

北京邮电大学计算机网络第二次实验报告

课程名称	计算机网络	学院	计算机学院	指导教师	蒋砚军
班 级	班内序号	学 号	学生姓名	成绩	
2020211304	09	2020211346	倪玮昊		
实 验 内 容	实验内容：1) 捕获在连接 Internet 过程中产生的网络层分组：DHCP 分组，ARP 分组，IP 数据分组，ICMP 分组；2) 分析各种分组的格式，说明各种分组在建立网络连接过程中的作用；3) 分析IP 数据分组分片的结构；4) 分析TCP 建立连接，拆除连接和数据通信的流程。	实验目的：通过本次实验了解计算机上网的工作过程，学习各种网络层分组的格式及其作用，理解长度大于1500 字节IP 数据组分片传输的结构。			
学生实验报告（附页）	见附页实验报告。				
实	评语:				
验					
成					
绩					
评	成绩:				
定					
			指导教师签名:		
		年	月	日	

一、实验总体概述：

①实验类型：

协议分析型

②实验内容：

- 1) 捕获在连接 Internet 过程中产生的网络层分组：DHCP 分组，ARP 分组，IP 数据分组，ICMP 分组。
- 2) 分析各种分组的格式，说明各种分组在建立网络连接过程中的作用。
- 3) 分析 IP 数据分组分片的结构。
- 4) 分析 TCP 建立连接，拆除连接和数据通信的流程。

③ 实验目的：

通过本次实验了解计算机上网的工作过程，学习各种网络层分组的格式及其作用，理解 长度大于 1500 字节 IP 数据组分片传输的结构。

④ 实验环境：

1 台系统为Win10 的电脑，并且采用的是 WLAN 的方式连接网络。

⑤ 实验分组：

2020211346 倪玮昊

二、实验步骤与分析：

①DHCP 分组：

1. 实验前的准备工作：

第一步：开启Wireshark监控，设置捕获过滤器，仅捕获UDP报文,Capture == >Interface== > 选中所用网卡== > 点击 Start,捕获的结果如下：

文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)

udp

No.	Time	Source	Destination	Protocol	Length	Info
5809	305.012224	192.168.0.107	192.168.0.255	UDP	305	54915 → 54
5818	306.013828	192.168.0.107	192.168.0.255	UDP	305	54915 → 54
5819	306.176788	120.232.21.227	192.168.0.107	OICQ	129	OICQ Protc
5828	307.017701	192.168.0.107	192.168.0.255	UDP	305	54915 → 54
5832	308.012664	192.168.0.107	192.168.0.255	UDP	305	54915 → 54

Frame 1: 305 bytes on wire (2440 bits), 305 bytes captured (2440 bits) on interface \Dev
 Ethernet II, Src: LiteonTe_4d:ad:27 (e0:0a:f6:4d:ad:27), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
 Internet Protocol Version 4, Src: 192.168.0.107, Dst: 192.168.0.255
 User Datagram Protocol, Src Port: 54915, Dst Port: 54915
 Data (263 bytes)

0000	ff ff ff ff ff ff e0 0a	f6 4d ad 27 08 00 45 00M'..E.
0010	01 23 1f 01 00 00 80 11	98 0e c0 a8 00 6b c0 a8	..#.....k..
0020	00 ff d6 83 d6 83 01 0f	e0 92 00 4c 41 50 54 4fLAPTO
0030	50 2d 4f 53 38 30 49 44	4d 52 00 00 00 00 00 00	P-OS80ID MR.....
0040	00 00 30 62 47 fd d5 01	00 00 f0 ba 7f ce 99 00	..0bG.....
0050	00 00 00 00 a1 f7 d5 01	00 00 33 27 00 00 00 003'....
0060	00 00 20 62 47 fd d5 01	00 00 e0 ad ab f7 d5 01	..bG.....
0070	00 00 50 78 03 fe d5 01	00 00 b0 be 7f ce 99 00	..Px.....

2. DHCP 分析:

DHCP 的结构有:

op (1)	htype (1)	hlen (1)	hops (1)
xid (4)			
secs (2)		flags (2)	
ciaddr (4)			
yiaddr (4)			
siaddr (4)			
giaddr (4)			
chaddr (16)			
sname (64)			
file (128)			
options (variable)			

各结构的定义如下:

op, 报文类型, 1表示请求报文, 2表示回应报文。

htype, 硬件地址类型, 1表示10Mb/s的以太网的硬件地址。

hlen, 硬件地址长度, 以太网中该值为6。

hops, 跳数。客户端设置为0, 也能被一个代理服务器设置。

xid, 事务ID, 由客户端选择的一个随机数, 被服务器和客户端用来在它们之间交流请求和响应, 客户端用它对请求和应答进行匹配。该ID由客户端设置并由服务器返回, 为32位整数。

