哈爾濱工業大學
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	王子奕		院系	计算学部		
班级	1937101		学号	1190200121		
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	2021 年 11 月 21 日		
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						

计算学部

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

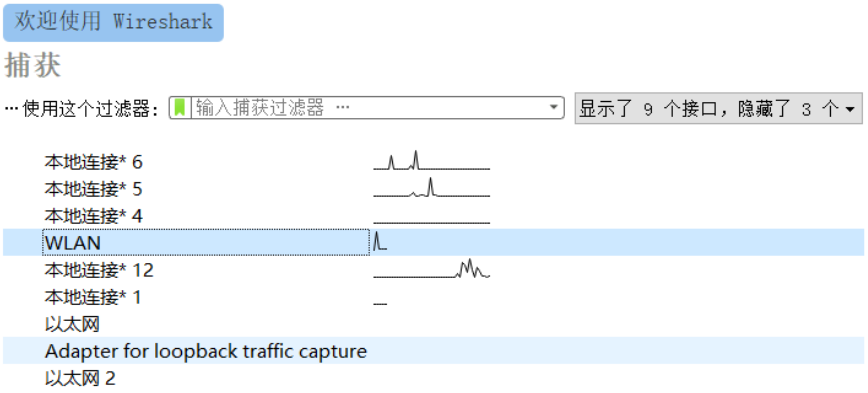
选做内容:

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

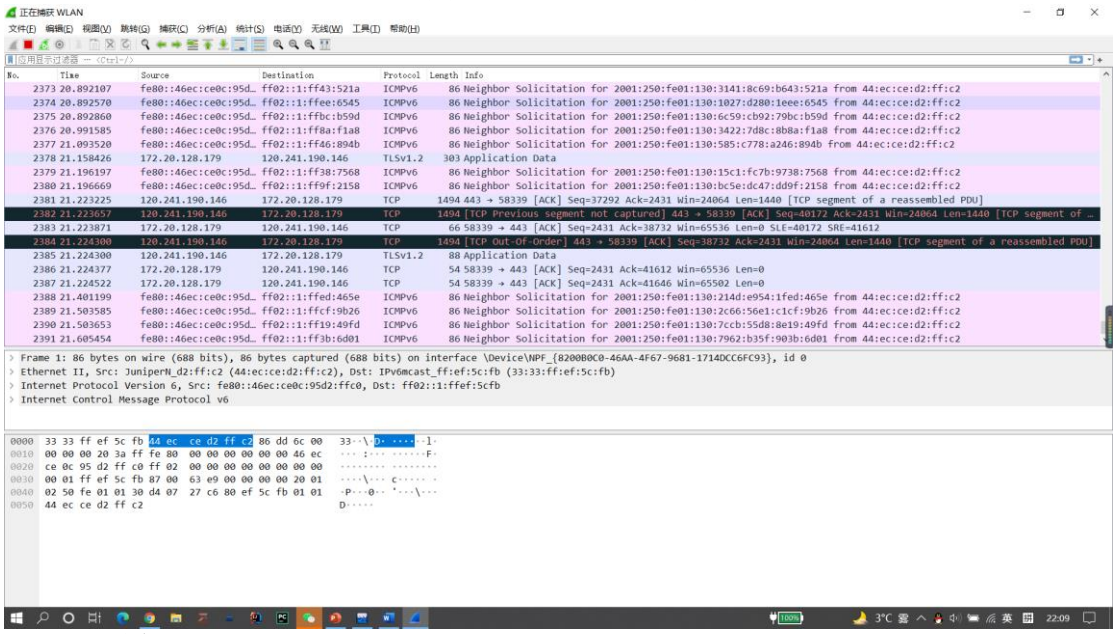
实验过程：

一、Wireshark 的使用

首先在 Wireshark 官网 https://www.wireshark.org/download.html 下载 Wireshark，之后捕获器选择接口进行捕获。由于我本地电脑使用的是无线网卡进行上网，所以我选择捕获 WLAN 接口。



在选定接口后，我可以在界面中看见 Wireshark 的抓包信息。



二、HTTP 分析

1) HTTP GET/response 交互

首先启动浏览器，然后启动 Wireshark 分组嗅探器。在窗口的显示过滤说明处输入“http”，分组列表子窗口中将只显示所俘获到的HTTP 报文。之后开始 Wireshark 分组俘获。

在打开的浏览器中访问 http://today.hit.edu.cn，捕获HTTP报文，之后停止分组俘获。最后将捕获的报文保存到文件中，用于后续分析。

2) HTTP 条件 GET/response 交互

首先启动浏览器，清空浏览器的缓存，然后启动 Wireshark 分组嗅探器。在窗口的显示过滤说明处输入“http”，分组列表子窗口中将只显示所俘获到的HTTP 报文。之后开始 Wireshark 分组俘获。

