

# 浙江大学

## 本科实验报告

课程名称： 计算机网络基础

实验名称： 网络协议分析

姓 名： 应承峻

学 院： 计算机学院

系： 计算机系

专 业： 软件工程

学 号： 3170103456

指导教师： 高艺

2019 年 1 月 11 日

# 浙江大学实验报告

实验名称： 网络协议分析 实验类型： 分析实验

同组学生： \_\_\_\_\_ 实验地点： 计算机网络实验室

## 一、 实验目的

- 进一步学习使用 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

- 熟练掌握网络协议分析软件 Wireshark 的使用
- 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

- 联网的 PC 机
- WireShark 协议分析软件

## 四、 操作方法与实验步骤

- 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：测试一个目标地址是否可达（在实验一基础上）
  - ✓ TRACE ROUTE：跟踪一个目标地址的途经路由（在实验一基础上）
  - ✓ NSLOOKUP：查询一个域名（在实验一基础上）
  - ✓ HTTP：访问一个网页
  - ✓ FTP：上传或下载一个文件
  - ✓ SMTP：发送一封邮件
  - ✓ POP3/IMAP：接收一封邮件
  - ✓ RTP：抓取一段音频流

提醒：为了避免捕获到大量无关数据包，影响实验观察，建议关闭所有无关软件。

## 五、 实验数据记录和处理

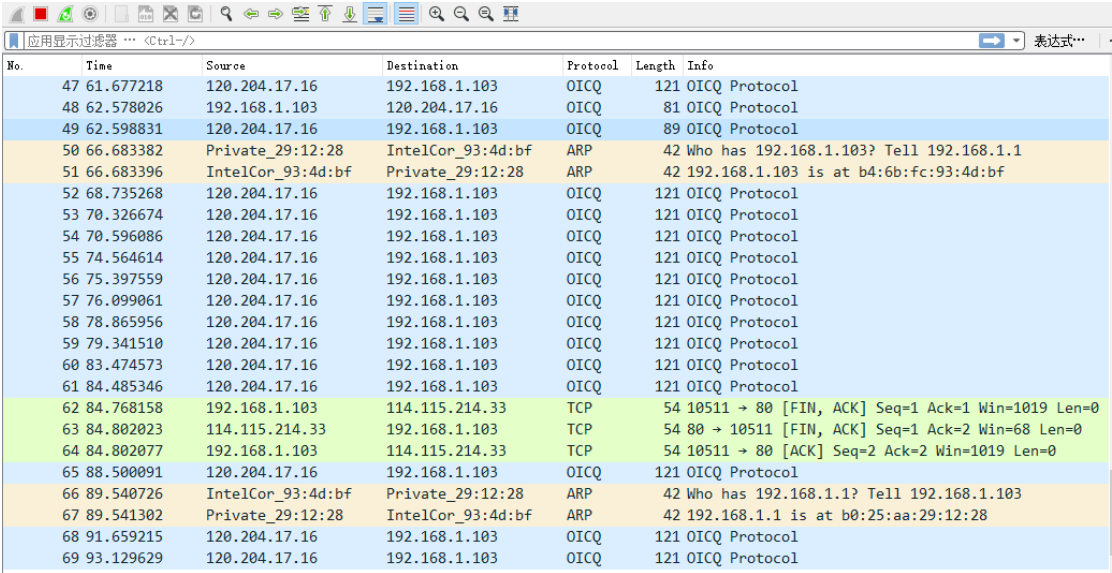
以下实验记录均需结合屏幕截图，进行文字标注和描述，图片应大小合适、关键部分清

晰可见，可直接在图片上进行标注，也可以单独用文本进行描述。（看完请删除本句）。

✧ Part One

- 打开 WireShark，开始捕获网络数据包后，你看到了什么？有哪些协议？

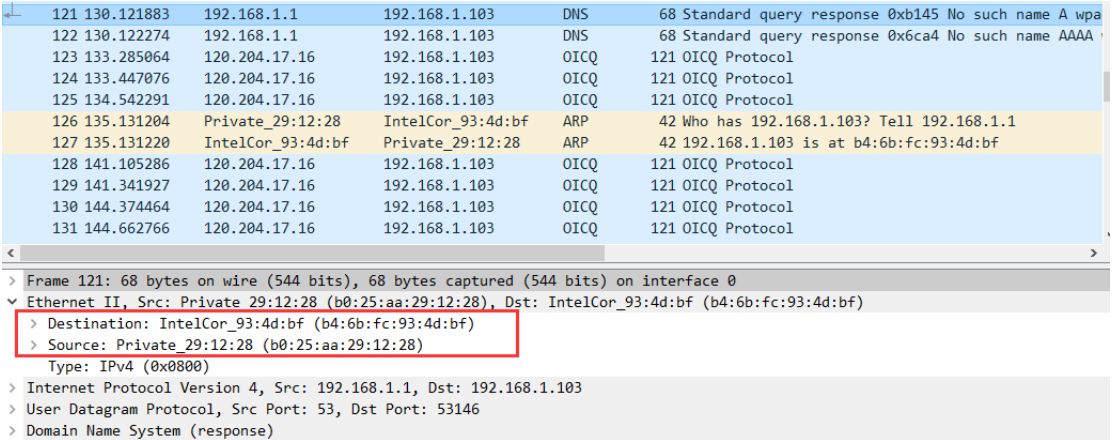
看到了捕获到的数据包，有 QICQ、ARP、TCP、UDP 等协议



No.	Time	Source	Destination	Protocol	Length	Info
47	61.677218	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
48	62.578026	192.168.1.103	120.204.17.16	OICQ	81	OICQ Protocol
49	62.598831	120.204.17.16	192.168.1.103	OICQ	89	OICQ Protocol
50	66.683382	Private_29:12:28	IntelCor_93:4d:bf	ARP	42	Who has 192.168.1.103? Tell 192.168.1.1
51	66.683396	IntelCor_93:4d:bf	Private_29:12:28	ARP	42	192.168.1.103 is at b4:6b:fc:93:4d:bf
52	68.735268	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
53	70.326674	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
54	70.596086	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
55	74.564614	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
56	75.397559	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
57	76.099061	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
58	78.865956	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
59	79.341510	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
60	83.474573	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
61	84.485346	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
62	84.768158	192.168.1.103	114.115.214.33	TCP	54	10511 → 80 [FIN, ACK] Seq=1 Ack=1 Win=1019 Len=0
63	84.802023	114.115.214.33	192.168.1.103	TCP	54	80 → 10511 [FIN, ACK] Seq=1 Ack=2 Win=68 Len=0
64	84.802077	192.168.1.103	114.115.214.33	TCP	54	10511 → 80 [ACK] Seq=2 Ack=2 Win=1019 Len=0
65	88.500091	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
66	89.540726	IntelCor_93:4d:bf	Private_29:12:28	ARP	42	Who has 192.168.1.1? Tell 192.168.1.103
67	89.541302	Private_29:12:28	IntelCor_93:4d:bf	ARP	42	192.168.1.1 is at b0:25:aa:29:12:28
68	91.659215	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
69	93.129629	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol

- 找一个包含 Ethernet 的数据包，这是什么协议？标出源和目标 MAC 地址。

这是一个 DNS 协议，源 MAC 地址和目标 MAC 地址如下图框线部分：



