20.160.92	DNS	250	Standard query response 0xeab3
22	2.305020	172.20.160.92	202.118.224.100	DNS	76	Standard query 0x5d76 AAAA ww
23	2.310579	202.118.224.100	172.20.160.92	DNS	139	Standard query response 0x5d76
41	2.385176	172.20.160.92	202.118.224.100	DNS	80	Standard query 0x8e11 A ocsp2
42	2.389982	202.118.224.100	172.20.160.92	DNS	297	Standard query response 0x8e11
44	2.391659	172.20.160.92	202.118.224.100	DNS	89	Standard query 0xe521 A globa
45	2.396635	202.118.224.100	172.20.160.92	DNS	237	Standard query response 0xe521
46	2.397205	172.20.160.92	202.118.224.100	DNS	89	Standard query 0xd792 AAAA gl
51	2.402107	202.118.224.100	172.20.160.92	DNS	141	Standard query response 0xd792
189	4.502573	172.20.160.92	202.118.224.100	DNS	77	Standard query 0xd621 A dsso.l
190	4.508267	202.118.224.100	172.20.160.92	DNS	330	Standard query response 0xd621
192	4.509228	172.20.160.92	202.118.224.100	DNS	82	Standard query 0x84f2 A sslba
193	4.516700	202.118.224.100	172.20.160.92	DNS	203	Standard query response 0x84f2

Fragment Offset: 0
Time to Live: 128
Protocol: UDP (17)
Header Checksum: 0x9239 [validation disabled]
[Header checksum status: Unverified]
Source Address: 172.20.160.92
Destination Address: 202.118.224.100

▼ User Datagram Protocol, Src Port: 55522, Dst Port: 53

Source Port: 55522
Destination Port: 53
Length: 39

0000 44 ec c6 d2 ff c2 3c 91 80 76 c8 11 08 00 45 00 0...< .V...E.
0010 00 3b b1 2c 00 00 80 11 92 39 ac 14 a0 5c ca 76 .;...-9...V
0020 e0 64 d8 e2 00 35 00 27 fa 59 78 e1 01 00 00 01 .d...S...Yx...
0030 00 00 00 00 00 00 03 77 77 05 62 61 69 64 75w ww-baidu
0040 03 63 6f 6d 00 00 01 00 01com.....

查询的目的地址均为相同的202.118.224.100，可以推测这是哈工大的dns服务器，经过ip地址查询确实这是一个来自哈尔滨市南岗区的教务网ip地址。

您的IP信息

您查询的IP:	202.118.224.100
所在地理位置:	中国 黑龙江省 哈尔滨市 教育网

问题讨论:

见上面实验结果

心得体会:

在本次的实验中，通过Wireshark分析各种网络协议的执行，可以加深对于各种协议交互过程的理解。并且对于协议中各个字段的作用有了更深的了解。

Wireshark是个十分强大的工具，不仅仅在本门实验中会利用到他的一些知识，在以后的学习工作生活中还有很大的作用。