在打开的浏览器中刷新 http://today.hit.edu.cn 的页面，捕获 HTTP报文，之后停止分组俘获。最后将捕获的报文保存到文件中，用于后续分析。

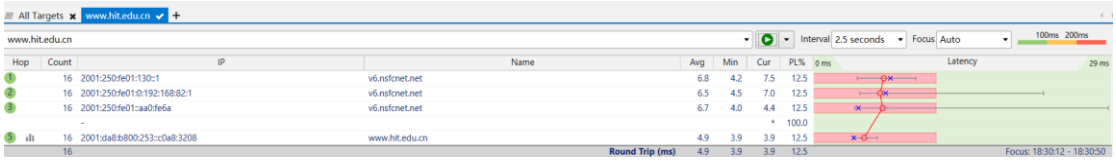
三、TCP 分析

首先下载位于 http://gaia.cs.umass.edu/wireshark-labs/alice.txt 的 alice.txt 文本文件，然后打开 http://gaia.cs.umass.edu/Wireshark-labs/TCP-Wireshark-file1.html ，选中自己下载的文本文件。

此时打开启动Wireshark，开始分组俘获。在浏览器中，单击“Upload alice.txt file”按钮，将文件上传到 gaia.cs.umass.edu 的服务器。在文件上传完毕，一个简短的贺词信息将显示在浏览器窗口中后，停止 Wireshark 的捕获。最后将捕获的报文保存到文件中，用于分析。

四、IP 分析

打开 Wireshark 进行数据包的捕获，之后使用 pingplotter 依次向 hit.edu.cn 发送大小为 56 字节、2000 字节和 3500 字节的 IP 数据包。最后将捕获的报文保存到文件中，用于后续分析。



五、抓取 ARP 数据包

首先利用 arp 指令查看本地的 ARP 缓存表。

```
接口: 172.20.203.229 --- 0xa
  Internet 地址         物理地址              类型
  172.20.0.1            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
  239.255.255.250       01-00-5e-7f-ff-fa     静态
  255.255.255.255       ff-ff-ff-ff-ff-ff     静态
```

之后开启 Wireshark 的分组捕获，在命令行中 ping 192.168.1.82，ping 通后停止捕获。

```
C:\Users\65448>ping www.baidu.com

正在 Ping www.a.shifen.com [39.156.66.18] 具有 32 字节的数据:
来自 39.156.66.18 的回复: 字节=32 时间=86ms TTL=50
来自 39.156.66.18 的回复: 字节=32 时间=25ms TTL=50
来自 39.156.66.18 的回复: 字节=32 时间=33ms TTL=50
来自 39.156.66.18 的回复: 字节=32 时间=49ms TTL=50

39.156.66.18 的 Ping 统计信息:
    数据包: 已发送 = 4, 已接收 = 4, 丢失 = 0 (0% 丢失),
往返行程的估计时间(以毫秒为单位):
    最短 = 25ms, 最长 = 86ms, 平均 = 48ms
```

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，之后将这段时间捕获的报文保存到文件中。

七、利用 Wireshark 进行 DNS 协议分析

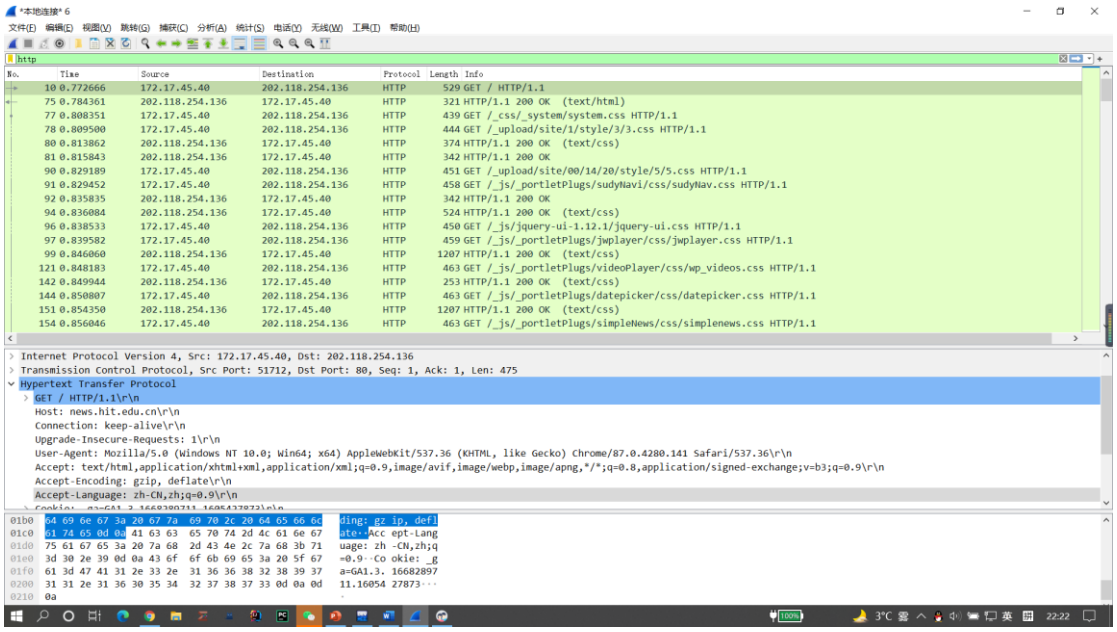
首先打开 Wireshark 进行抓包，在浏览器中访问 www.baidu.com ，完成后停止抓包，保存到文件，捕获 DNS 报文。

