哈尔滨工业大学

<<计算机网络>>

实验报告

(2018 年度春季学期)

姓名:	许家乐
学号:	1150310329
学院:	计算机学院
教师:	李全龙

实验五 利用 Wireshark 进行协议分析

一、实验目的

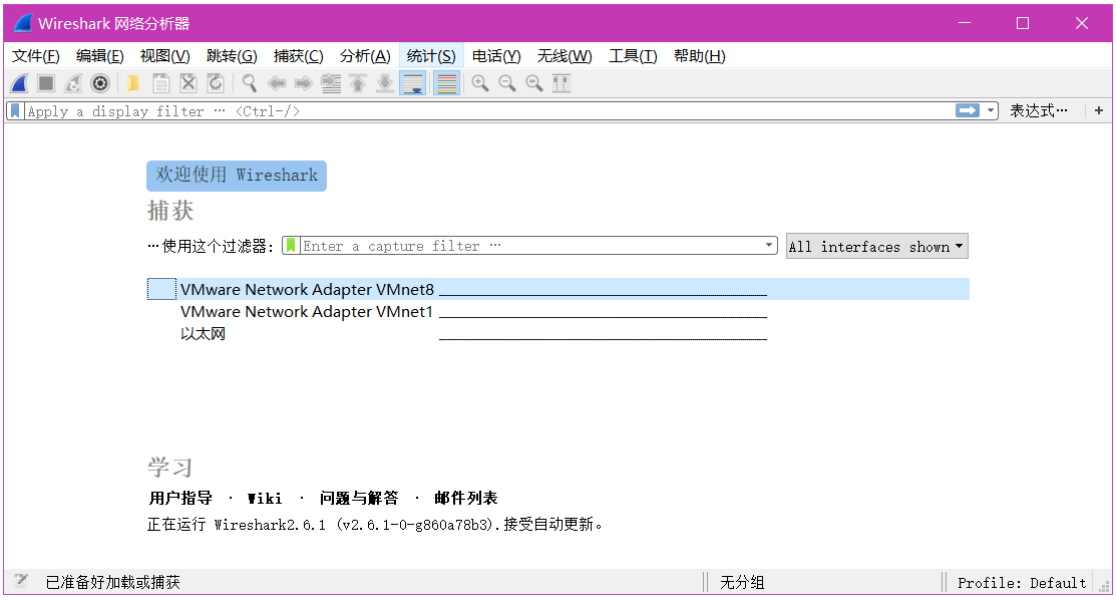
熟悉并掌握 Wireshark 的基本操作，了解网络协议实体间进行交互以及报文交换的情况。

二、实验内容

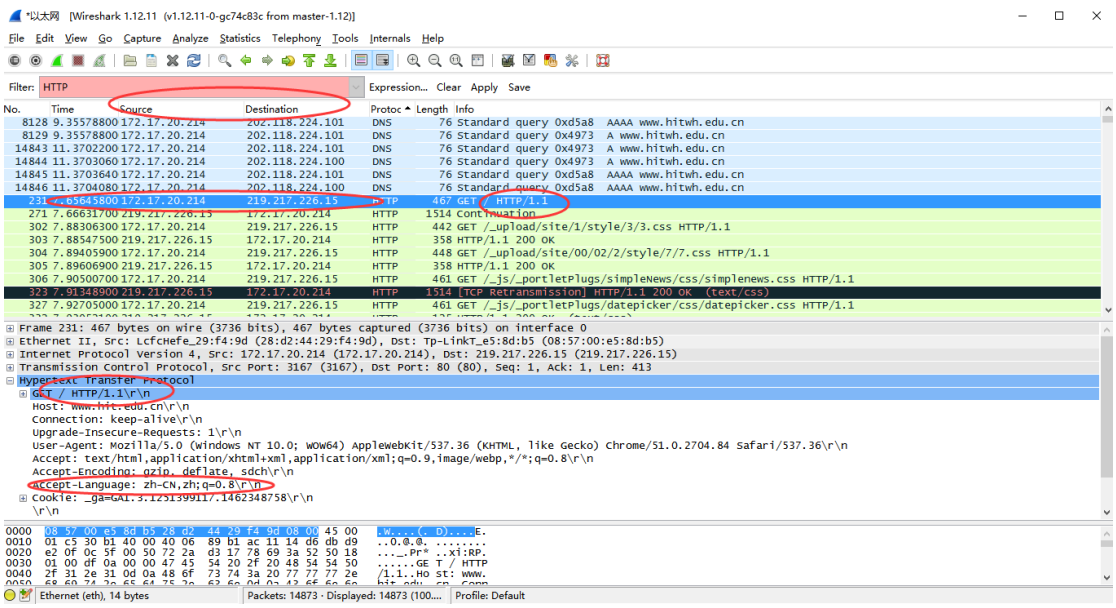
(1)学习 Wireshark 的使用
(2)利用 Wireshark 分析 HTTP 协议
(3)利用 Wireshark 分析 TCP 协议
(4)利用 Wireshark 分析 IP 协议
(5)利用 Wireshark 分析 Ethernet 数据帧
选做内容：
(a)利用 Wireshark 分析 DNS 协议
(b)利用 Wireshark 分析 UDP 协议
(c)利用 Wireshark 分析 ARP 协议