121	130.121883	192.168.1.1	192.168.1.103	DNS	68	Standard query response 0xb145 No such name A wpa
122	130.122274	192.168.1.1	192.168.1.103	DNS	68	Standard query response 0x6ca4 No such name AAAA
123	133.285064	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
124	133.447076	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
125	134.542291	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
126	135.131204	Private_29:12:28	IntelCor_93:4d:bf	ARP	42	Who has 192.168.1.103? Tell 192.168.1.1
127	135.131220	IntelCor_93:4d:bf	Private_29:12:28	ARP	42	192.168.1.103 is at b4:6b:fc:93:4d:bf
128	141.105286	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
129	141.341927	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
130	144.374464	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol
131	144.662766	120.204.17.16	192.168.1.103	OICQ	121	OICQ Protocol

> Frame 121: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface 0

> Ethernet II, Src: Private\_29:12:28 (b0:25:aa:29:12:28), Dst: IntelCor\_93:4d:bf (b4:6b:fc:93:4d:bf)

> Destination: IntelCor\_93:4d:bf (b4:6b:fc:93:4d:bf)

> Source: Private\_29:12:28 (b0:25:aa:29:12:28)

Type: IPv4 (0x0800)

> Internet Protocol Version 4, Src: 192.168.1.1, Dst: 192.168.1.103

> User Datagram Protocol, Src Port: 53, Dst Port: 53146

> Domain Name System (response)

- 找一个包含 IP 的数据包，这是什么协议？标出源 IP 地址、目标 IP 地址。

这是一个 TCP 协议。源 IP 地址：192.168.1.103，目标 IP 地址：52.80.229.89

143	156.001844	192.168.1.103	52.80.229.89	TCP	54 10614 → 80 [FIN, ACK] Seq=313 Ack=99 Win=261888 L
144	156.002255	192.168.1.103	52.80.229.89	TCP	66 10619 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 W
145	156.045498	52.80.229.89	192.168.1.103	TCP	58 80 → 10619 [SYN, ACK] Seq=0 Ack=1 Win=26883 Len=0
146	156.045596	192.168.1.103	52.80.229.89	TCP	54 10619 → 80 [ACK] Seq=1 Ack=1 Win=65535 Len=0
147	156.045775	192.168.1.103	52.80.229.89	HTTP	366 GET /message/updateTime?tags=commontags&updateTim
148	156.084536	52.80.229.89	192.168.1.103	TCP	54 80 → 10619 [ACK] Seq=1 Ack=313 Win=26800 Len=0
149	156.086587	52.80.229.89	192.168.1.103	HTTP	151 HTTP/1.1 200
150	156.086675	192.168.1.103	52.80.229.89	TCP	54 10619 → 80 [ACK] Seq=313 Ack=98 Win=65535 Len=0

> Frame 143: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0

> Ethernet II, Src: IntelCor\_93:4d:bf (b4:6b:fc:93:4d:bf), Dst: Private\_29:12:28 (b0:25:aa:29:12:28)

▼ Internet Protocol Version 4, Src: 192.168.1.103, Dst: 52.80.229.89

0100 .... = Version: 4

.... 0101 = Header Length: 20 bytes (5)

> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)

Total Length: 40

Identification: 0x21bd (8637)

> Flags: 0x4000, Don't fragment

Time to live: 64

Protocol: TCP (6)

Header checksum: 0x0000 [validation disabled]

[Header checksum status: Unverified]

Source: 192.168.1.103

Destination: 52.80.229.89

> Transmission Control Protocol, Src Port: 10614, Dst Port: 80, Seq: 313, Ack: 99, Len: 0