实验结果：

一、HTTP 分析

1) HTTP GET/response 交互

利用 Wireshark，我们可以获得如下的报文，截图如下：



1. 通过截图，可以发现我的浏览器使用的是 HTTP/1.1 协议；访问的服务器使用的也是 HTTP/1.0 协议。

```
78 0.809500     172.17.45.40      202.118.254.136   HTTP   444 GET /_upload/site/1/style/3/3.css HTTP/1.1
80 0.813862     202.118.254.136   172.17.45.40      HTTP   374 HTTP/1.1 200 OK  (text/css)
```

2. 浏览器向服务器指明其可以接收的语言如下，即：

Accept-Language zh-CN,zh;q=0.9\n

```
> Transmission Control Protocol, Src Port: 51712, Dst Port: 80, Seq: 1, Ack:
v Hypertext Transfer Protocol
  > GET / HTTP/1.1\r\n
    Host: news.hit.edu.cn\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: zh-CN,zh;q=0.9\r\n
```

3. 我的 IP 地址为 172.17.45.40，服务器的 IP 地址为 202.118.254.136。

```
Source Address: 172.17.45.40
Destination Address: 202.118.254.136
```

4. 服务器向本机浏览器返回的状态码为200。

```
HTTP/1.1 200 OK\r\n
> [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r
  Response Version: HTTP/1.1
  Status Code: 200
  [Status Code Description: OK]
  Response Phrase: OK
Content-Type: text/css\r\n
```

2) HTTP 条件 GET/response 交互

1. 在清除浏览器缓存之后，访问 today.hit.edu.cn，发出的第一个 HTTP GET 请求如下，可以发现不包含 IF-MODIFIED-SINCE 行：

在后续的GET请求中，头部信息出现了IF-MODIFIED-SINCE，其值表示上一次修改的时间

```
    Request URI: /
    Request Version: HTTP/1.1
Host: news.hit.edu.cn\r\n
Connection: keep-alive\r\n
Upgrade-Insecure-Requests: 1\r\n
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Accept: text/html,application/xhtml+xml,application/xm
Accept-Encoding: gzip, deflate\r\n
Accept-Language: zh-CN,zh;q=0.9\r\n
Cookie: _ga=GA1.3.1668289711.1605427873\r\n
\r\n
[Full request URI: http://news.hit.edu.cn/]
[HTTP request 1/28]
[Response in frame: 192]
[Next request in frame: 194]
```

```
Accept: image/webp,*/*
Accept-Encoding: gzip,
Accept-Language: zh-CN
Cache-Control: max-age
Dnt: 1\r\n
If-Modified-Since: Tue
```

2. 服务器接下来的服务器响应报文返回了明确的内容（若干 json 文件，通过右键-追踪流-HTTP流可以知道），状态码均为 200。

```
202 23.619422   172.17.45.40     202.118.254.136  HTTP  451 GET /_upload/site/00/14/20/style/5/5.css HTTP/1.1
203 23.619880   172.17.45.40     202.118.254.136  HTTP  458 GET /_js/_portletPlugs/sudyNavi/css/sudyNav.css HTTP/1
204 23.624027   202.118.254.136  172.17.45.40     HTTP  342 HTTP/1.1 200 OK
206 23.625118   202.118.254.136  172.17.45.40     HTTP  524 HTTP/1.1 200 OK  (text/css)
208 23.625276   172.17.45.40     202.118.254.136  HTTP  450 GET /_js/jquery-ui-1.12.1/jquery-ui.css HTTP/1.1
209 23.626307   172.17.45.40     202.118.254.136  HTTP  459 GET /_js/_portletPlugs/jwplayer/css/jwplayer.css HTTP/
```