三、实验过程及结果

（一）wireshark 的使用



（二）HTTP 分析
(1) HTTP GET/response 交互



①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？

我的浏览器：HTTP1.1

服务器：HTTP1.1

②你的浏览器向服务器指出它能接收何种语言版本的对象？

zh-CN，简体中文

③你的计算机的 IP 地址是多少？服务器 http://www.hit.edu.cn 的 IP 地址是多少？

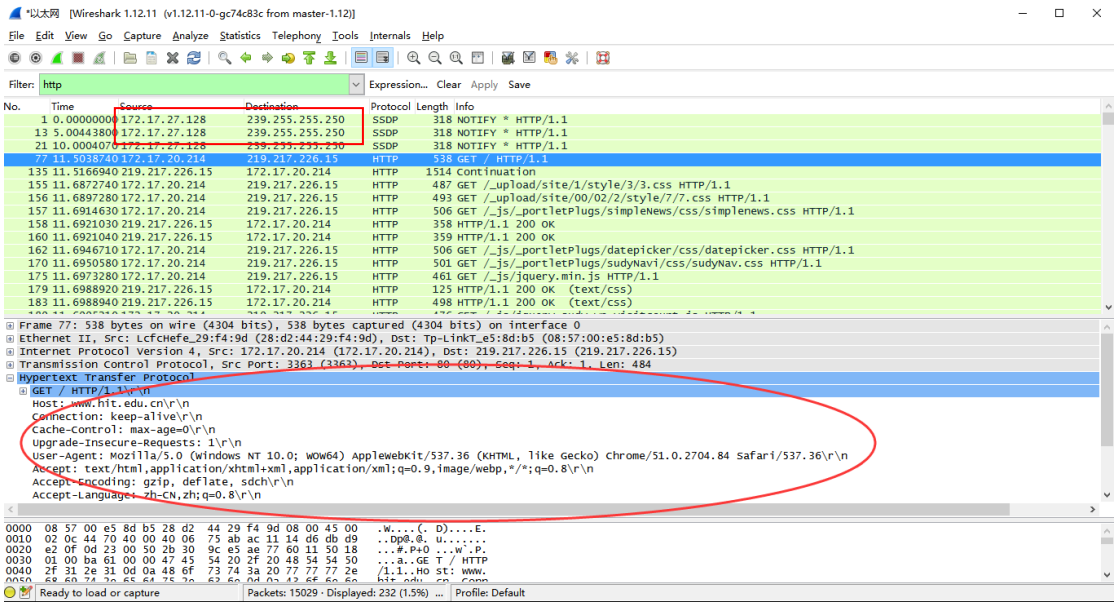
我的计算机：172.17.20.214

服务器：219.217.226.15

④从服务器向你的浏览器返回的状态代码是多少？

200

(2) HTTP 条件 GET/response 交互



①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

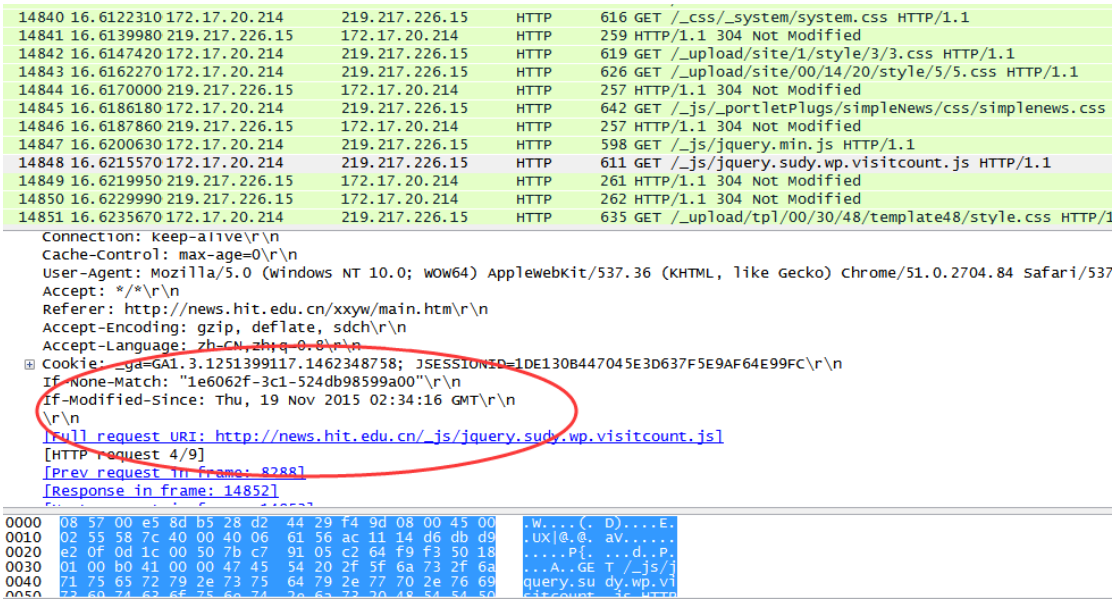
②分析服务器响应报文的内容，服务器是否明确返回了文件的内容？如何获知？

服务器明确返回了内容

HTTP Status Code（状态代码）为 304 时不明确返回文件

HTTP Status Code（状态代码）为 200 时明确返回文件

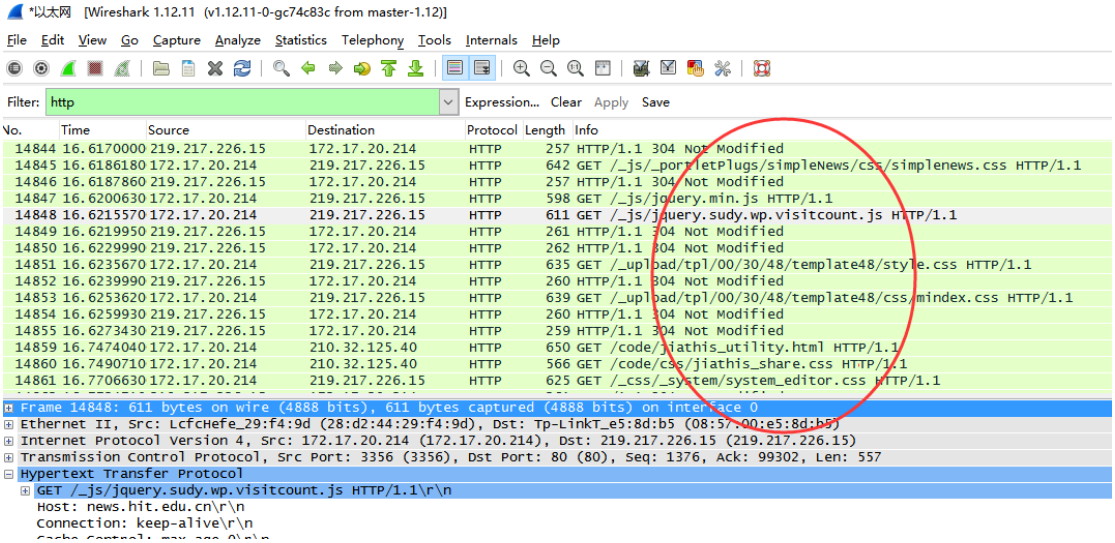
③分析你的浏览器向服务器发出的较晚的"HTTP GET"请求，在该请求报文中是否有一行是：IF-MODIFIED-SINCE？如果有，在该首 部行后面跟着的信息是什么？



