

浙江大学实验报告

专业: 计算机科学与技术
姓名: 吴同
学号: 3170104848
日期: 2019年9月29日
地点: 曹西304

课程名称: 计算机网络 指导老师: 张泉方 电子邮件: wutongcs@zju.edu.cn
实验名称: 网络协议分析 实验类型: 综合型 同组同学: 无

一、实验目的

- 学习使用 Wireshark 抓包工具
- 观察和理解常见网络协议的交互过程
- 理解数据包分层结构和格式

二、实验内容

- 掌握网络协议分析软件 Wireshark 的使用，学会配置过滤器
- 观察所在网络出现的各类网络协议，了解其种类和分层结构
- 观察捕获到的数据包格式，理解各字段含义
- 根据要求配置 Wireshark，捕获某一类协议的数据包，并分析解读

三、主要仪器设备

- 硬件设备: MacBook Pro (13-inch, 2017, Four Thunderbolt 3 Ports)
- 操作系统: macOS Mojave 10.14.5
- 软件: Wireshark 3.0.3、Google Chrome 76.0.3809.132

四、操作方法与实验步骤

- 安装网络包捕获软件 Wireshark
- 配置网络包捕获软件，捕获所有机器的数据包
- 观察捕获到的数据包，并对照解析结果和原始数据包
- 配置网络包捕获软件，只捕获特定 IP 或特定类型的包
- 抓取以下通信协议数据包，观察通信过程和数据包格式：PING、TRACE ROUTE、NSLOOKUP、HTTP

五、实验数据记录和处理

Part I

- 运行 Wireshark 软件，开始捕获数据包，列出协议名字。

协议名	截图				
	No.	Time	Source	Destination	Protocol
UDP	1	0.000000	192.168.31.229	180.153.91.44	UDP
TCP	5	0.596843	192.168.31.229	155.138.244.176	TCP
SSLv2	48	9.404224	36.155.240.32	192.168.31.229	SSLv2
ARP	103	14.318426	XiaomiEl_96:87:83	Apple_26:23:08	ARP
TLSv1.2	108	15.444714	192.168.31.229	40.119.211.203	TLSv1.2
DNS	204...	847.207101	192.168.31.229	192.168.31.1	DNS
HTTP	200...	3860.944098	192.168.31.229	39.156.66.179	HTTP
OCSP	201...	3865.468851	112.13.107.243	192.168.31.229	OCSP
SSDP	316...	6352.499628	192.168.31.229	239.255.255.250	SSDP
IGMPv2	316...	6358.341629	0.0.0.0	224.0.0.1	IGMPv2
SSHv2	460...	23191.704774	192.168.31.229	101.37.78.58	SSHv2
MDNS	461...	23224.678505	192.168.31.229	224.0.0.251	MDNS

- 分析一个包含 IP 的数据包。

图 1 是一个包含 IP 的数据包，是开始捕获后接收到的第 170 个数据帧。这个数据包有三层，最高层协议是 Transmission Control Protocol(TCP)，从 Ethernet 开始往上，各层协议的名字分别是：Ethernet II、Internet Protocol Version 4 (IPv4)、Transmission Control Protocol (TCP)，分别位于数据链路层、网络层、传输层。

数据包的头 48 位是目标的 Mac 地址 0xEC4118968783 (EC:41:18:96:87:83)，接下来 48 位是源的 Mac 地址 0x8C8590262308 (8C:85:90:26:23:08)。源和目标的 IP 地址写在 IP 头中，分别是 0xC0A81FE5 (192.168.31.229) 和 0x2877D3CB (40.119.211.203)。

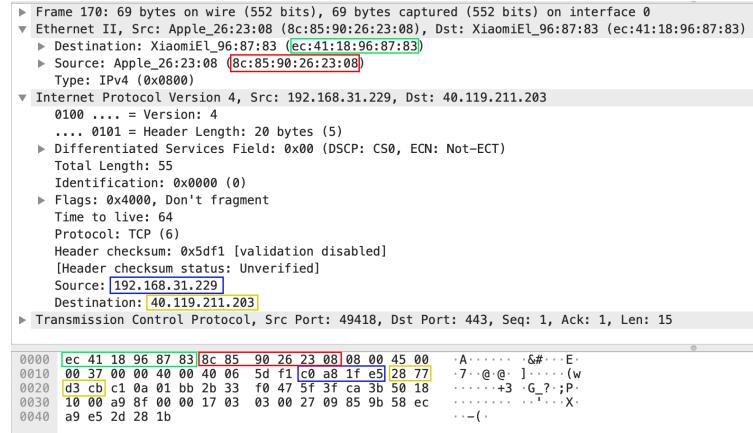


图 1: 标注 IP 层和 Ethernet 层源和目标地址的数据包

3. 配置显示过滤器，让界面只显示某一协议类型的数据包。

使用显示过滤器，过滤规则：tcp，界面只显示 TCP 协议的数据包。

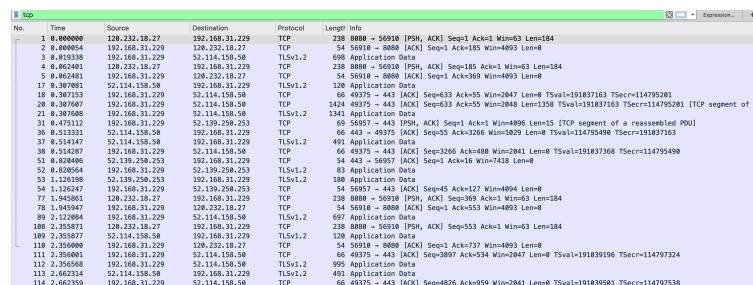


图 2: 显示过滤器 (tcp)

4. 配置显示过滤器，让界面只显示某个 IP 地址的数据包。

使用显示过滤器，过滤规则：ip.addr == 192.168.31.229，只显示 IP 为 192.168.31.229 的数据包。

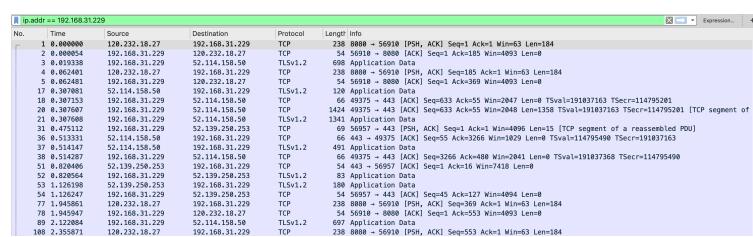


图 3: 显示过滤器 (ip.addr == 192.168.31.229)