3. 服务器对于较晚的 HTTP GET 请求，其返回的状态码为 304，而且并返回的内容极短，并没有包含具体的内容。

```
4 HTTP/1.1 304 Not Modified
4 HTTP/1.1 304 Not Modified
```

二、TCP 分析

1. 向 gaia.cs.umass.edu 服务器传送文件的客户端（源主机）主机的 IP 地址与 TCP 端口号为 172.20.163.51 和 51370。

2. gaia.cs.umass.edu 服务器 IP 地址为 128.119.245.12，接收端口号为80。

```
Internet Protocol Version 4, Src: 172.20.163.51, Dst: 128.119.245.12
Transmission Control Protocol, Src Port: 51370, Dst Port: 80, Seq: 152548, Ack: 1, Len: 485
```

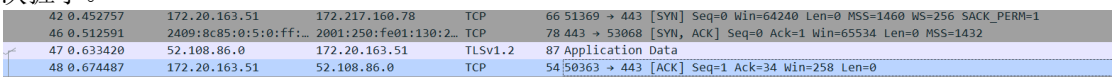
3. 客户服务器之间用于初始化 TCP 连接的 TCP SYN 报文段的序号为 0（绝对值为 2911508834）；该报文段将SYN标志位置为 1，表示该报文段为 SYN 段用于 TCP 建立连接。

```
Sequence Number: 0    (relative sequence number)
Sequence Number (raw): 2911508834
[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
   .... .... ...0 = Fin: Not set
```

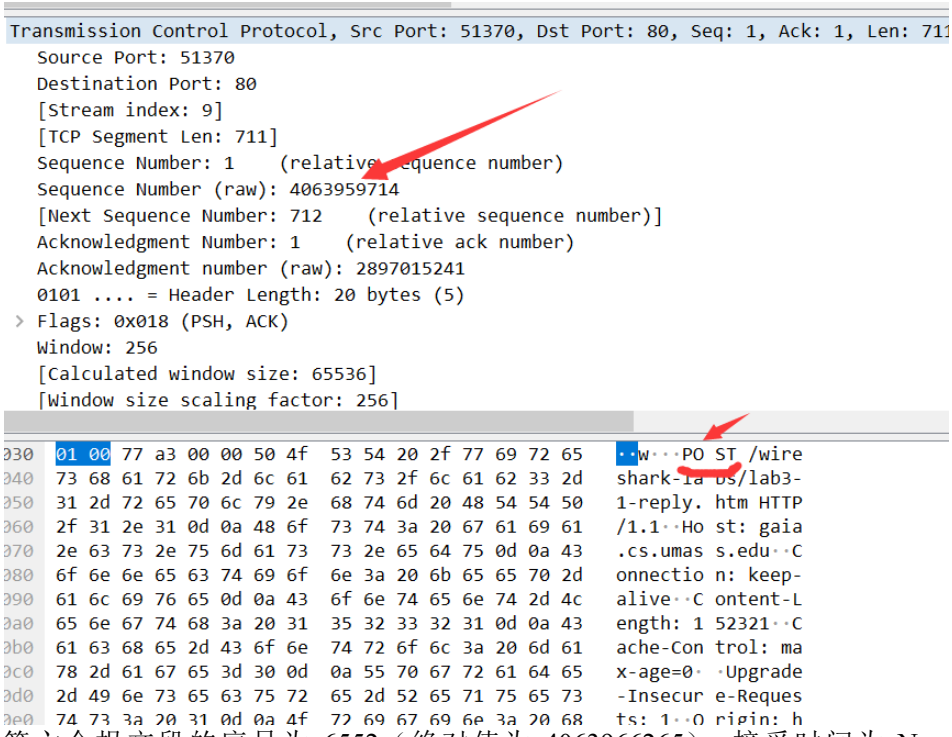
4. 服务器向客户端发送的 SYNACK 报文段序号是 1（绝对值为 3068109256）；Acknowledgement 字段值是 1（绝对值为 3578490484）。服务器将随机指定一个值以决定此值。在该报文段中，在INFO中写入 [SYN,ACK] 来标示的。

```
Sequence Number: 1    (relative sequence number)
Sequence Number (raw): 3068109256
[Next Sequence Number: 1    (relative sequence number)]
Acknowledgment Number: 34    (relative ack number)
Acknowledgment number (raw): 3578490484
```