有

这个字段后面代表的是时间，即咨询服务器在这个时候之后是否有更新

④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。



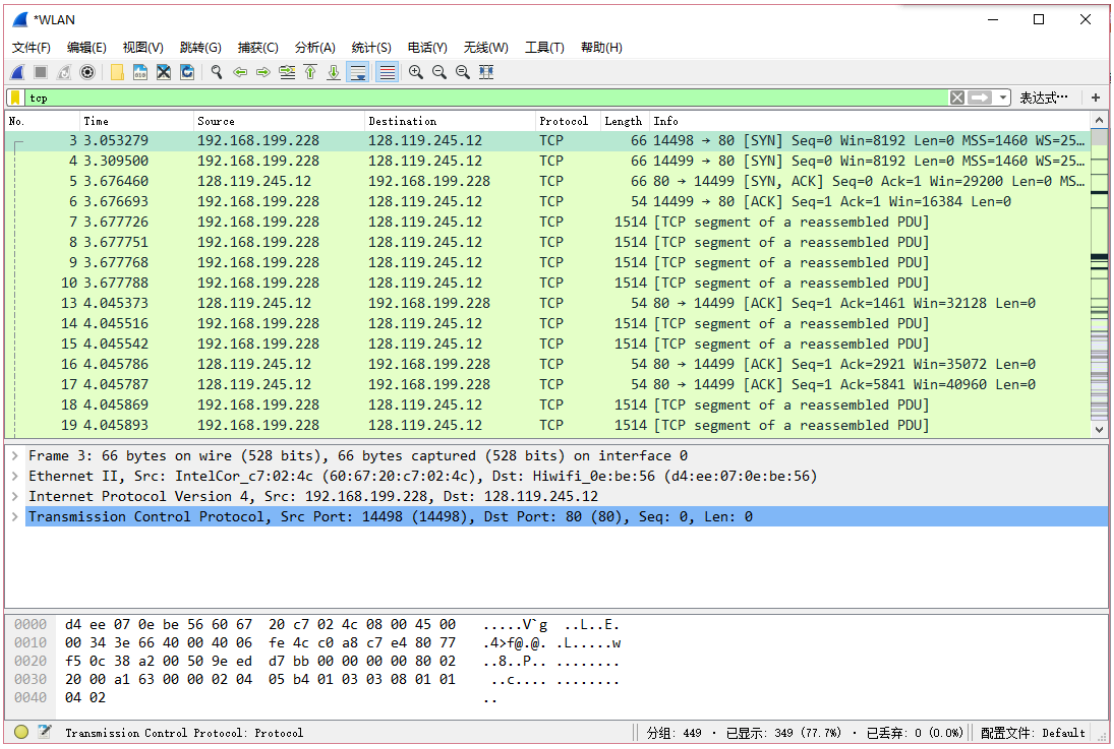
请求响应中的 HTTP 状态代码为 304。

不会明确返回文件，因为根据之前 HTTP 的 GET 请求中 IF-MODIFIED-SINCE

字段内的时间服务器判断结果为 Not Modified，于是客户端可以使用本地这个没有过期的缓存文件。

（三）TCP 分析

报文捕获:



①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？

192.168.199.228

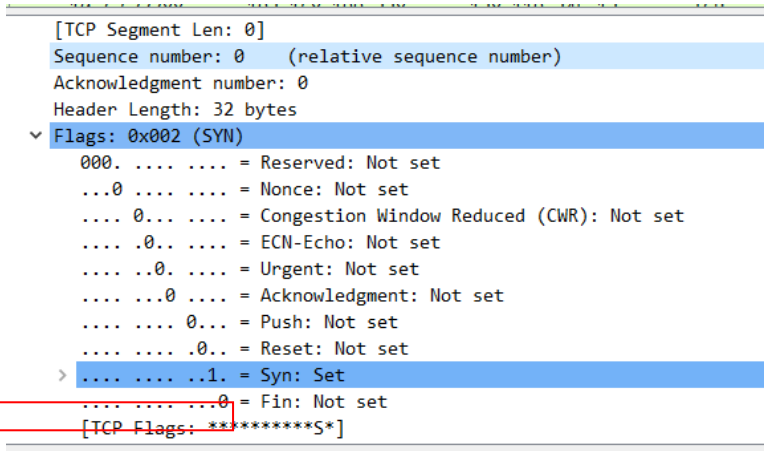
14498

②Gaia.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？

128.119.245.12

80

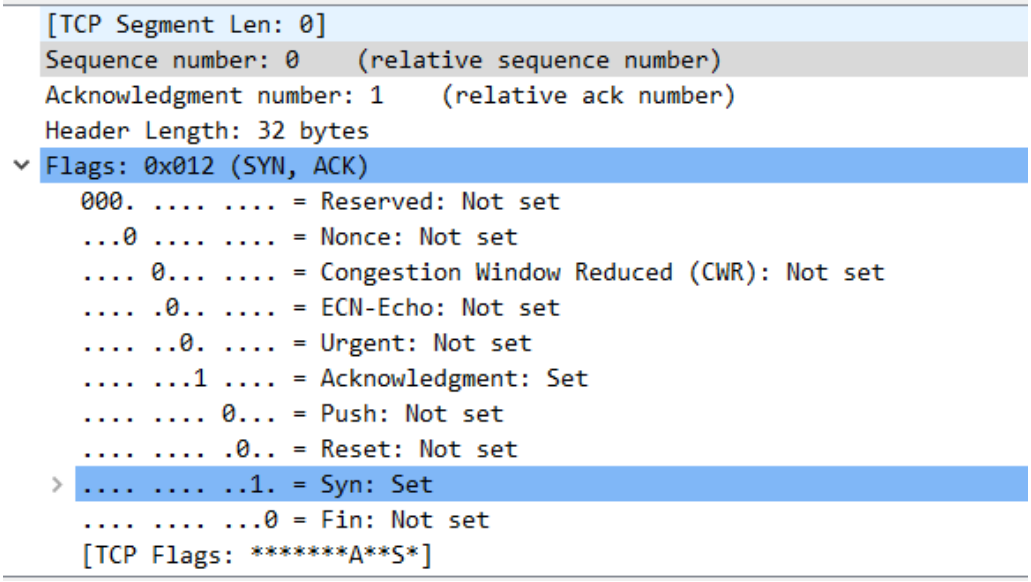
③客户服务器之间用于初始化 TCP 连接的 TCP SYN 报文段的序号（ sequence number）是多少？在该报文段中，是用什么来标示该报文段是 SYN 报文段的？