5. 配置捕获过滤器，只捕获某个 IP 地址的数据包。

使用捕获过滤器，过滤规则：host 120.232.18.27，只显示 IP 为 120.232.18.27 的数据包。

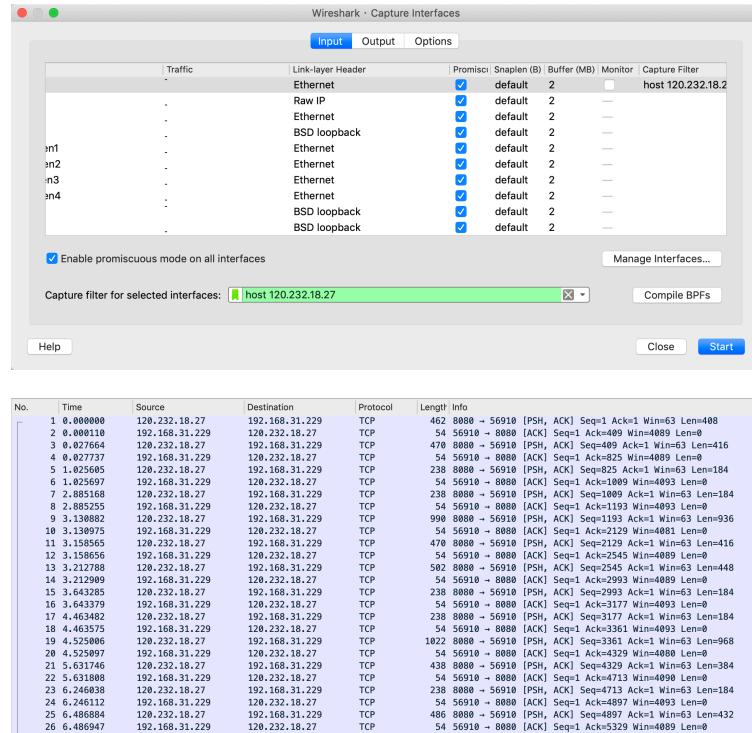


图 4: 配置捕获过滤器 (host 120.232.18.27)

6. 配置捕获过滤器，只捕获某类协议的数据包。

使用捕获过滤器，过滤规则：tcp port 443，只显示 443 端口的 tcp 协议的数据包。

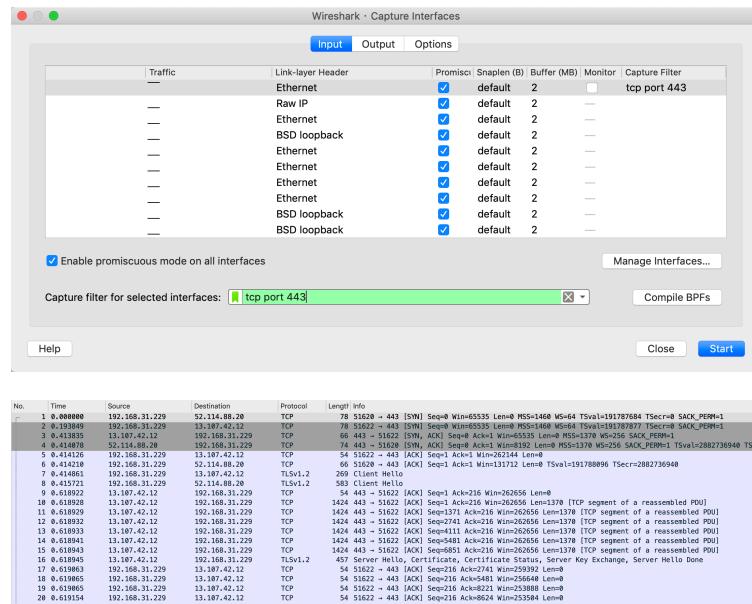


图 5: 配置捕获过滤器 (tcp port 443)

Part II**1. nslookup**

在终端中输入以下命令，抓取的数据包保存在 dns.pcap 文件中。

```
$ nslookup baidu.com
```

```
[→ ~ nslookup baidu.com
Server:      192.168.31.1
Address:     192.168.31.1#53

Non-authoritative answer:
Name:    baidu.com
Address: 220.181.38.148
Name:    baidu.com
Address: 39.156.69.79
```

图 6: nslookup 命令

查看捕获到两个数据包（图 7）。DNS 数据包由四层协议构成，分别是 Ethernet II、Internet Protocol Version 4 (IPv4)、User Datagram Protocol (UDP)、Domain Name System (DNS)。DNS 服务器端口为 53。

No.	Time	Source	Destination	Protocol
1	0.000000	192.168.31.229	192.168.31.1	DNS
2	0.003145	192.168.31.1	192.168.31.229	DNS

图 7: 捕获到两个数据包

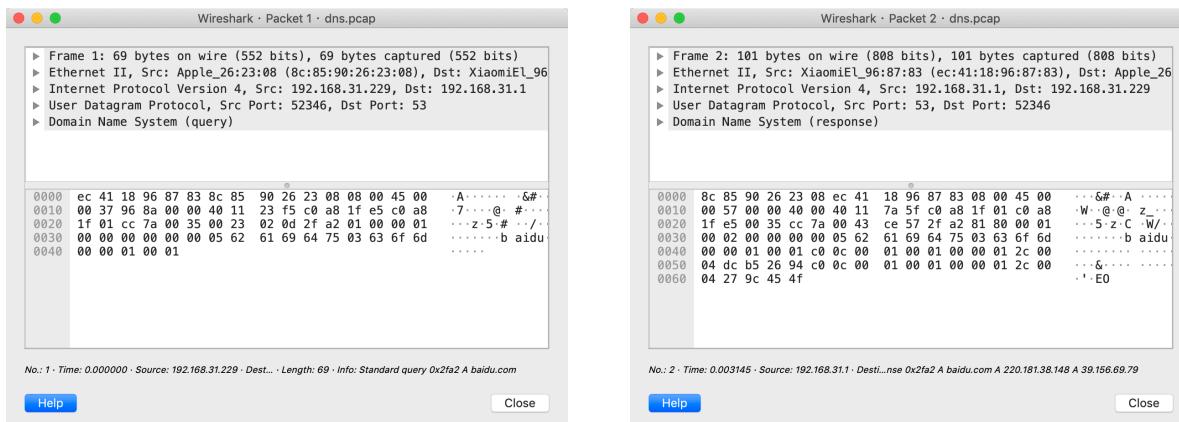


图 8: DNS 数据包详细内容

分析请求包，交易 ID 为 0x2FA2，查询类型为 A(Host Address) (1)，查询的域名内容为 baidu.com。

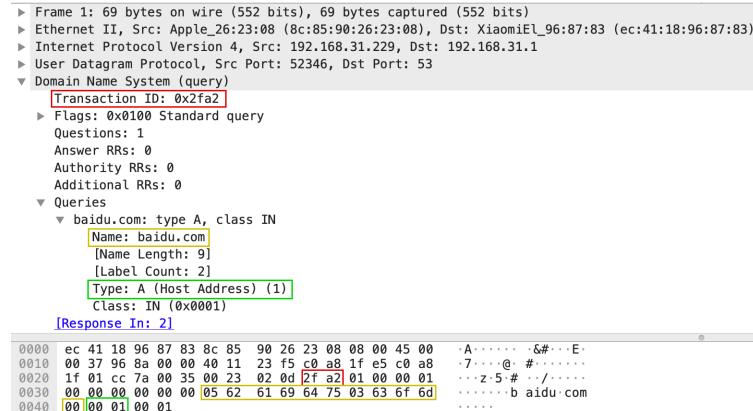


图 9: 请求包分析

分析响应包, 交易 ID 为 0x2FA2, 查询类型为 A(Host Address) (1), 查询的域名内容为 baidu.com, 查
询结果为 baidu.com: type A, class IN, addr 39.156.69.79 baidu.com: type A, class IN, addr 220.181.38.148。