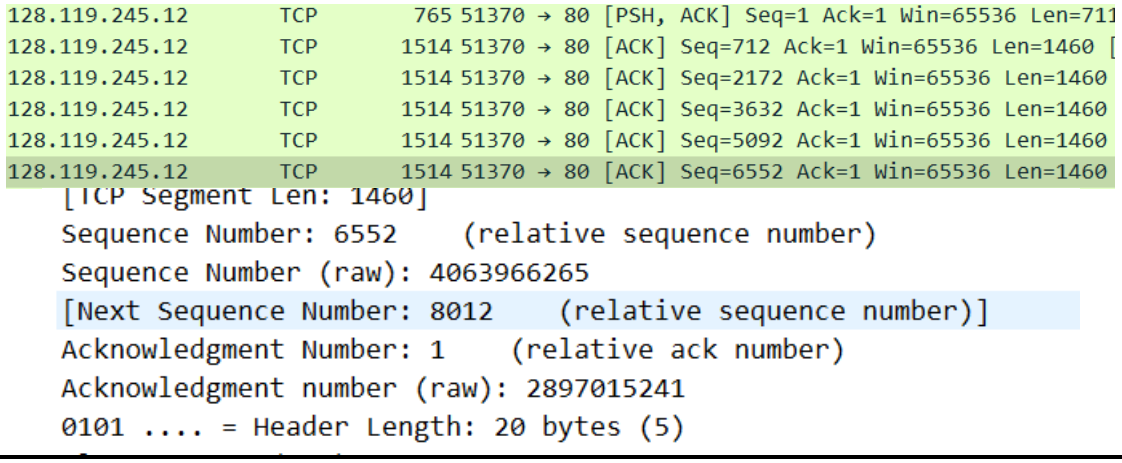
5. 下面的图片展示的就是 TCP 三次握手的过程。首先客户机想服务器端发送 SYN 请求报文；之后服务器向客户机回复 SYN，ACK 报文；最后客户机向服务器回复 ACK 报文段，完成三次握手。



6. 包含 HTTP POST 命令的 TCP 报文的序号是 1（绝对值为 4063959714）。



7. 第六个报文段的序号为 6552（绝对值为 4063966265）。接受时间为 Nov 17, 2021 23:11:41.238356000 中国标准时间。对应的 ACK 接受时间为 Nov 17, 2021 23:11:41.544453000 中国标准时间。



```
Frame 217: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bi
> Interface id: 0 (\Device\NPF_{8200B0C0-46AA-4F67-9681-1714DCC6FC93})
  Encapsulation type: Ethernet (1)
  Arrival Time: Nov 17, 2021 23:11:41.238356000 中国标准时间
  [Time shift for this packet: 0.000000000 seconds]
  Epoch Time: 1637161901.238356000 seconds
  [Time delta from previous captured frame: 0.000000000 seconds]
  [Time delta from previous displayed frame: 0.000000000 seconds]
  [Time since reference or first frame: 4.187142000 seconds]
  Frame Number: 217
```

```
.245.12       TCP      1514 51370 → 80 [ACK] Seq=22612 Ack=1 Win=65536 Len=1460 [TCP se
.163.51       TCP        56 80 → 51370 [ACK] Seq=1 Ack=6552 Win=42368 Len=0
```

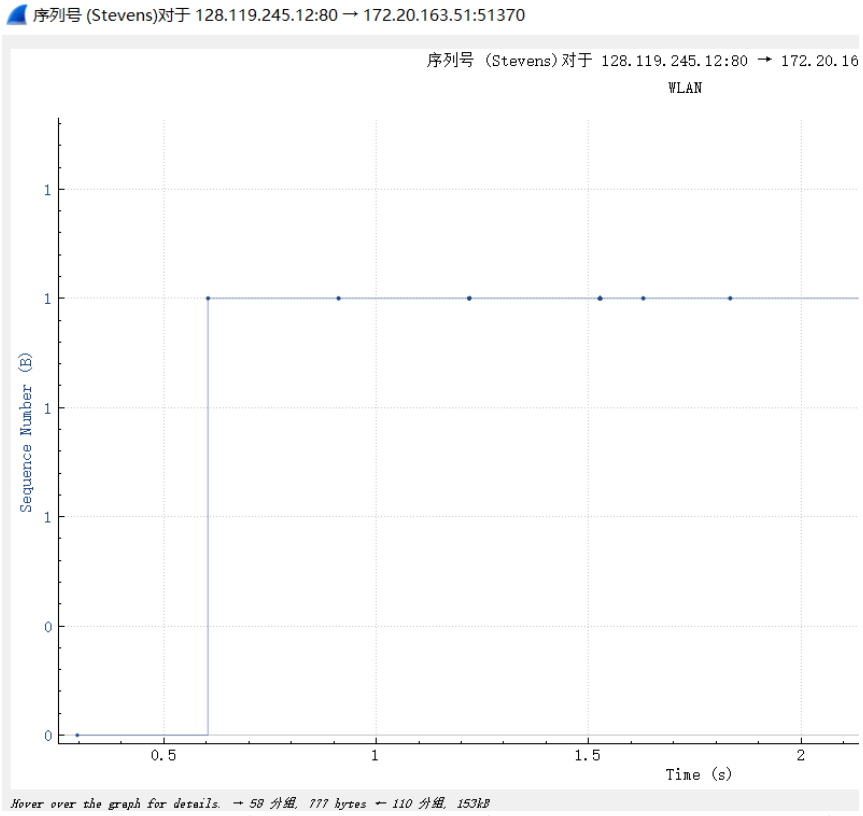
```
Encapsulation type: Ethernet (1)
Arrival Time: Nov 17, 2021 23:11:41.544453000 中国标准时间
[Time shift for this packet: 0.000000000 seconds]
```