如图，初始化 TCP 连接的 TCP SYN 报文段的序号是 0；

通过 Flags 标志位，将其中的 SYN 位置为 1，表示该报文段是 SYN 报文段

④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？

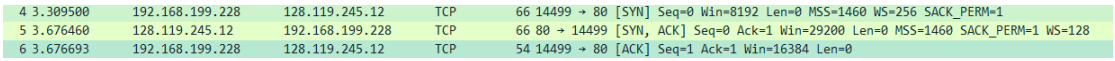


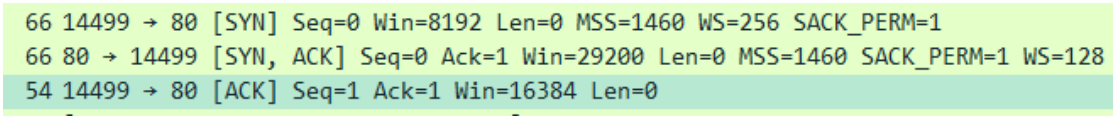
如上图， 服务器端向客户端发送的报文段序号为 0；

服务器发的 acknowledgment number 字段是根据上一次客户端发给服务器的 seq+1 得到的；

通过 Flags 标志位中的 SYN 位和 ACK 位都是 1 来确定该报文段是一个 SYN ACK 报文段的。

⑤你能从捕获的数据包中分析出 tcp 三次握手过程吗？

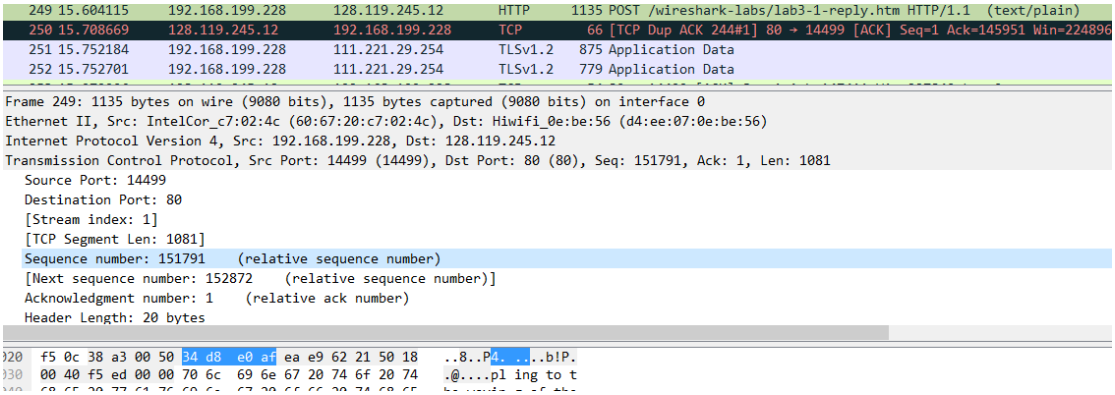




首先客户端向服务器发送 seq=0 的建立连接的请求

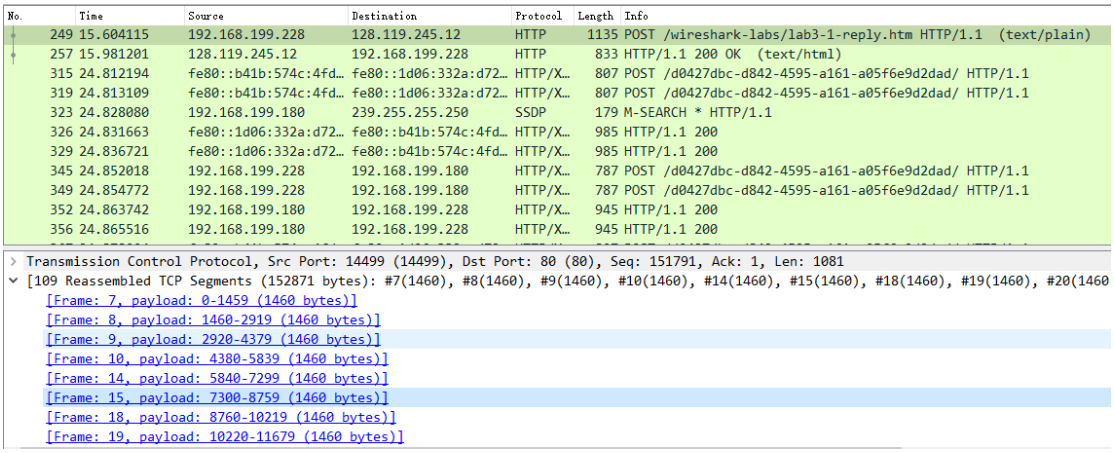
然后服务器向客户端返回 seq=0,ack=0+1=1 的响应

⑥包含 HTTP POST 命令的 TCP 报文段的序号是多少？

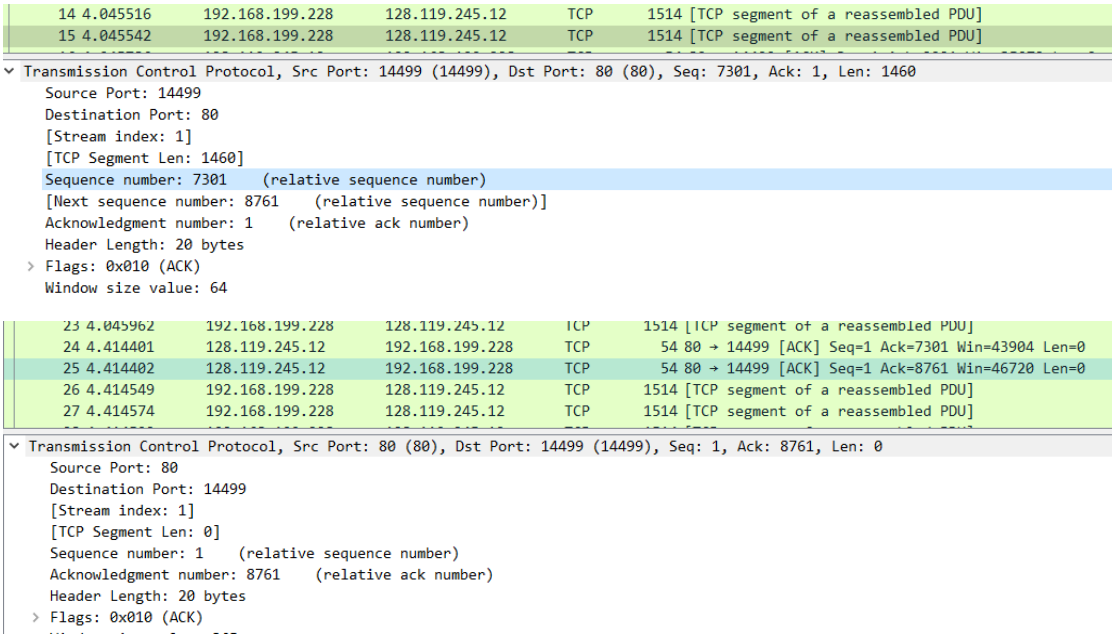


Seq=151791

⑦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？

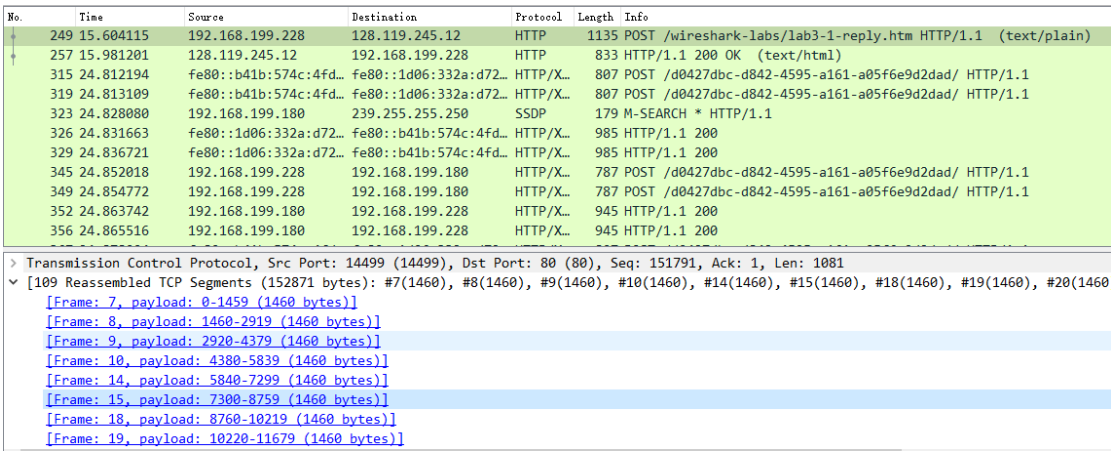


第六个报文段 Seq=7301，在 http post 发送之前， tcp 连接建立之后发送。



对应的 ack 即为服务器返回的第六个 ack。

⑧前六个 TCP 报文段的长度各是多少？

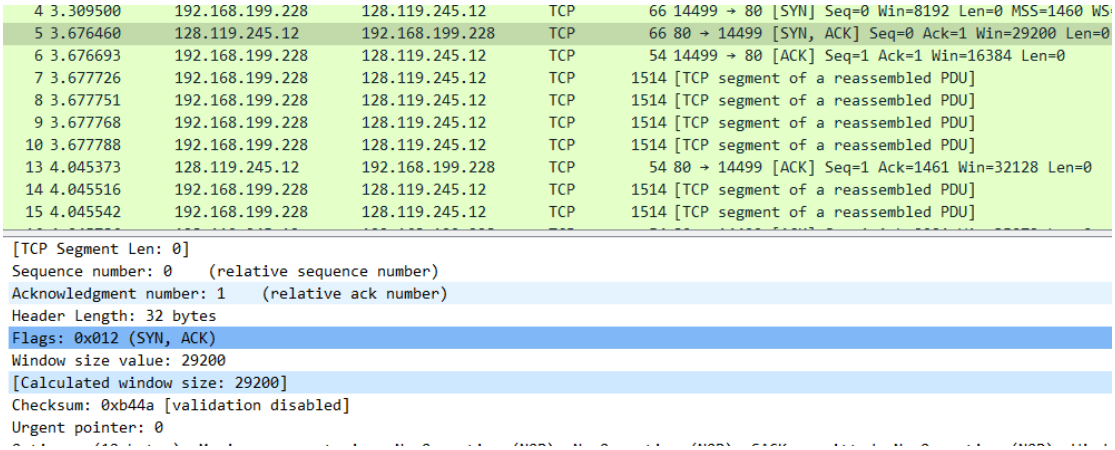


No.	Time	Source	Destination	Protocol	Length	Info
249	15.604115	192.168.199.228	128.119.245.12	HTTP	1135	POST /wireshark-labs/lab3-1-reply.htm HTTP/1.1 (text/plain)
257	15.981201	128.119.245.12	192.168.199.228	HTTP	833	HTTP/1.1 200 OK (text/html)
315	24.812194	fe80::b41b:574c:4fd…	fe80::1d06:332a:d72…	HTTP/X…	807	POST /d0427dbc-d842-4595-a161-a05f6e9d2dad/ HTTP/1.1
319	24.813109	fe80::b41b:574c:4fd…	fe80::1d06:332a:d72…	HTTP/X…	807	POST /d0427dbc-d842-4595-a161-a05f6e9d2dad/ HTTP/1.1