- 找一个 ARP 数据包，这是请求还是应答？标注发送者的 MAC 地址。

这是一个请求，发送者的 MAC 地址为 b0:25:aa:29:12:28

164	169.491430	Private_29:12:28	IntelCor_93:4d:bf	ARP	42 Who has 192.168.1.103? Tell 192.168.1.1
165	169.491443	IntelCor_93:4d:bf	Private_29:12:28	ARP	42 192.168.1.103 is at b4:6b:fc:93:4d:bf
166	171.204930	120.204.17.16	192.168.1.103	OICQ	121 OICQ Protocol
167	171.428664	120.204.17.16	192.168.1.103	OICQ	121 OICQ Protocol
168	173.239275	120.204.17.16	192.168.1.103	OICQ	121 OICQ Protocol
169	174.950155	192.168.1.103	52.80.229.89	TCP	54 10614 → 80 [RST, ACK] Seq=314 Ack=99 Win=0 Len=0
170	175.204273	120.204.17.16	192.168.1.103	OICQ	121 OICQ Protocol
171	175.699365	120.204.17.16	192.168.1.103	OICQ	121 OICQ Protocol
172	179.755311	202.89.233.101	192.168.1.103	TCP	54 443 → 10618 [RST, ACK] Seq=9909 Ack=1026 Win=0 Le
173	182.734840	192.168.1.103	120.204.17.16	OICQ	81 OICQ Protocol

> Frame 164: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0

> Ethernet II, Src: Private\_29:12:28 (b0:25:aa:29:12:28), Dst: IntelCor\_93:4d:bf (b4:6b:fc:93:4d:bf)

▼ Address Resolution Protocol (request)

Hardware type: Ethernet (1)

Protocol type: IPv4 (0x0800)

Hardware size: 6

Protocol size: 4

Opcode: request (1)

Sender MAC address: Private\_29:12:28 (b0:25:aa:29:12:28)

Sender IP address: 192.168.1.1

Target MAC address: 00:00:00\_00:00:00 (00:00:00:00:00:00)

Target IP address: 192.168.1.103

请在下面的每次捕获任务完成后，保存 Wireshark 抓包记录（.pcap 格式），随报告一起提交。每一个协议一个单独文件，文件名请取得便于理解。

## ◇ Part Two

- 使用 Ping 命令，测试某个 IP 地址的连通性，并捕获这次的数据包。数据包由几层协议构成？分别是什么协议？选择一个请求包和一个响应包，展开最高层协议的详细内容，标出请求包和应答包、类型、序号。

测试 IP 地址 10.79.248.102 的连通性

ip.addr == 10.79.248.102

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.1.103	10.79.248.102	ICMP	74	Echo (ping) request id=0x0001, seq=33/8448, ttl=64
2	0.002112	10.79.248.102	192.168.1.103	ICMP	74	Echo (ping) reply id=0x0001, seq=33/8448, ttl=57
4	1.008019	192.168.1.103	10.79.248.102	ICMP	74	Echo (ping) request id=0x0001, seq=34/8704, ttl=64
5	1.028860	10.79.248.102	192.168.1.103	ICMP	74	Echo (ping) reply id=0x0001, seq=34/8704, ttl=57
9	2.023435	192.168.1.103	10.79.248.102	ICMP	74	Echo (ping) request id=0x0001, seq=35/8960, ttl=64
10	2.025047	10.79.248.102	192.168.1.103	ICMP	74	Echo (ping) reply id=0x0001, seq=35/8960, ttl=57
11	3.039771	192.168.1.103	10.79.248.102	ICMP	74	Echo (ping) request id=0x0001, seq=36/9216, ttl=64
12	3.046278	10.79.248.102	192.168.1.103	ICMP	74	Echo (ping) reply id=0x0001, seq=36/9216, ttl=57

数据包由 4 层协议组成，分别是 Frame, Ethernet II, IPv4, ICMP 协议

```
> Frame 12: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
> Ethernet II, Src: Private_29:12:28 (b0:25:aa:29:12:28), Dst: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf)
> Internet Protocol Version 4, Src: 10.79.248.102, Dst: 192.168.1.103
> Internet Control Message Protocol
```

请求包最高层协议的内容，类型为 8（Echo Request），序号为 36

```
> Frame 11: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf), Dst: Private_29:12:28 (b0:25:aa:29:12:28)
> Internet Protocol Version 4, Src: 192.168.1.103, Dst: 10.79.248.102
< Internet Control Message Protocol
  Type: 8 (Echo (ping) request)
  Code: 0
  Checksum: 0x4d37 [correct]
  [Checksum Status: Good]
  Identifier (BE): 1 (0x0001)
  Identifier (LE): 256 (0x0100)
  Sequence number (BE): 36 (0x0024)
  Sequence number (LE): 9216 (0x2400)
  [Response frame: 12]
< Data (32 bytes)
  Data: 6162636465666768696a6b6c6d6e6f707172737475767761...
  [Length: 32]
```

响应包最高层协议的内容，类型为 0（Echo Reply），序号为 36

```
> Frame 12: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
> Ethernet II, Src: Private_29:12:28 (b0:25:aa:29:12:28), Dst: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf)
> Internet Protocol Version 4, Src: 10.79.248.102, Dst: 192.168.1.103
< Internet Control Message Protocol
  Type: 0 (Echo (ping) reply)
  Code: 0
  Checksum: 0x5537 [correct]
  [Checksum Status: Good]
  Identifier (BE): 1 (0x0001)
  Identifier (LE): 256 (0x0100)
  Sequence number (BE): 36 (0x0024)
  Sequence number (LE): 9216 (0x2400)
  [Request frame: 11]
  [Response time: 6.507 ms]
< Data (32 bytes)
  Data: 6162636465666768696a6b6c6d6e6f707172737475767761...
  [Length: 32]
```

- 使用 Tracert 命令（Mac 下使用 Traceroute 命令），跟踪某个外部 IP 地址的路由，并捕获这次的数据包。数据包由几层协议构成？分别是什么协议？查看并标记多个请求包的 IP 协议层的 TTL 字段，发现了什么规律？选择一个请求包和一个响应包，展开最高层协议的详细内容，标出类型、序号等关键字段。与 Ping 命令的数据包有什么不同？