8. 前六个 TCP 报文段的长度各是 711、1460、1460、1460、1460 和 1460。

```
765 51370 → 80 [PSH, ACK] Seq=1 Ack=1 Win=65536 Len=711 [TCP segment of a reassembled PDU]
1514 51370 → 80 [ACK] Seq=712 Ack=1 Win=65536 Len=1460 [TCP segment of a reassembled PDU]
1514 51370 → 80 [ACK] Seq=2172 Ack=1 Win=65536 Len=1460 [TCP segment of a reassembled PDU]
1514 51370 → 80 [ACK] Seq=3632 Ack=1 Win=65536 Len=1460 [TCP segment of a reassembled PDU]
1514 51370 → 80 [ACK] Seq=5092 Ack=1 Win=65536 Len=1460 [TCP segment of a reassembled PDU]
1514 51370 → 80 [ACK] Seq=6552 Ack=1 Win=65536 Len=1460 [TCP segment of a reassembled PDU]
```

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 62848，限制发送端的传输以后，接收端的缓存是够用。

10. 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。



11. 最后 ACK 包的序列号为 153033，计时器为 6.378741000 秒，而第一个包的计时器为 3.888682000 秒，所以吞吐量为 $\frac{153033 \times 8bit}{6.378741000s - 3.888682000s} = 0.4917\ Mbps$。

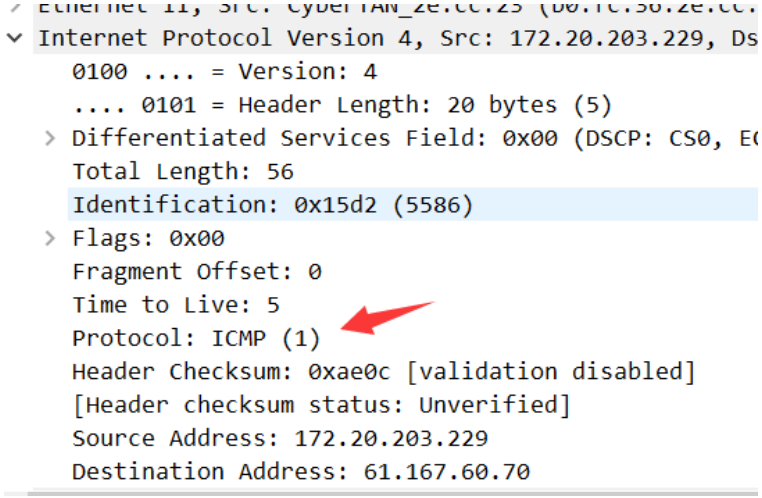
三、IP 分析

运行pingplotter并对hit,edu.cn进行追踪

1. 我的 IP 地址为 172.20.203.229。

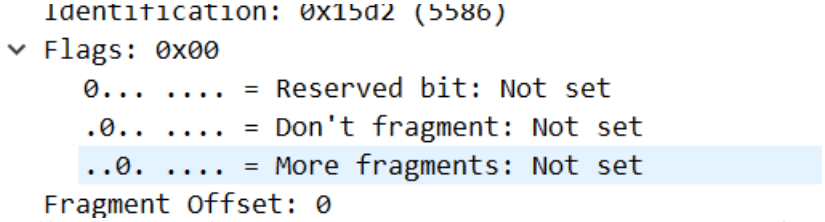
```
0.033096    172.20.203.229    61.167.60.70    ICMP    70 Echo (ping)
```