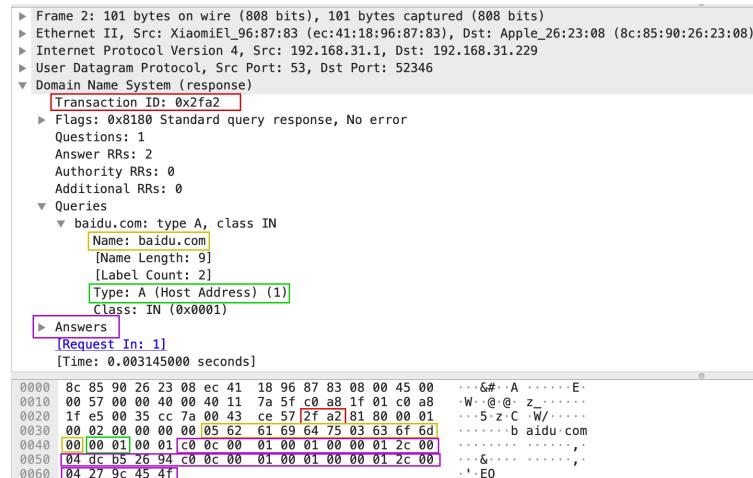


图 10: 响应包分析

2. ping

在终端中输入以下命令, 收发三次数据包后按 **ctrl+C** 停止, 图 12 为捕获的数据包, 保存在 **ping_ip.pcap** 文件中。抓取到 ARP 协议和 ICMP 协议的数据包。

```
$ ping 192.168.31.5
```

```
[~] ~ ping 192.168.31.5
PING 192.168.31.5 (192.168.31.5): 56 data bytes
64 bytes from 192.168.31.5: icmp_seq=0 ttl=64 time=3.770 ms
64 bytes from 192.168.31.5: icmp_seq=1 ttl=64 time=3.451 ms
64 bytes from 192.168.31.5: icmp_seq=2 ttl=64 time=3.021 ms
^C
--- 192.168.31.5 ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 3.021/3.414/3.770/0.307 ms
```

图 11: ping 192.168.31.5 命令

23 0.364411	Apple_26:23:08		Broadcast	ARP	42 Who has 192.168.31.5? T=1	192.168.31.229
24 0.367908	Apple_26:23:08		Apple_26:23:08	ARP	68 192.168.31.5 is at 0c:2e:7f:11:1a	
25 0.368001	Apple_26:23:08		192.168.31.229	ICMP	98 Echo (ping) request id=0x2440, seq=0/0, ttl=64 (request in 25)	
26 0.368081	192.168.31.5		192.168.31.229	ICMP	98 Echo (ping) reply id=0x2440, seq=0/0, ttl=64 (reply in 26)	
27 0.408008	52.169.56.33		192.168.31.229	TCP	66 445 - 57289 [ACK] Seq=765 Ack=33 Win=1024 Len=0 Tsvl=1550121624 Tscr=41792484	
28 0.408011	52.169.56.33		192.168.31.229	TCP	66 445 - 57289 [ACK] Seq=766 Win=2048 Len=0 Tsvl=1417575231 Tscr=41792484	
29 0.423108	52.169.56.33		192.168.31.229	TCP	66 57289 - 443 [ACK] Seq=33 Ack=766 Win=2048 Len=0 Tsvl=1417575231 Tscr=41581281	
30 0.423105	52.169.56.33		192.168.31.229	TCP	66 445 - 57288 [ACK] Seq=962 Ack=33 Win=1024 Len=0 Tsvl=1359813769 Tscr=415747836	
31 0.423108	52.169.56.33		192.168.31.229	TCP	66 57289 - 443 [ACK] Seq=33 Ack=766 Win=2048 Len=0 Tsvl=1417575246 Tscr=41581281	
32 0.423232	192.168.31.229		52.169.56.33	TCP	66 57288 - 443 [ACK] Seq=33 Ack=993 Win=2048 Len=0 Tsvl=1417575246 Tscr=358013769	
33 1.368568	192.168.31.229		192.168.31.229	ICMP	98 Echo (ping) request id=0x2440, seq=0/0, ttl=64 (request in 33)	
34 1.368571	192.168.31.229		192.168.31.229	ICMP	98 Echo (ping) reply id=0x2440, seq=0/0, ttl=64 (reply in 34)	
35 2.372623	192.168.31.229		192.168.31.5	ICMP	98 Echo (ping) request id=0x2440, seq=0/312, ttl=64 (request in 35)	
36 2.375528	192.168.31.5		192.168.31.229	ICMP	98 Echo (ping) reply id=0x2440, seq=0/512, ttl=64 (request in 35)	

图 12: ping 192.168.31.5 抓取的数据包

在终端中输入以下命令, 收发三次数据包后按 ctrl+C 停止, 图 14 为捕获的数据包, 保存在 ping_name.pcap 文件中。抓取到 DNS 协议和 ICMP 协议的数据包。

```
$ ping www.zju.edu.cn
```

```
[~] ~ ping www.zju.edu.cn
PING www.zju.edu.cn (10.203.6.101): 56 data bytes
64 bytes from 10.203.6.101: icmp_seq=0 ttl=59 time=2.357 ms
64 bytes from 10.203.6.101: icmp_seq=1 ttl=59 time=1.776 ms
64 bytes from 10.203.6.101: icmp_seq=2 ttl=59 time=3.155 ms
^C
--- www.zju.edu.cn ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 1.776/2.429/3.155/0.565 ms
```