▼ Data (64 bytes)

Data: 00...  
[Length: 64]

- 使用 `nslookup` 命令，查询某个域名，并捕获这次的数据包。数据包由几层协议构成？分别是什么协议？标记 UDP 协议层的端口字段。选择一个请求包和一个响应包，展开最高层协议的详细内容，标出类型、序号、域名信息。

数据包由 5 层协议构成分别是 Frame, Ethernet II, IPv4, UDP 和 DNS 协议

```
> Frame 8: 224 bytes on wire (1792 bits), 224 bytes captured (1792 bits) on interface 0  
> Ethernet II, Src: VivoMobi_60:5e:9b (2c:ff:ee:60:5e:9b), Dst: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf)  
> Internet Protocol Version 4, Src: 192.168.43.237, Dst: 192.168.43.7  
> User Datagram Protocol, Src Port: 53, Dst Port: 59002  
> Domain Name System (response)
```

```
▼ User Datagram Protocol, Src Port: 53, Dst Port: 59002  
  Source Port: 53  
  Destination Port: 59002  
  Length: 190  
  Checksum: 0x2b1b [unverified]  
  [Checksum Status: Unverified]  
  [Stream index: 3]  
  > [Timestamps]
```

请求包:

```
> Frame 9: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface 0  
> Ethernet II, Src: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf), Dst: VivoMobi_60:5e:9b (2c:ff:ee:60:5e:9b)  
> Internet Protocol Version 4, Src: 192.168.43.7, Dst: 192.168.43.237  
> User Datagram Protocol, Src Port: 59003, Dst Port: 53  
▼ Domain Name System (query)  
  Transaction ID: 0x0003  
  ▼ Flags: 0x0100 Standard query  
    0... .. = Response: Message is a query  
    .000 0... .. = Opcode: Standard query (0)  
    .... .. = Truncated: Message is not truncated  
    .... ..1 .... = Recursion desired: Do query recursively  
    .... ..0... .. = Z: reserved (0)  
    .... ..0 .... = Non-authenticated data: Unacceptable  
  Questions: 1  
  Answer RRs: 0  
  Authority RRs: 0  
  Additional RRs: 0  
  ▼ Queries  
    > cn.bing.com: type AAAA, class IN  
    [Response in: 10]
```

响应包:

```

Domain Name System (response)
Transaction ID: 0x0003
Flags: 0x8180 Standard query response, No error
1... .. = Response: Message is a response
.000 0... .. = Opcode: Standard query (0)
... ..0... .. = Authoritative: Server is not an authority for domain
... ..0... .. = Truncated: Message is not truncated
... ..1... .. = Recursion desired: Do query recursively
... ..1... .. = Recursion available: Server can do recursive queries
... ..0... .. = Z: reserved (0)
... ..0... .. = Answer authenticated: Answer/authority portion was not authenticated by the server
... ..0... .. = Non-authenticated data: Unacceptable
... ..0000 = Reply code: No error (0)
Questions: 1
Answer RRs: 3
Authority RRs: 0
Additional RRs: 0
Queries
> cn.bing.com: type AAAA, class IN
Answers
> cn.bing.com: type CNAME, class IN, cname cn-bing-com.cn.a-0001.a-msedge.net
> cn-bing-com.cn.a-0001.a-msedge.net: type CNAME, class IN, cname cn.cn-0001.cn-msedge.net
> cn.cn-0001.cn-msedge.net: type CNAME, class IN, cname cn-0001.cn-msedge.net
[Request In: 9]
[Time: 0.000839000 seconds]

```

### ✧ Part Three

- 运行 `ipconfig /flushdns` 命令清空 DNS 缓存，然后打开浏览器，访问一个网页，并捕获这次的数据包（网页完全打开后，停止捕获）。数据包由几层协议构成？分别是什么协议？标出数据包的源和目标 IP 地址、源和目标端口。

数据包由 5 层协议构成分别是 Frame, Ethernet II, IPv4, UDP 和 DNS 协议：

```

<
> Frame 1: 71 bytes on wire (568 bits), 71 bytes captured (568 bits)
> Ethernet II, Src: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf), Dst: Private_29:12:28 (b0:25:aa:29:12:28)
> Internet Protocol Version 4, Src: 192.168.1.103, Dst: 192.168.1.1
> User Datagram Protocol, Src Port: 58942, Dst Port: 53
> Domain Name System (query)

```

端口：

```

User Datagram Protocol, Src Port: 58942, Dst Port: 53
Source Port: 58942
Destination Port: 53
Length: 37
Checksum: 0x83ef [unverified]
[Checksum Status: Unverified]
[Stream index: 0]
> [Timestamps]

```

IP 地址：

```

Internet Protocol Version 4, Src: 192.168.1.103, Dst: 192.168.1.1
0100 .... = Version: 4
.... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 57
Identification: 0x35aa (13738)
> Flags: 0x0000
Time to live: 64
Protocol: UDP (17)
Header checksum: 0x0000 [validation disabled]
[Header checksum status: Unverified]
Source: 192.168.1.103
Destination: 192.168.1.1

```



- 找到建立 TCP 连接的三个数据包（称为三次握手），展开 TCP 协议层的 Flags 字段，分别标记三个数据包的 SYN 标志位和 ACK 标志位。

（注：本题直接用上一题抓到的包）

3 0.003616	192.168.1.103	202.89.233.100	TCP	66 4766 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
4 0.038819	202.89.233.100	192.168.1.103	TCP	66 443 → 4766 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1440 WS=256 SACK_PERM=1
5 0.038913	192.168.1.103	202.89.233.100	TCP	54 4766 → 443 [ACK] Seq=1 Ack=1 Win=66048 Len=0

第一次握手：

```

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
[TCP Flags: .....S.]