2. 通过分析 IP 数据包，可以分析出上层协议为 ICMP。

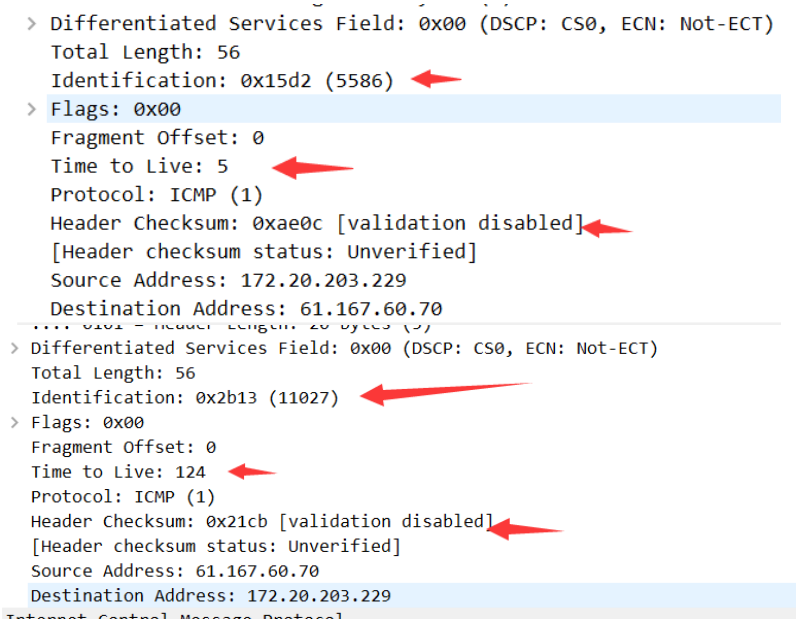


3. 通过上面的 Header Length 行和 Total Length 行可以分析得出 IP 头有 20 字节，该IP数据包的净载为 56-20=36。

4. 观察 Flags 区，可以发现 More fragments 标志为空，没有其余的帧并且帧的偏移为0，可以推断出没有进行分片。



5. 观察 IP 数据包的可以发现 Identification、TTL 和 Checksum 的值总是变化。
其余字段在同一个探测中，是常量
在同一个探测中，identification的值每次-1



6. 必须保持常量的是版本号、首部长度、Differentiated Services Field 以及协议（始终为 ICMP）。必须改变的是 TTL、Checksum 和 Identification，TTL 为生存时间，每次转发必然改变；因为不同地方发出的ICMP数据包对应的IP分组数可能不同（所以Identification字段的值可能不同），发送的ICMP数据包到主机的跳步数可能不同（所以TTL字段的值可能不同）。由于TTL的改变，Checksum 自然也会改变；Identification 则是用于区分不同的 ICMP

报文。

7. Identification 自 5586 开始，接下来的包依次增加 1。

8. Identification 段为 43962，TTL 为 124。

```
Differentiated Services Field: 0x00 (DSCP: CS0, E
Total Length: 56
Identification: 0xabba (43962)
Flags: 0x00
Fragment Offset: 0
Time to Live: 124
```

9. Identification 段发生变化，这样可以区分不同的 ICMP time-to-live exceeded 消息；但 TTL 保持不变，为 124（均为一次转发）。

10. 该消息被分解为不止一个数据包。

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

11. IP 头部可以在 Flags 域中，看到 More fragments 被置为 1 且偏移量为 0，表示该分片不为最后一片。该分片的长度为 1500 字节。

```
Total Length: 1500
Identification: 0x04c5 (1221)
Flags: 0x20, More fragments
    0... .... = Reserved bit: Not set
    .0.. .... = Don't fragment: Not set
    ..1. .... = More fragments: Set
Fragment Offset: 0
```

12. 原始数据被分成了3片。

```
IPv4    1514 Fragmented IP protocol (proto=ICMP 1, off=0, ID=0505) [Reassembled in #709]
IPv4    1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=0505) [Reassembled in #709]
ICMP     554 Echo (ping) request  id=0x0001, seq=86/22016, ttl=1 (no response found!)
```

13. 标志位部分、偏移量和 Checksum 部分发生了变化。

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

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

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

四、抓取 ARP 数据包

1. ARP 表的格式如下。在 ARP 表中，每一项表示一个 IP 地址到物理地址的映射。每一项第一列是IP地址，第二列是物理地址，第三列是类型。

```
接口: 172.20.203.229 --- 0xa
  Internet 地址         物理地址              类型
  172.20.0.1            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
  239.255.255.250       01-00-5e-7f-ff-fa     静态
  255.255.255.255       ff-ff-ff-ff-ff-ff     静态
```