图 13: ping www.zju.edu.cn 命令

34 0.594548	192.168.31.229	192.168.31.1	DNS	74 Standard query response A www.zju.edu.cn	
35 0.595932	192.168.31.1	192.168.31.229	DNS	98 Standard query response A www.zju.edu.cn A 10.203.6.101	
36 0.596292	192.168.31.229	16.283.6.191	ICMP	98 Echo (ping) request id=0x6550, seq=0/0, ttl=64 (reply in 37)	
37 0.596305	16.283.6.191	192.168.31.229	ICMP	98 Echo (ping) reply id=0x6550, seq=0/0, ttl=64 (request in 37)	
38 1.824818	52.169.56.33	192.168.31.229	TCP	66 443 - 57908 [ACK] Seq=10888 Ack=1876 Win=3244 Len=0 Tsvl=1508446968 Tscr=417796638	
39 1.824817	52.169.56.33	192.168.31.229	TCP	66 443 - 57908 [ACK] Seq=10888 Ack=1876 Win=3244 Len=0 Tsvl=1508446968 Tscr=417796638	
40 1.824817	52.169.56.33	192.168.31.229	TCP	66 443 - 57908 [ACK] Seq=10888 Ack=1876 Win=3244 Len=0 Tsvl=1508446968 Tscr=417796638	
41 1.824817	121.51.36.141	192.168.31.229	TCP	66 443 - 57908 [ACK] Seq=10888 Ack=1876 Win=3244 Len=0 Tsvl=1508446968 Tscr=417796638	
42 1.835588	192.168.31.229	121.51.36.141	TCP	54 57398 - 88 [ACK] Seq=1 Ack=4561 Min=4993 Len=0	
43 1.599743	192.168.31.229	16.283.6.191	ICMP	98 Echo (ping) request id=0x6550, seq=1/256, ttl=64 (reply in 44)	
44 1.599743	16.283.6.191	192.168.31.229	ICMP	98 Echo (ping) reply id=0x6550, seq=1/256, ttl=64 (request in 43)	
45 2.680715	192.168.31.229	16.283.6.191	ICMP	98 Echo (ping) request id=0x6550, seq=2/512, ttl=64 (reply in 46)	
46 2.683761	18.203.6.181	192.168.31.229	ICMP	98 Echo (ping) reply id=0x6550, seq=2/512, ttl=59 (request in 45)	

图 14: ping www.zju.edu.cn 抓取的数据包

如图 15 所示, 一个 ICMP 数据包包含三层协议, 分别是 Ethernet II、Internet Protocol Version 4 (IPv4)、Internet Control Message Protocol (ICMP)。

36 0.596292	192.168.31.229	10.203.6.101	ICMP	98 Echo (ping) request id=0x6550, seq=0/0, ttl=64 (reply in 37)	
> Ethernet II, Src: Apple_26:23:08 (0c:2e:7f:11:1a:08), Dst: Xiaomi_E1 (96:87:83:ec:41:18)					
> Internet Protocol Version 4, Src: 192.168.31.229, Dst: 10.203.6.101					
> Internet Control Message Protocol					

图 15: ICMP 数据包的三层

分析 ARP 请求包，操作码为 1，发送者 IP 地址为 192.168.31.229，发送着 MAC 地址为 8C:85:90:26:23:08，查询的目标 IP 地址为 192.168.31.5，Ethernet 层的目标 MAC 地址为 FF:FF:FF:FF:FF:FF，查询结果为 00:00:00:00:00:00。

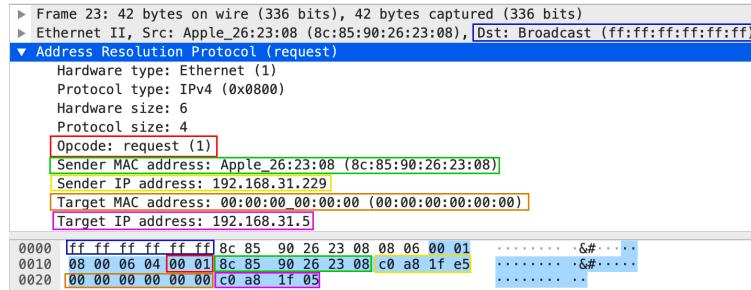


图 16: ARP 请求包

分析 ARP 响应包，操作码为 2，发送者 IP 地址为 192.168.31.5，发送着 MAC 地址为 B8:27:EB:82:1F:A0，查询的目标 IP 地址为 192.168.31.229，Ethernet 层的目标 MAC 地址为 8C:85:90:26:23:08，查询结果为 8C:85:90:26:23:08。