```

第二次握手：

```

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
.... ....0 = Fin: Not set
[TCP Flags: .....A..S.]

```

第三次握手：

```

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
.... ....0 = Fin: Not set
[TCP Flags: .....A....]

```

- 选择一个包，点击右键，选择跟踪一个 TCP 流，截取完整的 HTTP 请求消息和部分响应消息，标记 HTTP 请求头部的 Method 字段、URI 字段和 Host 字段，标记 HTTP 响应头部的 Status Code 字段、Content-Type 和 Content-Length 字段，以及区分响应头部和体部的标记（单独的回车换行符）。

（注：本题直接用上一题抓到的包）

HTTP 请求：

```
Hypertext Transfer Protocol
GET /message/updateTime?tags=commontags&updateTime=d751713988987e9331980363e24189ce1544026800519&serviceVersion=1.1.2.627 HTTP/1.1\r\n
Content-Length: 4\r\n
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; WOW64; Trident/7.0; .NET4.0C; .NET4.0E)\r\n
Host: api.foxitreader.cn\r\n
Cache-Control: no-cache\r\n
\r\n
[Full request URI: http://api.foxitreader.cn/message/updateTime?tags=commontags&updateTime=d751713988987e9331980363e24189ce1544026800519&serviceVer-
[HTTP request 1/1]
[Response in frame: 93]
File Data: 4 bytes
Data (4 bytes)
Data: 6e756c6c
[Length: 4]
```

HTTP 响应:

```
Hypertext Transfer Protocol
HTTP/1.1 200 \r\n
[Expert Info (Chat/Sequence): HTTP/1.1 200 \r\n]
Response Version: HTTP/1.1
Status Code: 200
[Status Code Description: OK]
Date: Sat, 11 Jan 2020 11:41:21 GMT\r\n
Content-Length: 0\r\n
[Content Length: 0]
Connection: keep-alive\r\n
\r\n
[HTTP response 1/1]
[Time since request: 0.036252000 seconds]
[Request in frame: 91]
[Request URI: http://api.foxitreader.cn/message/updateTime?tags=commontags&updateTime=d751713988987e9331980363e24189ce1544026800519&serviceVersion=1.]
```

- 使用过滤器 `tcp.stream eq X`, 让 X 从 0 开始变化, 直到没有数据。观察总共捕获到了几个 TCP 连接 (一个 TCP 流对应一个 TCP 连接)? 存在几个 HTTP 会话 (一对 HTTP 请求和响应对应一次 HTTP 会话)? 注意: 一个 TCP 流上可能存在多个 HTTP 会话。

总共观察到 6 个 TCP 连接 (1~6, 0 不算), 只存在 1 个 HTTP 会话

No.	Time	Source	Destination	Protocol	Length	Info
98	6.134629	112.34.111.235	192.168.1.103	TLSv1.2	85	Encrypted Alert
99	6.135322	112.34.111.235	192.168.1.103	TCP	54	443 → 4975 [FIN, ACK] Seq=32 Ack=1 Win=17771 Len=0
100	6.135355	192.168.1.103	112.34.111.235	TCP	54	4975 → 443 [ACK] Seq=1 Ack=33 Win=65205 Len=0

No.	Time	Source	Destination	Protocol	Length	Info

#### ✧ Part Four

- 打开邮件客户端 Foxmail 或 Outlook, 写一封电子邮件 (建议采用直接送达方式), 并捕获这次的数据包。捕获到的数据包由几层协议构成? 分别是什么协议? 标出数据包的源和目标 IP 地址、源和目标端口。

数据包由 5 层协议构成分别是 Frame, Ethernet II, IPv4, TCP 和 SMTP 协议:

```
<
> Frame 36: 76 bytes on wire (608 bits), 76 bytes captured (608 bits) on interface 0
> Ethernet II, Src: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf), Dst: Private_29:12:28 (b0:25:aa:29:12:28)
> Internet Protocol Version 4, Src: 192.168.1.103, Dst: 220.181.12.13
> Transmission Control Protocol, Src Port: 9634, Dst Port: 25, Seq: 1, Ack: 66, Len: 22
> Simple Mail Transfer Protocol
```

IP 地址:

```
▼ Internet Protocol Version 4, Src: 192.168.1.103, Dst: 220.181.12.13
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 62
    Identification: 0x4af8 (19192)
  ▼ Flags: 0x4000, Don't fragment
    0... .... = Reserved bit: Not set
    .1.. .... = Don't fragment: Set
    ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment offset: 0
    Time to live: 64
    Protocol: TCP (6)
    Header checksum: 0x0000 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.1.103
    Destination: 220.181.12.13
```

端口号:

```
▼ Transmission Control Protocol, Src Port: 9634, Dst Port: 25, Seq: 1, Ack: 66, Len: 22
  Source Port: 9634
  Destination Port: 25
  [Stream index: 1]
  [TCP Segment Len: 22]
  Sequence number: 1 (relative sequence number)
  [Next sequence number: 23 (relative sequence number)]
  Acknowledgment number: 66 (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x018 (PSH, ACK)
  Window size value: 256
  [Calculated window size: 65536]
  [Window size scaling factor: 256]
  Checksum: 0xab02 [unverified]
  [Checksum Status: Unverified]
  Urgent pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
  TCP payload (22 bytes)
```

- 跟踪 TCP 流，查看 SMTP 握手消息采用的是什么（HELO 还是 EHLO）？标出 SMTP 协议层中的客户端机器名、发件人地址、收件人地址、认证的用户名和密码（如果是 EHLO 握手方式）、邮件正文（内容过长可截取关键部分）。