323	24.828080	192.168.199.180	239.255.255.250	SSDP	179	M-SEARCH * HTTP/1.1
326	24.831663	fe80::1d06:332a:d72…	fe80::b41b:574c:4fd…	HTTP/X…	985	HTTP/1.1 200
329	24.836721	fe80::1d06:332a:d72…	fe80::b41b:574c:4fd…	HTTP/X…	985	HTTP/1.1 200
345	24.852018	192.168.199.228	192.168.199.180	HTTP/X…	787	POST /d0427dbc-d842-4595-a161-a05f6e9d2dad/ HTTP/1.1
349	24.854772	192.168.199.228	192.168.199.180	HTTP/X…	787	POST /d0427dbc-d842-4595-a161-a05f6e9d2dad/ HTTP/1.1
352	24.863742	192.168.199.180	192.168.199.228	HTTP/X…	945	HTTP/1.1 200
356	24.865516	192.168.199.180	192.168.199.228	HTTP/X…	945	HTTP/1.1 200


```
> Transmission Control Protocol, Src Port: 14499 (14499), Dst Port: 80 (80), Seq: 151791, Ack: 1, Len: 1081
v [109 Reassembled TCP Segments (152871 bytes): #7(1460), #8(1460), #9(1460), #10(1460), #14(1460), #15(1460), #18(1460), #19(1460), #20(1460
    [Frame: 7, payload: 0-1459 (1460 bytes)]
    [Frame: 8, payload: 1460-2919 (1460 bytes)]
    [Frame: 9, payload: 2920-4379 (1460 bytes)]
    [Frame: 10, payload: 4380-5839 (1460 bytes)]
    [Frame: 14, payload: 5840-7299 (1460 bytes)]
    [Frame: 15, payload: 7300-8759 (1460 bytes)]
    [Frame: 18, payload: 8760-10219 (1460 bytes)]
    [Frame: 19, payload: 10220-11679 (1460 bytes)]
```