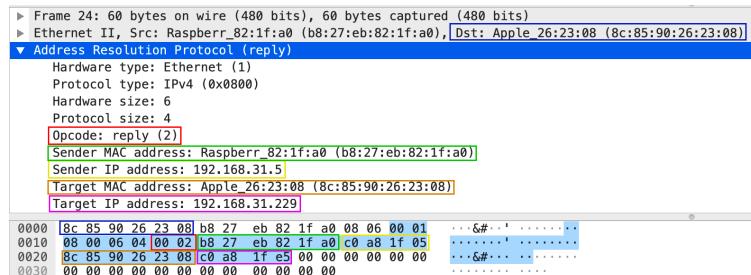


图 17: ARP 响应包

分析 ICMP 请求包，类型为 Type: 8 (Echo (ping) request)，序号为 0。

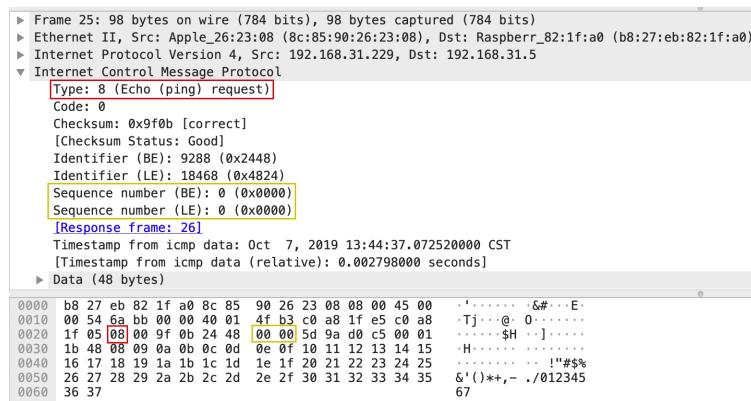


图 18: ICMP 请求包

分析 ICMP 响应包，类型为 Type: 0 (Echo (ping) reply)，序号为 0。

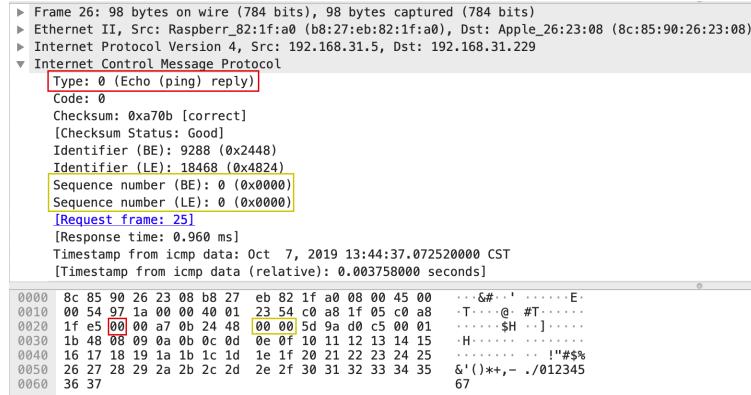


图 19: ICMP 响应包

3. traceroute

在终端中输入以下命令，抓取的数据包保存在 traceroute.pcap 文件中。

```
$ traceroute 155.138.244.176
```

```
[~] ~ traceroute 155.138.244.176
traceroute to 155.138.244.176 (155.138.244.176), 64 hops max, 52 byte packets
 1 bogon (172.20.10.1)  6.132 ms  5.056 ms  2.218 ms
 2 * * *
 3 bogon (10.245.101.110)  96.541 ms  98.089 ms  101.489 ms
 4 * * *
 5 113.5.0.141 (113.5.0.141)  108.238 ms  97.469 ms  100.877 ms
 6 113.5.0.133 (113.5.0.133)  186.877 ms  91.861 ms  98.830 ms
 7 1.58.81.105 (1.58.81.105)  450.625 ms
 1.58.81.29 (1.58.81.29)  301.579 ms
 1.62.62.41 (1.62.62.41)  113.481 ms
 8 219.158.105.213 (219.158.105.213)  123.670 ms  162.458 ms  313.149 ms
 9 219.158.5.158 (219.158.5.158)  314.534 ms  127.730 ms  169.883 ms
 10 219.158.3.142 (219.158.3.142)  610.233 ms  122.809 ms  187.130 ms
 11 219.158.98.18 (219.158.98.18)  303.513 ms  306.879 ms  306.970 ms
 12 219.158.34.246 (219.158.34.246)  307.318 ms  302.038 ms  306.172 ms
 13 dls-b22-link.telia.net (62.115.118.246)  614.337 ms  921.265 ms *
 14 vultr-ic-318447-dls-b22.c.telia.net (62.115.149.45)  318.588 ms *  430.693 ms
 15 * * *
 16 * * *
 17 155.138.244.176.vultr.com (155.138.244.176)  413.483 ms  604.136 ms  608.550 ms
```

图 20: traceroute 155.138.244.176 命令

traceroute 使用的数据包协议类型是 UDP 和 ICMP。数据包由三层协议构成，分别是 Ethernet II、Internet Protocol Version 4 (IPv4)、User Datagram Protocol (UDP) /Internet Control Message Protocol (ICMP)。

配置显示过滤器规则为 `udp and not dns and not mdns and not icmp`，查看捕获的 traceroute 请求包。观察请求包中 IP 协议层的 TTL 字段，第一个请求的 TTL 等于 1，同样的 TTL 请求连续发送了 3 个，然后每次 TTL 增加 1，最后一个请求的 TTL 等于 17。

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.20.10.2	155.138.244.176	UDP	66	449880 → 33435 Len=24
3	0.007158	172.20.10.2	155.138.244.176	UDP	66	449880 → 33436 Len=24
5	0.012161	172.20.10.2	155.138.244.176	UDP	66	449880 → 33437 Len=24
7	0.014390	172.20.10.2	155.138.244.176	UDP	66	449880 → 33438 Len=24
32	5.015733	172.20.10.2	155.138.244.176	UDP	66	449880 → 33439 Len=24
82	10.020913	172.20.10.2	155.138.244.176	UDP	66	449880 → 33440 Len=24
85	15.020978	172.20.10.2	155.138.244.176	UDP	66	449880 → 33441 Len=24
87	15.110822	172.20.10.2	155.138.244.176	UDP	66	449880 → 33442 Len=24
89	15.216148	172.20.10.2	155.138.244.176	UDP	66	449880 → 33443 Len=24
91	15.317634	172.20.10.2	155.138.244.176	UDP	66	449880 → 33444 Len=24
138	20.321119	172.20.10.2	155.138.244.176	UDP	66	449880 → 33445 Len=24
194	25.324680	172.20.10.2	155.138.244.176	UDP	66	449880 → 33446 Len=24
235	30.327889	172.20.10.2	155.138.244.176	UDP	66	449880 → 33447 Len=24
237	30.436764	172.20.10.2	155.138.244.176	UDP	66	449880 → 33448 Len=24
239	30.534257	172.20.10.2	155.138.244.176	UDP	66	449880 → 33449 Len=24
241	30.635153	172.20.10.2	155.138.244.176	UDP	66	449880 → 33450 Len=24
243	30.822503	172.20.10.2	155.138.244.176	UDP	66	449880 → 33451 Len=24
245	30.914382	172.20.10.2	155.138.244.176	UDP	66	449880 → 33452 Len=24
247	31.013241	172.20.10.2	155.138.244.176	UDP	66	449880 → 33453 Len=24
251	31.762253	172.20.10.2	155.138.244.176	UDP	66	449880 → 33454 Len=24
253	32.064424	172.20.10.2	155.138.244.176	UDP	66	449880 → 33455 Len=24
382	50.196134	172.20.10.2	155.138.244.176	UDP	66	449880 → 33456 Len=24
409	50.320364	172.20.10.2	155.138.244.176	UDP	66	449880 → 33457 Len=24
411	50.482837	172.20.10.2	155.138.244.176	UDP	66	449880 → 33458 Len=24
413	50.796008	172.20.10.2	155.138.244.176	UDP	66	449880 → 33459 Len=24
415	51.111021	172.20.10.2	155.138.244.176	UDP	66	449880 → 33460 Len=24
417	51.238767	172.20.10.2	155.138.244.176	UDP	66	449880 → 33461 Len=24
419	51.408669	172.20.10.2	155.138.244.176	UDP	66	449880 → 33462 Len=24
421	52.019927	172.20.10.2	155.138.244.176	UDP	66	449880 → 33463 Len=24
423	52.142694	172.20.10.2	155.138.244.176	UDP	66	449880 → 33464 Len=24
427	52.329837	172.20.10.2	155.138.244.176	UDP	66	449880 → 33465 Len=24
429	52.633938	172.20.10.2	155.138.244.176	UDP	66	449880 → 33466 Len=24
431	52.949816	172.20.10.2	155.138.244.176	UDP	66	449880 → 33467 Len=24
433	53.247837	172.20.10.2	155.138.244.176	UDP	66	449880 → 33468 Len=24
437	53.561074	172.20.10.2	155.138.244.176	UDP	66	449880 → 33469 Len=24
439	53.863102	172.20.10.2	155.138.244.176	UDP	66	449880 → 33470 Len=24
441	54.169425	172.20.10.2	155.138.244.176	UDP	66	449880 → 33471 Len=24
443	54.784862	172.20.10.2	155.138.244.176	UDP	66	449880 → 33472 Len=24
445	55.705370	172.20.10.2	155.138.244.176	ICMP	66	449880 → 33473 Len=24

图 21: 捕获的 traceroute 请求包

配置显示过滤器规则为 icmp，查看捕获的 traceroute 相应包。观察相应包的信息，第一个发送者 IP 是 172.20.10.1，ICMP 层的类型为 11 (Time-to-live exceeded)。最后一个发送者 IP 是 155.138.244.176，ICMP 层的类型是 3 (Destination unreachable)。