握手消息采用 EHLO

```

220 163.com Anti-spam GT for Coremail System (163com[20141201])
EHLO DESKTOP-Q2K9ADR
250-mail
250-PIPELINING
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN PLAIN
250-coremail 1Uxr2xKj7kG0xkI17xGrU7I0s8FY2U3Uj8Cz28x1UUUUU7Ic2I0Y2Ure2DDwUCa0xDrUUUUj
250-STARTTLS
250 8BITMIME
AUTH LOGIN
334 dXNlcm5hbWU6
WkpVU1JFX0cxM18yMDE5QDE2My5jb20=
334 UGFzc3dvcmQ6
aHR0bmIyMzM=
235 Authentication successful
MAIL FROM: <ZJUSRE_G13_2019@163.com>
250 Mail OK
RCPT TO: <ZJUSRE_G13_2019@163.com>
250 Mail OK
DATA
354 End data with <CR><LF>.<CR><LF>
Date: Sat, 11 Jan 2020 22:31:33 +0800
From: "ZJUSRE_G13_2019@163.com" <ZJUSRE_G13_2019@163.com>
To: ZJUSRE_G13_2019 <ZJUSRE_G13_2019@163.com>
Subject: hello
X-Priority: 3
X-GUID: 84F9AD0A-958D-4C2C-8ED8-AB0EF5103B5D
X-Has-Attach: no
X-Mailer: Foxmail 7.2.15.65[cn]
Mime-Version: 1.0
Message-ID: <202001112231306418711@163.com>
Content-Type: multipart/alternative;
        boundary="-----_001_NextPart882511847830_-----"

This is a multi-part message in MIME format.

-----_001_NextPart882511847830_-----

```

发送的邮件内容

```

<html><head><meta http-equiv=3D"content-type" content=3D"text/html; charse=
t=3Dus-ascii"><style>body { line-height: 1.5; }body { font-size: 10.5pt; f=
ont-family: 'Microsoft YaHei UI'; color: rgb(0, 0, 0); line-height: 1.5; }=
</style></head><body>=0A<div><span></span><br></div>=0A<div>hello2019</div>
<hr style=3D"width: 210px; height: 1px;" color=3D"#b5c4df" size=3D"1" ali=
gn=3D"left">=0A<div><span><div style=3D"MARGIN: 10px; FONT-FAMILY: verdana=
; FONT-SIZE: 10pt"><div>ZJUSRE_G13_2019@163.com</div></div></span></div>=
=0A</body></html>
-----_001_NextPart882511847830_-----

```

- 打开邮件客户端 **Foxmail** 或 **Outlook**，收取自己邮箱中的邮件（请在邮件服务器中设置允许 **POP3** 或者 **IMAP**），并捕获这次的数据包。捕获到的数据包由几层协议构成？分别是什么协议？标出数据包的源和目标 IP 地址、源和目标端口。

数据包由 5 层协议构成分别是 Frame, Ethernet II, IPv4, TCP 和 IAMP 协议：

```
<
> Frame 7: 136 bytes on wire (1088 bits), 136 bytes captured (1088 bits) on interface 0
> Ethernet II, Src: Private_29:12:28 (b0:25:aa:29:12:28), Dst: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf)
> Internet Protocol Version 4, Src: 123.126.97.78, Dst: 192.168.1.103
> Transmission Control Protocol, Src Port: 143, Dst Port: 9685, Seq: 1, Ack: 1, Len: 82
> Internet Message Access Protocol
```

IP 地址:

```
Internet Protocol Version 4, Src: 123.126.97.78, Dst: 192.168.1.103
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 122
    Identification: 0xf6d7 (63191)
  > Flags: 0x4000, Don't fragment
    0... .. = Reserved bit: Not set
    .1.. .. = Don't fragment: Set
    ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment offset: 0
    Time to live: 48
    Protocol: TCP (6)
    Header checksum: 0xb4ca [validation disabled]
    [Header checksum status: Unverified]
    Source: 123.126.97.78
    Destination: 192.168.1.103
```

端口号:

```
Transmission Control Protocol, Src Port: 143, Dst Port: 9685, Seq: 1, Ack: 1, Len: 82
  Source Port: 143
  Destination Port: 9685
  [Stream index: 1]
  [TCP Segment Len: 82]
  Sequence number: 1 (relative sequence number)
  [Next sequence number: 83 (relative sequence number)]
  Acknowledgment number: 1 (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x018 (PSH, ACK)
    Window size value: 115
    [Calculated window size: 14720]
    [Window size scaling factor: 128]
    Checksum: 0x70e7 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
  TCP payload (82 bytes)
```

- 跟踪 TCP 流，标出 POP3 或 IMAP 协议层中的认证用户名和密码、以及接收的邮件正文（内容过长可截取关键部分）。

认证用户名和密码:

```
Wireshark · 追踪 TCP 流 (tcp.stream eq 1) · IAMP.pcapng

* OK Coremail System IMAP Server Ready(163com[10774b260cc7a37d26d71b52404dcf5c])
C1 CAPABILITY
* CAPABILITY IMAP4rev1 XLIST SPECIAL-USE ID LITERAL+ STARTTLS XAPPLEPUSHSERVICE UIDPLUS X-CM-EXT-1
C1 OK CAPABILITY completed
C2 ID ("name" "com.tencent.foxmail" "version" "7.2.15.65" "os" "windows" "os-version" "6.2" "vendor" "tencent limited" "contact" "foxmail@foxmail.com")
* ID ("name" "Coremail Imap" "vendor" "Mailtech" "TransID" "t51f3QD3i7MJ3hle")
C2 OK ID completed
C3 LOGIN ZJUSRE_G13_2019@163.com "httnb233"
C3 OK LOGIN completed
C4 CAPABILITY
* CAPABILITY IMAP4rev1 XLIST SPECIAL-USE ID LITERAL+ STARTTLS XAPPLEPUSHSERVICE UIDPLUS X-CM-EXT-1
C4 OK CAPABILITY completed
C5 STATUS "INBOX" (MESSAGES RECENT UIDVALIDITY)
```