⑨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？

如图，接收端公示的最小的可用缓存空间是 29200，该窗口大小会一直增加，所以不会出现接收端的缓存是否仍然不够用的情况。

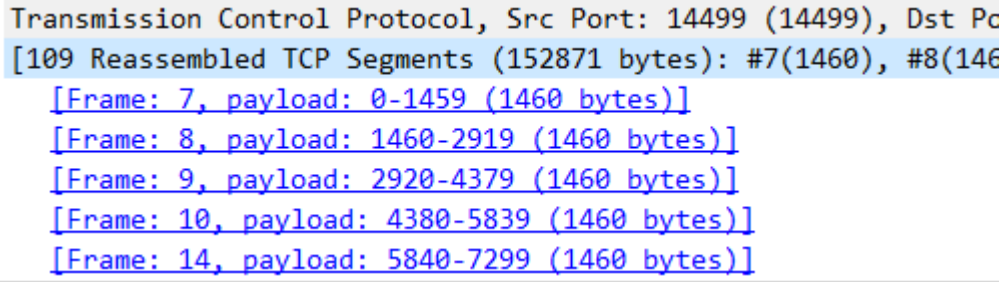


4	3.309500	192.168.199.228	128.119.245.12	TCP	66	14499 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS
5	3.676460	128.119.245.12	192.168.199.228	TCP	66	80 → 14499 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0
6	3.676693	192.168.199.228	128.119.245.12	TCP	54	14499 → 80 [ACK] Seq=1 Ack=1 Win=16384 Len=0
7	3.677726	192.168.199.228	128.119.245.12	TCP	1514	[TCP segment of a reassembled PDU]
8	3.677751	192.168.199.228	128.119.245.12	TCP	1514	[TCP segment of a reassembled PDU]
9	3.677768	192.168.199.228	128.119.245.12	TCP	1514	[TCP segment of a reassembled PDU]
10	3.677788	192.168.199.228	128.119.245.12	TCP	1514	[TCP segment of a reassembled PDU]
13	4.045373	128.119.245.12	192.168.199.228	TCP	54	80 → 14499 [ACK] Seq=1 Ack=1461 Win=32128 Len=0
14	4.045516	192.168.199.228	128.119.245.12	TCP	1514	[TCP segment of a reassembled PDU]
15	4.045542	192.168.199.228	128.119.245.12	TCP	1514	[TCP segment of a reassembled PDU]


```
[TCP Segment Len: 0]
Sequence number: 0    (relative sequence number)
Acknowledgment number: 1    (relative ack number)
Header Length: 32 bytes
Flags: 0x012 (SYN, ACK)
Window size value: 29200
[Calculated window size: 29200]
Checksum: 0xb44a [validation disabled]
Urgent pointer: 0
```

⑩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？

没有出现重传，因为客户端发送的报文序列号没有出现重复。



```
Transmission Control Protocol, Src Port: 14499 (14499), Dst Pc
[109 Reassembled TCP Segments (152871 bytes): #7(1460), #8(146
    [Frame: 7, payload: 0-1459 (1460 bytes)]
    [Frame: 8, payload: 1460-2919 (1460 bytes)]
    [Frame: 9, payload: 2920-4379 (1460 bytes)]
    [Frame: 10, payload: 4380-5839 (1460 bytes)]
    [Frame: 14, payload: 5840-7299 (1460 bytes)]
```

⑪TCP 连接的 throughput 是多少？请写出你的计算过程

由图可知，发送数据总的长度为 152871B+109*54B=158757B

发送时间间隔约为 1.673847s

因此吞吐量为 158757B/1.673847s=94845.59bps

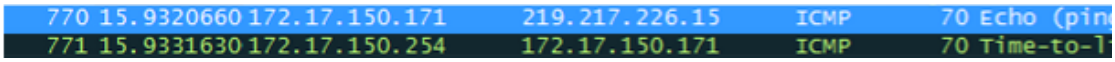
（四）IP 分析

A. 对捕获的数据包进行分析

1. 在你的捕获窗口中，应该能看到由你的主机发出的一系列 ICMP Echo Request 包和中间路由器返回的一系列 ICMP TTL-exceeded 消息。选 择第一个你的主

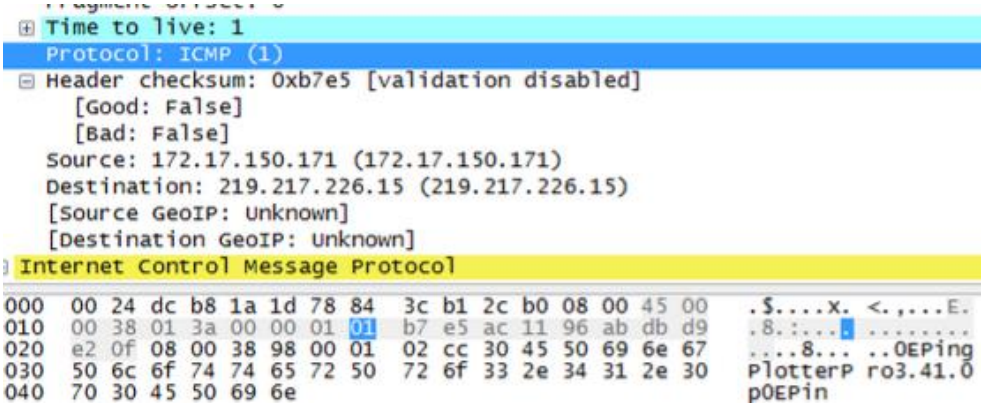
机发出的 ICMP Echo Request 消息，在 packet details 窗口 展开数据包的 Internet Protocol 部分

①你主机的 IP 地址是什么？



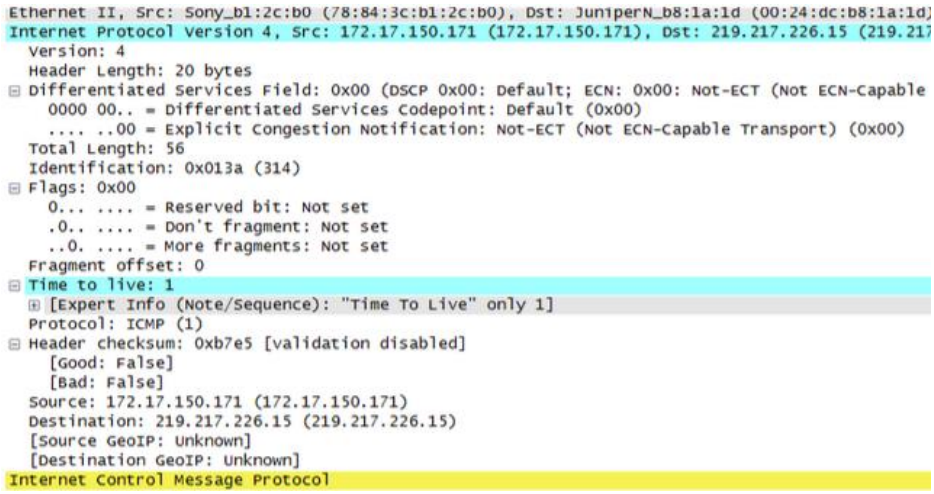
172.17.150.171

②在 IP 数据包头中，上层协议（upper layer）字段的值是什么？



01.