2. ARP数据包的格式如下图所示，共由九部分构成：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）。

3. 可以通过 Opcode 字段判断，若为 1 则是请求包；若为 2 则是应答包。

```
Opcode: request (1)    Opcode: reply (2)
```

4. 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。
2. 本机 IP 地址为 172.20.100.197，目的主机 IP 地址为 111.30.159.62。

```
42 3.113561    172.20.13.125    218.203.59.116    DNS    69 Standard query 0xe166 AAAA wa.qq.com
```

3. 主机发送 QQ 消息的端口号为 56915，QQ 服务器的端口号是 53。

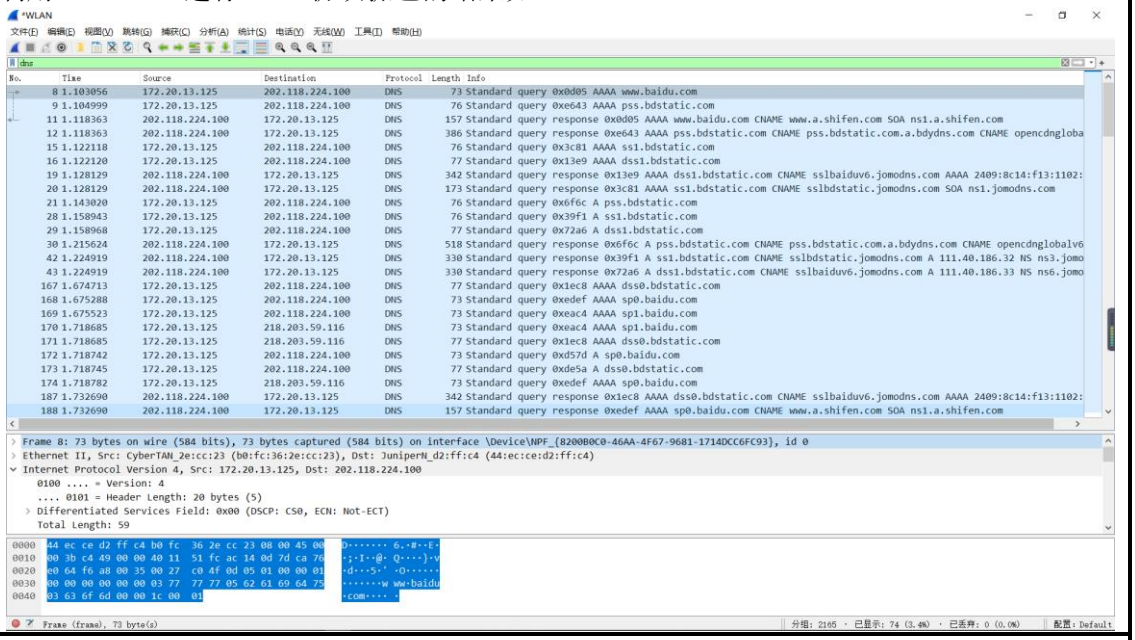
```
User Datagram Protocol, Src Port: 56915, Dst Port: 53
    Source Port: 56915
    Destination Port: 53
    Length: 35
    Checksum: 0x0a2b [unverified]
    [Checksum Status: Unverified]
    [Stream index: 1]
  > [Timestamps]
    UDP payload (27 bytes)
```

4. UDP 数据报由五部分构成，分别是源端口号（4 字节），目的端口号（4 字节），长度（4 字节），校验和（4 字节）和应用层数据。

5. 因为 UDP 是不可靠的数据传输，需要上层协议来实现可靠数据传输，因此每次发送 ICQ 报文后又回复一个 ICQ 数据包来确认。UDP 是无连接的，因为可以看到发送数据之前没有连接的建立过程（如 TCP 的三次握手），没有序列号，因此为无连接数据传输。

六、利用 Wireshark 进行 DNS 协议分析

利用 Wireshark 进行 DNS 协议抓包的结果如上。



问题讨论：

1. 在进行ICMP报文抓取时，win10系统访问的网站都使1pv6的，导致ICMP也是ICMPv6，与实验要求的各字段对不上

在使用Wireshark抓包时，可以通过IP协议看到对应的IP协议版本，发现TCP部分的实验中，网站为IPv4版本，就使用该网站作为目标网站

2. 找不到IF-Modified 浏览器内核改成ie即可解决
心得体会：
通过本次实验，我熟悉并掌握 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。在对于不同的协议报文分析的过程中，我对于各个网络协议的理解更为深入，对于部分细节也有了全新的认识。