接收的邮件正文:

```
<html><head><meta http-equiv=3D"content-type" content=3D"text/html; charse=
t=3Dus-ascii"><style>body { line-height: 1.5; }body { font-size: 10.5pt; f=
ont-family: 'Microsoft YaHei UI'; color: rgb(0, 0, 0); line-height: 1.5; }=
</style></head><body>=0A<div><span></span><br></div>=0A<div>hello2019</div>
<hr style=3D"width: 210px; height: 1px;" color=3D"#b5c4df" size=3D"1" ali=
gn=3D"left">=0A<div><span><div style=3D"MARGIN: 10px; FONT-FAMILY: verdana=
; FONT-SIZE: 10pt"><div>ZJUSRE_G13_2019@163.com</div></div></span></div>=
=0A</body></html>
-----=_001_NextPart882511847830_-----
)
```

## ✧ Part Five

本部分需要边操作，边捕获，请在每次操作后暂停捕获，或者使用过滤器。建议通过 FTP 命令行进行实验，也可以使用 FTP 图形客户端。

- 运行 FTP xxx.com 命令，连接并登录服务器，输入用户名和帐号（如果是免费服务器，可以使用匿名帐号 Anonymous，密码是任意的邮箱）。捕获到的数据包由几层协议构成？分别是什么协议？标出数据包的源和目标 IP 地址、源和目标端口。

数据包由 5 层协议构成分别是 Frame, Ethernet II, IPv4, TCP 和 FTP 协议:

```
> Frame 1372: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface 0
> Ethernet II, Src: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf), Dst: Private_29:12:28 (b0:25:aa:29:12:28)
> Internet Protocol Version 4, Src: 192.168.1.103, Dst: 10.214.149.223
> Transmission Control Protocol, Src Port: 3917, Dst Port: 21, Seq: 37, Ack: 131, Len: 28
> File Transfer Protocol (FTP)
  [Current working directory: ]
```

端口号:

```

Transmission Control Protocol, Src Port: 3917, Dst Port: 21, Seq: 37, Ack: 131, Len: 28
  Source Port: 3917
  Destination Port: 21
  [Stream index: 1]
  [TCP Segment Len: 28]
  Sequence number: 37 (relative sequence number)
  [Next sequence number: 65 (relative sequence number)]
  Acknowledgment number: 131 (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x018 (PSH, ACK)

```

IP 地址:

```

Internet Protocol Version 4, Src: 192.168.1.103, Dst: 10.214.149.223
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 68
  Identification: 0x0b5d (2909)
  Flags: 0x4000, Don't fragment
    0... .... = Reserved bit: Not set
    .1.. .... = Don't fragment: Set
    ..0. .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment offset: 0
  Time to live: 64
  Protocol: TCP (6)
  Header checksum: 0x0000 [validation disabled]
  [Header checksum status: Unverified]
  Source: 192.168.1.103
  Destination: 10.214.149.223

```

- 跟踪 TCP 流，标注客户端发出的登录命令、用户名、密码以及服务器的响应。

927	3.165162	10.214.149.223	192.168.1.103	FTP	74	Response: 220 (vsFTPd 3.0.2)
928	3.165438	192.168.1.103	10.214.149.223	FTP	64	Request: AUTH TLS
1018	3.252726	10.214.149.223	192.168.1.103	FTP	92	Response: 530 Please login with USER and PASS.
1019	3.252969	192.168.1.103	10.214.149.223	FTP	64	Request: AUTH SSL
1024	3.430856	10.214.149.223	192.168.1.103	FTP	92	Response: 530 Please login with USER and PASS.
1370	5.237985	192.168.1.103	10.214.149.223	FTP	76	Request: USER anonymous
1371	5.564367	10.214.149.223	192.168.1.103	FTP	88	Response: 331 Please specify the password.
1372	5.564592	192.168.1.103	10.214.149.223	FTP	82	Request: PASS anonymous@example.com
1373	5.626103	10.214.149.223	192.168.1.103	FTP	72	Response: 230 Login successful.

- 执行列目录操作 (ls)，在新捕获的数据包中跟踪 TCP 流，标注客户端发出的命令、以及服务器的响应。查看是否建立了一个新的 TCP 连接，跟踪该连接的 TCP 流。建议连接校内服务器，如果服务器在校外，可能需要先执行 passive 命令（下同）。新建了一个 TCP 连接。

No.	Time	Source	Destination	Protocol	Length	Info
6	0.313532	192.168.1.103	10.214.149.223	TCP	66	4206 → 9423 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=128 SACK_PERM=1
7	0.328802	10.214.149.223	192.168.1.103	TCP	66	9423 → 4206 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
8	0.328888	192.168.1.103	10.214.149.223	TCP	54	4206 → 9423 [ACK] Seq=1 Ack=1 Win=4194304 Len=0
10	0.345221	10.214.149.223	192.168.1.103	FTP-DA...	462	FTP Data: 408 bytes (PASV) (LIST)
11	0.346370	10.214.149.223	192.168.1.103	TCP	54	9423 → 4206 [FIN, ACK] Seq=409 Ack=1 Win=29312 Len=0
12	0.346428	192.168.1.103	10.214.149.223	TCP	54	4206 → 9423 [ACK] Seq=1 Ack=410 Win=4193792 Len=0
13	0.346560	192.168.1.103	10.214.149.223	TCP	54	4206 → 9423 [FIN, ACK] Seq=1 Ack=410 Win=4193792 Len=0
14	0.350388	10.214.149.223	192.168.1.103	TCP	54	9423 → 4206 [ACK] Seq=410 Ack=2 Win=29312 Len=0