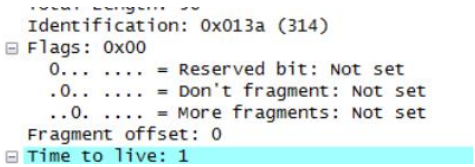
③IP 头有多少字节？该 IP 数据包的净载为多少字节？并解释你是怎样确定该 IP 数据包的净载大小的？



IP 头有 20 字节。

IP 包的净载为 Total Length-Header Length=56B-20B=36B

④该 IP 数据包分片了吗？解释你是如何确定该 P 数据包是否进行了分片



没有，分片位移为 0，More fragments 为 0 表示后面无分段。

2.单击 Source 列按钮，这样将对捕获的数据包按源 IP 地址排序。选择第一个你的主机发出的 ICMP Echo Request 消息，在 packet details 窗口展开数据包的 Internet Protocol 部分。在“listing of captured packets”窗口，你会看到许多后续的 ICMP 消息（或许还有你主机上运行的其他协议的数据包）

①你主机发出的一系列 ICMP 消息中 IP 数据报中哪些字段总是发生改变？

ID、TTL、Header checksun 这三个字段总在变化。

②哪些字段必须保持常量？哪些字段必须改变？为什么？

必须改变：

ID 鉴别码， 用于区分不同的数据包；

TTL 来自于 traceroute 的要求， 用来测试路径上的路由信息；

Header Checksum 首部校验和，前面的字段改变，该值也必须跟着改变；

必须保持常量：

除以上(ID,TTL,Header Checksum) 外的字段保持常量。

③描述你看到的 IP 数据包 Identification 字段值的形式。

16 位，在某一范围内是+1 递增的。

3.找到由最近的路由器（第一跳）返回给你主机的 ICMPTime-to-live exceeded 消息。

思考下列问题：

①Identification 字段和 TTL 字段的值是什么？ 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？

不变，IP 是无连接服务，相同的标识是为了分段后组装成同一段，给同一个主机返回的 ICMP，标识不代表序号， TTL 消息是相同的，因此 Identification 不变;因为是第一跳路由器发回的数据报，故 TTL 是最大值减 1，总是等于 254。

4.单击 Time 列按钮，这样将对捕获的数据包按时间排序。找到在将包大小改为 2000 字节后你的主机发送的第一个 ICMP Echo Request 消息。

思考下列问题：

①该消息是否被分解成不止一个 IP 数据报？

```
[2 IPv4 Fragments (1980 bytes): #2850(1480), #2851(500)]
  [Frame: 2850, payload: 0-1479 (1480 bytes)]
  [Frame: 2851, payload: 1480-1979 (500 bytes)]
  [Fragment count: 2]
```

是的，该消息被分解成了 2 片

②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

```
Flags: 0x01 (More Fragments)
  0... .... = Reserved bit: Not set
  .0.. .... = Don't fragment: Not set
  ..1. .... = More fragments: Set
```

More fragments=1 表示分片了且不是最后一片，该分片的长度是 1500B

C. 找到在将包大小改为 3500 字节后你的主机发送的第一个 ICMP Echo Request 消息。

思考下列问题：

①原始数据包被分成了多少片？

```
[3 IPv4 Fragments (3480 bytes): #5036(1480), #5037(1480), #5038(520)]
  [Frame: 5036, payload: 0-1479 (1480 bytes)]
  [Frame: 5037, payload: 1480-2959 (1480 bytes)]
  [Frame: 5038, payload: 2960-3479 (520 bytes)]
  [Fragment count: 3]
```

三片

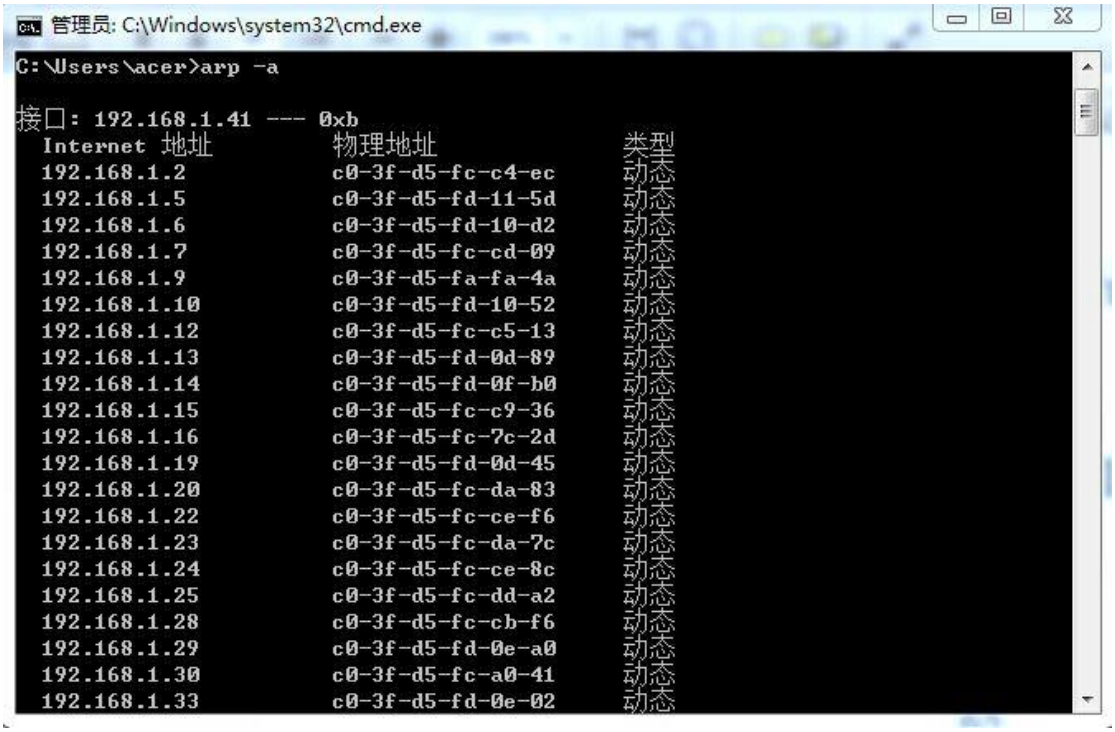
②这些分片中 IP 数据报头部哪些字段发生了变化？

前 2 个分片More fragments=1，后两个分片offset 变为 1480 和 2960

（五）抓取 ARP 数据包

（1）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?