No.	Time	Source	Destination	Protocol	Length	Info
2	0.005815	172.20.10.1	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
4	0.012069	172.20.10.1	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
6	0.014393	172.20.10.1	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
86	15.177353	10.245.101.110	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
88	15.215991	10.245.101.110	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
96	15.317498	10.245.101.110	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
230	30.435997	113.5.0.141	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
238	30.534139	113.5.0.141	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
300	30.815027	113.5.0.141	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
242	30.823227	113.5.0.133	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
244	30.834280	113.5.0.133	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
246	31.013119	113.5.0.133	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
248	31.463755	113.5.0.133	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
252	32.063692	113.5.0.133	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
254	32.321773	113.5.0.133	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
408	30.319356	219.158.105.213	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
410	58.482707	219.158.105.213	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
412	58.795887	219.158.105.213	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
414	51.118422	219.158.5.158	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
416	51.238653	219.158.5.158	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
418	51.408561	219.158.5.158	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
420	51.473213	219.158.5.158	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
422	52.147562	219.158.3.142	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
425	52.329694	219.158.3.142	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
428	52.633244	219.158.98.18	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
430	52.948722	219.158.98.18	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
432	53.147637	219.158.98.18	172.20.10.2	ICMP	118	118 Time-to-live exceeded (Time to live exceeded in transit)
434	53.154255	219.158.34.246	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
436	53.862964	219.158.34.246	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
440	54.169178	219.158.34.246	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
442	54.783582	62.115.118.246	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
444	55.705245	62.115.118.246	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
469	61.023963	62.115.140.246	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
500	61.023963	62.115.149.45	172.20.10.2	ICMP	70	70 Time-to-live exceeded (Time to live exceeded in transit)
656	96.885824	155.138.244.176	172.20.10.2	ICMP	94	94 Destination unreachable (Port unreachable)
659	97.489917	155.138.244.176	172.20.10.2	ICMP	94	94 Destination unreachable (Port unreachable)
661	98.098501	155.138.244.176	172.20.10.2	ICMP	94	94 Destination unreachable (Port unreachable)

图 22: 捕获的 traceroute 响应包

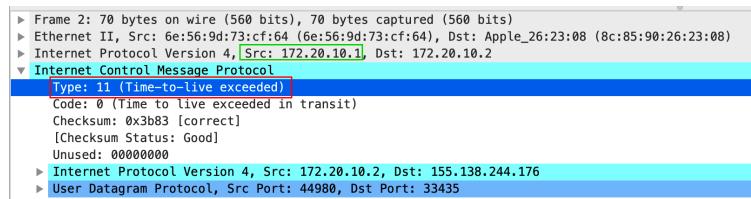


图 23: 第一组响应包

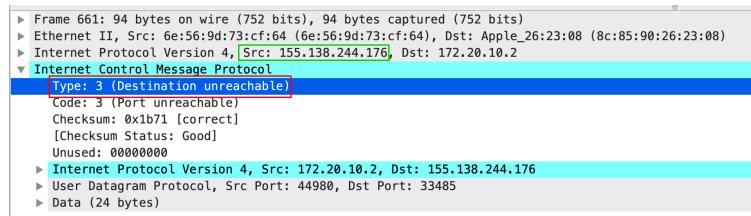


图 24: 最后一组响应包

Part III

1. 捕获访问网站数据

在终端中输入以下命令，清空 DNS 缓存。设置捕获过滤器规则为 tcp port 80 or udp port 53，在浏览器中打开 www.zju.edu.cn 网页（图 25），抓取的数据包保存在 http.pcap 文件中。捕获到的 DNS 数据包和 HTTP 数据包都由四层协议构成，DNS 数据包的四层分别是 Ethernet II、Internet Protocol Version 4 (IPv4)、User Datagram Protocol (UDP)、Domain Name System (DNS)，HTTP 数据包的四层分别是 Ethernet II、Internet Protocol Version 4 (IPv4)、Transmission Control Protocol (TCP)、Hypertext Transfer Protocol (HTTP)。

```
$ sudo dscacheutil -flushcache
```

图 25: www.zju.edu.cn 网页

DNS 协议数据包的源 IP 地址为 192.168.31.229，目标 IP 地址为 192.168.31.1，源端口为 64965，目标端口为 53。

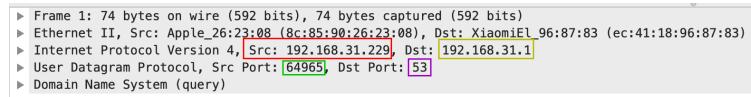


图 26: DNS 协议数据包

HTTP 协议数据包的源 IP 地址为 192.168.31.229，目标 IP 地址为 10.203.6.101，源端口为 64769，目标端口为 80。

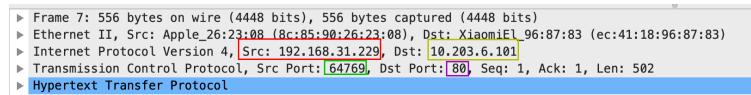


图 27: HTTP 协议数据包

2. 域名列表

设置显示过滤器规则为 dns，为了打开网页，浏览器查询了这些域名：

- www.zju.edu.cn
- hm.baidu.com
- ac.zju.edu.cn
- bksy.zju.edu.cn
- courses.zju.edu.cn
- culture.zju.edu.cn
- grs.zju.edu.cn
- helps.zju.edu.cn
- iczu.zju.edu.cn
- libweb.zju.edu.cn
- map.zju.edu.cn
- mp.weixin.qq.com
- my.zju.edu.cn
- ocw.zju.edu.cn
- person.zju.edu.cn
- piclib.zju.edu.cn

- rd.zju.edu.cn
- rwsk.zju.edu.cn
- ugrs.zju.edu.cn
- weibo.com
- www.acv.zju.edu.cn
- www.beian.gov.cn
- www.career.zju.edu.cn
- www.ce.zju.edu.cn
- www.innovation2030.zju.edu.cn
- www.itri.zju.edu.cn
- www.journals.zju.edu.cn
- www.news.zju.edu.cn
- www.qsc.zju.edu.cn
- www.toutiao.com
- www.tyys.zju.edu.cn
- www.zdxqn.zju.edu.cn
- www.gstatic.com
- clients1.google.com
- www.zuef.zju.edu.cn
- xncfzyjy.zju.edu.cn
- xwfw.zju.edu.cn
- zdxb.cuepa.cn
- zdzsc.zju.edu.cn
- zuaa.zju.edu.cn
- zuits.zju.edu.cn

3. TCP 连接数

使用显示过滤器 `tcp.stream eq X`, 让 X 从 0 开始变化, X 为 2 时没有数据。浏览器为了获取网页数据, 总共建立了 2 个连接。

The figure consists of three vertically stacked screenshots of the Wireshark interface, each showing a list of network packets. The top screenshot is titled 'tcp.stream eq 0' and shows a large number of TCP packets between two hosts, 192.168.31.229 and 10.203.6.101. The middle screenshot is titled 'tcp.stream eq 1' and shows a smaller set of packets. The bottom screenshot is titled 'tcp.stream eq 2' and shows no data.