No.	Time	Source	Destination	Protocol	Length	Info
50	33.223849	192.168.1.103	10.214.149.223	TCP	66	4208 → 42949 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=128 SACK_PERM=1
51	33.250817	10.214.149.223	192.168.1.103	TCP	66	42949 → 4208 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
52	33.250404	192.168.1.103	10.214.149.223	TCP	54	4208 → 42949 [ACK] Seq=1 Ack=1 Win=4194304 Len=0
53	33.263320	10.214.149.223	192.168.1.103	FTP-DA...	462	FTP Data: 408 bytes (PASV) (LIST)
54	33.263320	10.214.149.223	192.168.1.103	TCP	54	42949 → 4208 [FIN, ACK] Seq=409 Ack=1 Win=29312 Len=0
55	33.263440	192.168.1.103	10.214.149.223	TCP	54	4208 → 42949 [ACK] Seq=1 Ack=410 Win=4193792 Len=0
56	33.263639	192.168.1.103	10.214.149.223	TCP	54	4208 → 42949 [FIN, ACK] Seq=1 Ack=410 Win=4193792 Len=0
57	33.268321	10.214.149.223	192.168.1.103	TCP	54	42949 → 4208 [ACK] Seq=410 Ack=2 Win=29312 Len=0

- 执行更换目录操作 (cd)，在新捕获的数据包中跟踪 TCP 流，标注客户端发出的命令、以及服务器的响应。

```
<
> Frame 10: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
> Ethernet II, Src: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf), Dst: Private_29:12:28 (b0:25:aa:29:12:28)
> Internet Protocol Version 4, Src: 192.168.1.103, Dst: 10.214.149.223
> Transmission Control Protocol, Src Port: 9086, Dst Port: 21, Seq: 1, Ack: 1, Len: 12
v File Transfer Protocol (FTP)
  > CWD course\r\n
  [Current working directory: ]
```

服务器的响应:

```
<
> Frame 17: 91 bytes on wire (728 bits), 91 bytes captured (728 bits) on interface 0
> Ethernet II, Src: Private_29:12:28 (b0:25:aa:29:12:28), Dst: IntelCor_93:4d:bf (b4:6b:fc:93:4d:bf)
> Internet Protocol Version 4, Src: 10.214.149.223, Dst: 192.168.1.103
> Transmission Control Protocol, Src Port: 21, Dst Port: 9086, Seq: 1, Ack: 13, Len: 37
✓ File Transfer Protocol (FTP)
  > 250 Directory successfully changed.\r\n
  [Current working directory: course]
```

- 执行下载文件操作 (**get filename**)，如果是二进制文件，先执行 **binary** 命令。在新捕获的数据包中跟踪 **TCP** 流，标注客户端发出的命令、以及服务器的响应。查看是否建立了一个新的 **TCP** 连接，跟踪该连接的 **TCP** 流（内容较长时截取部分关键内容）。

是建立了一个新的 TCP 连接。

客户端发送的命令:

[illegible]

服务端的响应:





- 邮件客户端发送一封电子邮件，需要几次请求、响应消息的交互？消息的一般格式是什么？邮件正文结束的标记是什么？

答：①需要 2 次请求响应交互，一次用于登录验证，另一次用于发送邮件。

②消息的一般格式是 `multipart/alternative`

③以回车+”.”+回车结束

- 邮件客户端接收一封电子邮件，需要几次请求、响应消息的交互？消息的一般格式是什么？用户名和密码是否经过了加密处理？

答：①需要 2 次请求响应交互，一次用于登录验证，另一次用于接收邮件。

②消息的一般格式是 `text/html`

③没有经过加密处理

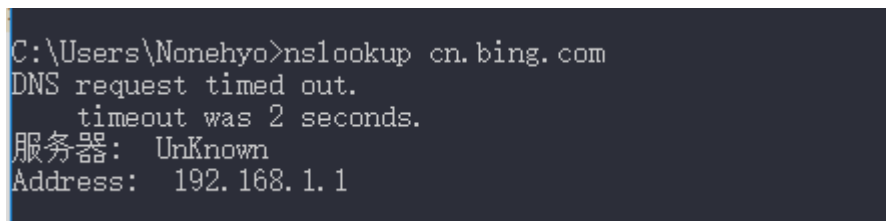
- 登录 FTP 服务器时，会产生几个 TCP 连接？列目录和上传或者下载文件时，会产生几个 TCP 连接？

答：①2 个 TCP 连接 ②1 个 TCP 连接

## 七、 讨论、心得

在本次实验中，我主要通过了 WireShark 对计算机网络的各种命令以及邮件传输、FTP 传输进行了抓包处理，并分析了其内部的协议。在实验的过程中主要遇到如下问题：

1. 使用 `nslookup` 命令时，出现如下问题：



```
C:\Users\Nonehyo>nslookup cn.bing.com
DNS request timed out.
    timeout was 2 seconds.
服务器:  UnKnown
Address:  192.168.1.1
```

推测可能是网关出现问题，所以改用手机热点连接后即可。

2. 当使用 QQ 邮箱登录 Foxmail 时，抓不到 SMTP 或者 POP3/IAMP 协议的数据包，初步判断可能是服务器协议配置的问题，于是换了一个 163 邮箱就可以了。