输入 apr －a 查看主机上 ARP 缓存的内容，结果如下图所示（截图显示部分）：



ARP 缓存中的每一列分别表示 IP 地址所对应的物理地址和类型（动态配置或静态配置）

（2）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：

①ARP 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 数据包格式如下图：

由 9 部分构成，分别是硬件类型（2 字节），协议类型（2 字节），硬件地址长度（1 字节），协议地址长度（1 字节），OP（2 字节），发送端 MAC 地址（6 字节），发送端 IP 地址（4 字节），目的 MAC 地址（6 字节），目的 IP 地址（4 字节）。

截取的一个 ARP 数据包如下：

```
Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: 50:78:1c:14:8b:a6 (50:78:1c:14:8b:a6)
    Sender IP address: 192.168.1.254 (192.168.1.254)
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 192.168.1.134 (192.168.1.134)
```

②如何判断一个 ARP 数据是请求包还是应答包？

通过 OP 字段。当 OP 字段值为 0x0001 时是请求包，当 OP 字段值为 0x0002 时是应答包。

③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？

因为进行 ARP 查询时并不知道目的 IP 地址对应的 MAC 地址，所以需要广播查询；而 ARP 响应报文知道查询主机的 MAC 地址（通过查询主机发出的查询报文获得），且局域网中的其他主机不需要此次查询的结果，因此 ARP 响应要在一个有着明确目的局域网地址的帧中传送。

（五）抓取 UDP 数据包

①消息是基于 UDP 的还是 TCP 的?

UDP。如图（User Datagram Protocol）：

```
User Datagram Protocol, Src Port: 4001 (4001), Dst Port: 8000 (8000)
```

②你的主机 ip 地址是什么？目的主机 ip 地址是什么?

我的主机 IP 地址：192.168.1.41

目的主机 IP 地址：182.254.110.92

```
91 1.677293     172.17.154.141     112.175.245.200     UDP     79 65470 → 9803  Len=37
```

③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？

发送 QQ 消息端口号：4001

QQ 服务器端口号：8000

```
User Datagram Protocol, Src Port: 4001 (4001), Dst Port: 8000 (8000)
    Source Port: 4001 (4001)
    Destination Port: 8000 (8000)
    Length: 55
```

④数据报的格式是什么样的？都包含哪些字段，分别占多少字节？

UDP 数据报格式如下图：

UDP 数据报格式有首部和数据两个部分。首部很简单，共 8 字节。包括：

源端口号： 2 字节

目的端口号： 2 字节

长度： 2 字节， UDP 用户数据报的总长度，以字节为单位。

校验和： 2 字节，用于校验 UDP 数据报的数字段和包含 UDP 数据报首部的“伪首部”。其校验方法同 IP 分组首部中的首部校验和。

⑤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？

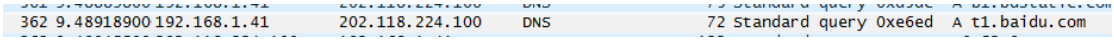
因为服务器需返回接收的结果给客户端。

因为服务器只提供了一次返回的 ACK，所以不保证数据一定送达。

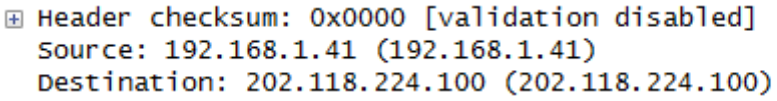
可以看出。 UDP 数据包没有序列号，因此不能像 TCP 协议那样先握手再发送数据，因为每次只发送一个数据报，然后等待服务器响应。

（六）利用 WireShark 进行 DNS 协议分析

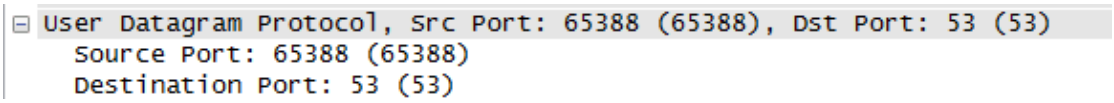
① 打开浏览器，输入 www.baidu.com,DNS 查询消息如下图：



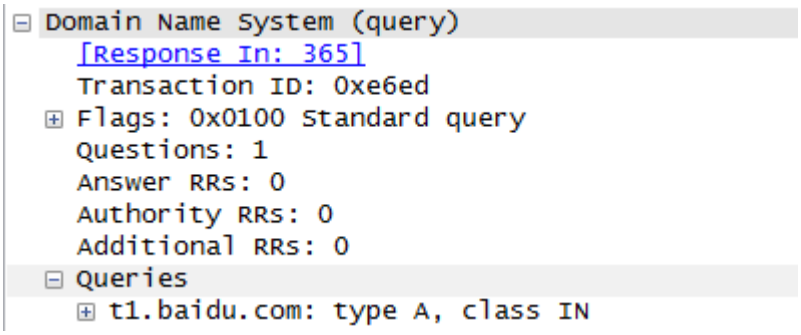
② 我的电脑 IP 地址:192.168.1.41,本地域名服务器 IP 地址:202.118.224.100.如图：



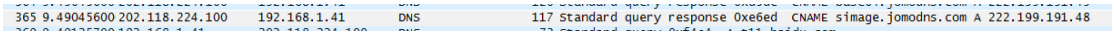
③ UDP 报文的源端口号 65388，目的端口号 53



④ DNS 查询报文内容如下图，表示查询主机域名为 t1.baidu.com 的主机的 IP 地址



⑤ DNS 回复信息：



主机域名为 t1.baidu.com 的主机 IP 地址为：222.199.191.48

四、实验心得

通过本次实验，我有以下几点收获：

①对各个协议的报文格式以及工作过程都有了进一步的认识与掌握；

②了解了网络协议实体间进行交互及报文交换的情况；

③掌握了使用 Wireshark 工具进行抓包的方法