tcp.stream eq 0						
No.	Time	Source	Destination	Protocol	Length	Info
3	0.001974	192.168.31.229	10.203.6.101	TCP	78	
5	0.003843	10.203.6.101	192.168.31.229	TCP	66	
6	0.003879	192.168.31.229	10.203.6.101	TCP	54	
7	0.004419	192.168.31.229	10.203.6.101	HTTP	556	
10	0.017958	10.203.6.101	192.168.31.229	TCP	1424	
11	0.017960	10.203.6.101	192.168.31.229	TCP	1424	
12	0.017961	10.203.6.101	192.168.31.229	TCP	1424	
13	0.017962	10.203.6.101	192.168.31.229	TCP	1424	
14	0.018009	192.168.31.229	10.203.6.101	TCP	54	
15	0.018032	192.168.31.229	10.203.6.101	TCP	54	
16	0.018087	10.203.6.101	192.168.31.229	TCP	1424	
17	0.018089	10.203.6.101	192.168.31.229	TCP	1424	
18	0.018096	192.168.31.229	10.203.6.101	TCP	54	
19	0.018107	192.168.31.229	10.203.6.101	TCP	54	
20	0.018917	10.203.6.101	192.168.31.229	TCP	1424	
21	0.018919	10.203.6.101	192.168.31.229	TCP	1424	
22	0.018920	10.203.6.101	192.168.31.229	TCP	1424	
23	0.018920	10.203.6.101	192.168.31.229	HTTP	1124	
24	0.018941	192.168.31.229	10.203.6.101	TCP	54	
25	0.018968	192.168.31.229	10.203.6.101	TCP	54	
26	0.019295	192.168.31.229	10.203.6.101	TCP	54	
107	0.798366	192.168.31.229	10.203.6.101	HTTP	582	
108	0.805366	10.203.6.101	192.168.31.229	HTTP	266	
109	0.805435	192.168.31.229	10.203.6.101	TCP	54	

tcp.stream eq 1						
No.	Time	Source	Destination	Protocol	Length	Info
4	0.003743	192.168.31.229	10.203.6.101	TCP	78	
8	0.005452	10.203.6.101	192.168.31.229	TCP	66	
9	0.005495	192.168.31.229	10.203.6.101	TCP	54	

tcp.stream eq 2						
No.	Time	Source	Destination	Protocol	Length	Info

图 28: 显示过滤器 `tcp.stream eq X`

4. HTTP 会话数

使用显示过滤器 `http`, 有两对 HTTP 请求和响应, 浏览器与 WEB 服务器之间进行了 2 次 HTTP 会话。

The figure shows a screenshot of the Wireshark interface with a list of HTTP packets. It displays two distinct sessions: one from 192.168.31.229 to 10.203.6.101 and another from 10.203.6.101 back to 192.168.31.229. The last few rows show the final responses of these sessions.

No.	Time	Source	Destination	Protocol	Length	Info
7	0.004419	192.168.31.229	10.203.6.101	HTTP	556	GET / HTTP/1.1
23	0.018920	10.203.6.101	192.168.31.229	HTTP	1124	HTTP/1.1 200 OK (text/html)
107	0.798366	192.168.31.229	10.203.6.101	HTTP	582	GET /visitcount?siteId=3&type=1&columnId=5 HTTP/1.1
108	0.805366	10.203.6.101	192.168.31.229	HTTP	266	HTTP/1.1 200 OK

图 29: 显示过滤器 `http`

5. 跟踪 TCP 流

选择有两个 HTTP 会话的 TCP 流跟踪。

```

> Frame 7: 556 bytes on wire (4448 bits), 556 bytes captured (4448 bits)
> Ethernet II, Src: Apple_26:23:08 (0c:85:90:26:23:08), Dst: XiaomiEL_96:87:83 (ec:41:18:96:87:83)
> Internet Protocol Version 4, Src: 192.168.31.229, Dst: 10.203.6.101
> Transmission Control Protocol, Src Port: 64769, Dst Port: 80, Seq: 1, Ack: 1, Len: 502
▼ Hypertext Transfer Protocol
  ▶ GET / HTTP/1.1\r\n
    > [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
      Host: www.zju.edu.cn\r\n
      Connection: keep-alive\r\n
      Upgrade-Insecure-Requests: 1\r\n
      User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.90 Safari/537.36\r\n
      Accept: */*\r\n
      Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7\r\n
      Cookie: Hm_lvt_fe30bbc1ee45421ec1679d1b8df8453=1570446377;r=\n
  \r\n
  [Full request URL: http://www.zju.edu.cn/]
  [HTTP request 1/2]
  [Response in frame: 23]
  [Next request in frame: 107]

```

图 30: 第一个会话的请求部分

```

> Frame 23: 1124 bytes on wire (8992 bits), 1124 bytes captured (8992 bits)
> Ethernet II, Src: XiaomiEL_96:87:83 (ec:41:18:96:87:83), Dst: Apple_26:23:08 (0c:85:90:26:23:08)
> Internet Protocol Version 4, Src: 10.203.6.101, Dst: 192.168.31.229
> Transmission Control Protocol, Src Port: 64769, Dst Port: 80, Seq: 12331, Ack: 503, Len: 1070
> IP (version=4, headerlen=20 bytes, offset=0)
  ▶ IP (version=4, headerlen=20 bytes, offset=0)
    ▶ TCP Segments (1340 bytes): #10(1370), #11(1370), #13(1370), #16(1370), #17(1370), #20(1370), #21(1370), #22(1370), #23(1070)
    ▶ Hypertext Transfer Protocol
      ▶ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
        Response Version: HTTP/1.1
        Status Code: 200
        [Status Code Description: OK]
        Response Phrase: OK
        Date: Mon, 07 Oct 2019 11:09:03 GMT\r\n
        Server: Apache/2.4.33 (Ubuntu)\r\n
        X-Frame-Options: SAMEORIGIN\r\n
        X-Content-Type-Options: nosniff\r\n
        Content-Security-Policy: default-src 'self'; script-src 'self' 'unsafe-eval' 'unsafe-inline'; style-src 'self' 'unsafe-style-src'; font-src 'self'; object-src 'self'; img-src 'self' data:; frame-src 'self'; media-src 'self'; child-src 'self'; frame-ancestors 'self'; script-src 'self' 'unsafe-eval' 'unsafe-inline'; style-src 'self' 'unsafe-style-src'; font-src 'self'; object-src 'self'; img-src 'self' data:; frame-src 'self'; media-src 'self'; child-src 'self'; frame-ancestors 'self';\r\n
        Accept-Ranges: bytes\r\n
        Vary: Accept-Encoding\r\n
        Content-Encoding: gzip\r\n
        Content-Length: 13155\r\n
        Content-Type: text/html\r\n
  \r\n
  [HTTP response 1/2]
  [Request since previous: 0.014501000 seconds]
  [Request in frame: 21]
  [Next request in frame: 107]
  [Next response in frame: 108]
  [Request URL: http://www.zju.edu.cn/]
  Content mode: entity body (gzip); 13155 bytes -> 56444 bytes
  File Data: 56444 bytes
  Line-based text data: text/html (1145 lines)

```

图 31: 第一个会话的响应部分

```

> Frame 107: 582 bytes on wire (4656 bits), 582 bytes captured (4656 bits)
> Ethernet II, Src: Apple_26:23:08 (0c:85:90:26:23:08), Dst: XiaomiEL_96:87:83 (ec:41:18:96:87:83)
> Internet Protocol Version 4, Src: 192.168.31.229, Dst: 10.203.6.101
> Transmission Control Protocol, Src Port: 64769, Dst Port: 80, Seq: 503, Ack: 13401, Len: 528
▼ Hypertext Transfer Protocol
  ▶ GET /visitcount?siteId=3&type=1&columnId=5 HTTP/1.1\r\n
    > [Expert Info (Chat/Sequence): GET /visitcount?siteId=3&type=1&columnId=5 HTTP/1.1\r\n]
      Request Method: GET
      Request URI: /visitcount?siteId=3&type=1&columnId=5
      Request Version: HTTP/1.1
      Host: www.zju.edu.cn\r\n
      Connection: keep-alive\r\n
      User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/77.0.3865.90 Safari/537.36\r\n
      Accept: image/webp,image/apng,image/*,*/*;q=0.8\r\n
      Referer: http://www.zju.edu.cn/\r\n
      Accept-Encoding: gzip, deflate\r\n
      Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7\r\n
      Cookie: Hm_lvt_fe30bbc1ee45421ec1679d1b8df8453=1570446546; Hm_lpvt_fe30bbc1ee45421ec1679d1b8df8453=1570446546\r\n
  \r\n
  [Full request URL: http://www.zju.edu.cn/_visitcount?siteId=3&type=1&columnId=5]
  [HTTP request 2/2]
  [Prev request in frame: 7]
  [Response in frame: 108]

```

图 32: 第二个会话的请求部分

```

Frame 108: 266 bytes on wire (2128 bits), 266 bytes captured (2128 bits)
Ethernet II, Src: Xiaomi_E1_96:87:83 (ec:41:18:96:87:83), Dst: Apple_26:23:08 (8c:85:90:26:23:08)
Internet Protocol Version 4, Src: 10.203.6.101, Dst: 192.168.31.229
Transmission Control Protocol, Src Port: 80, Dst Port: 64769, Seq: 13401, Ack: 1031, Len: 212
HyperText Transfer Protocol
  ▾ HTTP/1.1 200 OK\r\n
    ▶ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Response Version: HTTP/1.1
      Status Code: 200
      [Status Code Description: OK]
      Response Phrase: OK
      Date: Mon, 07 Oct 2019 11:09:04 GMT\r\n
      Server: Apache\r\n
      X-Frame-Options: SAMEORIGIN\r\n
      Frame-Options: SAMEORIGIN\r\n
      Content-Length: 0\r\n
      Set-Cookie: JSESSIONID=A2D436E81D514CB780F48D8CCD871C81; Path=/\r\n
      \r\n
      [HTTP response 2/2]
      [Time since request: 0.007000000 seconds]
      [Prev request in frame: 7]
      [Prev response in frame: 23]
      [Request in frame: 107]
      [Request URI: http://www.zju.edu.cn/_visitcount?siteId=3&type=1&columnId=5]
  
```

图 33: 第二个会话的响应部分

六、 实验结果分析与思考

1. 如果只想捕获某个特定 WEB 服务器 IP 地址相关的 HTTP 数据包，捕获过滤器应该怎么写？

host [IP] and port 80

2. ping 发送的是什么类型的协议数据包？什么情况下会出现 ARP 数据包？ping 一个域名和 ping 一个 IP 地址出现的数据包有什么不同？

ping 发送 ICMP 类型的数据包。

如果测试的 IP 地址不在 ARP 表中，则会出现 ARP 数据包。在本实验中，测试的 192.168.31.5 与本机处在同一网段，且没有在 ARP 表中，所以会有 ARP 数据包出现。当测试的 IP 地址与本机不在同一网段时，则向网关 192.168.31.1 发送数据，网关的 IP 地址在 ARP 表中，所以未出现 ARP 数据包。

ping 一个 IP 地址时，若其在 ARP 表中，则直接发送 ICMP 数据包。ping 一个域名时，则要先发送 DNS 数据包对域名进行解析。

3. traceroute 发送的是什么类型的协议数据包，整个路由跟踪过程是如何进行的？

traceroute 发送 UDP 类型的数据包。其首先发送 TTL 为 1 的数据包，到达第一个路由器后 TTL 减 1，这个数据包被丢弃，发回 TTL exceeded 的 ICMP 数据包。主机收到后产生 TTL 加 1 的 UDP 数据包，经过每一个路由器 TTL 都会减 1，减为 0 时发回 TTL exceeded 的 ICMP 数据包。依此往复，直到数据包到达目的主机。而当这个数据包到达目的主机时，由于其端口号大于 30000，而 UDP 协议的端口号不可高于 30000，所以目的主机会发回 Destination unreachable 的 ICMP 数据包。当收到这个数据包时，主机即可确定数据包已经到达目的，不再继续发送。

本实验在 macOS 下完成，属于 Unix-like 操作系统，与 Windows 下的过程有所不同。Windows 下主机发送 ICMP 数据包，而 Unix-like 的系统默认发送 UDP 数据包。

4. 如何理解 TCP 连接和 HTTP 会话？他们之间存在什么关系？

HTTP 位于应用层，TCP 位于传输层。HTTP1.1 协议中 TCP 采用长连接，进行三次握手后建立连接，两个用户之间可以持续通信，连接不会主动关闭。HTTP 会话过程首先建立 TCP 连接，接下来进行

持续的请求和接收数据。HTTP 会话所在的应用层位于 TCP 连接所在的传输层的上层，HTTP 会话负责建立连接和发送接收数据，但不需要掌握建立连接的细节和数据传输的细节。

5. DNS 为什么选择使用 UDP 协议进行传输？而 HTTP 为什么选择使用 TCP 协议？

DNS 数据包的数据量较小，通常不需要分包，使用 UDP 协议传输，可以极大地提高传输效率。TCP 要三次握手四次挥手，每次请求和响应都要有 ACK 报文，而 UDP 只有一次请求一次响应。可靠性的问题也容易在应用层进行控制。HTTP 的数据量通常较大，且对数据可靠性和完整性的要求较高，采用 TCP 协议传输可以有效且高效地保障大量数据传输的可靠性。

当 DNS 传输的数据量较大时，也会采用 TCP 传输。

七、讨论与心得

本次实验使用 Wireshark 软件，对流经电脑网卡的数据进行抓取和分析。通过本次实验，我对数据在网络中传输的形式有了一定的了解，对分层的概念有了更深入的理解，通过对几种协议数据头的分析，认识到不同特点的数据各自所采用的传输协议的不同。

在实验过程中，我遇到了抓取的数据包存在干扰的情况。首先我尽可能多地关闭了其他无关软件，并在开始实验前先查阅资料，了解将要发生的过程会产生什么样的数据包，并有针对性地进行清除缓存等操作。这样在实验过程中，可以较精准地获取想要的数据包进行分析。

我在过去曾使用浏览器的开发者工具进行过抓包的实践。使用 Wireshark 与浏览器开发者工具的最大不同就是，Wireshark 可以抓取流经网卡的所有数据包，并且获得的是数据包的原始数据，包含各层协议，而在浏览器中抓取的只有当前页面下发送和接收的数据，并且看到的只有顶层 HTTP/HTTPS 协议的内容。