secs, 由客户端填充, 表示从客户端开始获得IP地址或IP地址续借后所使用的秒数。

flags, 标志字段。这个16比特的字段, 目前只有最左边的一个比特有用, 该位为0, 表示单播, 为1表示广播。

ciaddr, 客户端的IP地址。只有客户端是Bound、Renew、Rebinding状态, 并且能响应ARP请求时, 才能被填充。

yiaddr, "你自己的"或客户端的IP地址。

siaddr, 表明DHCP协议流程的下一个阶段要使用的服务器的IP地址。

giaddr, DHCP中继器的IP地址。//注意: 不是地址池中定义的网关

chaddr, 客户端硬件地址。客户端必须设置它的"chaddr"字段。UDP数据包中的以太网帧首部也有该字段, 但通常通过查看UDP数据包来确定以太网帧首部中的该字段获取该值比较困难或者说不可能, 而在UDP协议承载的DHCP报文中设置该字段, 用户进程就可以很容易地获取该值。?sname, 可选的服务器主机名, 该字段是空结尾的字符串, 由服务器填写。

file, 启动文件名, 是一个空结尾的字符串。DHCP Discover报文中是"generic"名字或空字符, DHCP Offer报文中提供有效的目录路径全名。

options, 可选参数域, 格式为"代码+长度+数据"。

3. 数据分析:

之后, 在 cmd 窗口输入ipconfig/release 和ipconfig/renew 指令, 可以看到有如下效果:

```
C:\Users\16062>ipconfig/release

Windows IP 配置

不能在 蓝牙网络连接 上执行任何操作, 它已断开媒体连接。

以太网适配器 以太网 3:

    连接特定的 DNS 后缀 . . . . . :
    本地链接 IPv6 地址. . . . . : fe80::b422:9b6:5d1b:3f7e%7
    默认网关. . . . . :

无线局域网适配器 本地连接* 1:

    媒体状态 . . . . . : 媒体已断开连接
    连接特定的 DNS 后缀 . . . . . :

无线局域网适配器 本地连接* 2:

    媒体状态 . . . . . : 媒体已断开连接
    连接特定的 DNS 后缀 . . . . . :

无线局域网适配器 WLAN:

    连接特定的 DNS 后缀 . . . . . :
    本地链接 IPv6 地址. . . . . : fe80::39c9:40d2:aa3f:3ce3%6
    默认网关. . . . . :

以太网适配器 蓝牙网络连接:

    媒体状态 . . . . . : 媒体已断开连接
    连接特定的 DNS 后缀 . . . . . :
```

```
C:\Users\16062>ipconfig/renew
```

Windows IP 配置

不能在 本地连接* 1 上执行任何操作，它已断开媒体连接。
不能在 本地连接* 2 上执行任何操作，它已断开媒体连接。
不能在 蓝牙网络连接 上执行任何操作，它已断开媒体连接。

以太网适配器 以太网 3:

```
连接特定的 DNS 后缀 . . . . . :  
本地链接 IPv6 地址. . . . . : fe80::b422:9b6:5d1b:3f7e%7  
IPv4 地址 . . . . . : 10.38.25.193  
子网掩码 . . . . . : 255.255.255.255  
默认网关. . . . . :
```

无线局域网适配器 本地连接* 1:

```
媒体状态 . . . . . : 媒体已断开连接  
连接特定的 DNS 后缀 . . . . . :
```

无线局域网适配器 本地连接* 2:

```
媒体状态 . . . . . : 媒体已断开连接  
连接特定的 DNS 后缀 . . . . . :
```

无线局域网适配器 WLAN:

```
连接特定的 DNS 后缀 . . . . . :  
本地链接 IPv6 地址. . . . . : fe80::39c9:40d2:aa3f:3ce3%6  
IPv4 地址 . . . . . : 192.168.0.107  
子网掩码 . . . . . : 255.255.255.0  
默认网关. . . . . : 192.168.0.1
```

以太网适配器 蓝牙网络连接:

```
媒体状态 . . . . . : 媒体已断开连接  
连接特定的 DNS 后缀 . . . . . :
```

正在捕获 WLAN

文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)

udp.port==68

No.	Time	Source	Destination	Protocol	Length	Info
5426	26.554687	192.168.0.107	192.168.0.1	DHCP	342	DHCP Release
5998	39.551250	0.0.0.0	255.255.255.255	DHCP	344	DHCP Discover
6024	40.149241	192.168.0.1	192.168.0.107	DHCP	316	DHCP Offer
6025	40.150333	0.0.0.0	255.255.255.255	DHCP	370	DHCP Request
6029	40.229039	192.168.0.1	192.168.0.107	DHCP	316	DHCP ACK

> Frame 5426: 342 bytes on wire (2736 bits), 342 bytes captured (2736 bits) on interface \

> Ethernet II, Src: LiteonTe_4d:ad:27 (e0:0a:f6:4d:ad:27), Dst: Tp-LinkT_b6:17:67 (68:77:2

> Internet Protocol Version 4, Src: 192.168.0.107, Dst: 192.168.0.1

> User Datagram Protocol, Src Port: 68, Dst Port: 67

> Dynamic Host Configuration Protocol (Release)

0000	68 77 24 b6 17 67 e0 0a f6 4d ad 27 08 00 45 00	hw\$. . g . . M . ' . . E .
0010	01 48 e0 35 00 00 80 11 d7 b2 c0 a8 00 6b c0 a8	. H . 5 k . .
0020	00 01 00 44 00 43 01 34 7b 5d 01 01 06 00 b6 e7	. . . D . C . 4 { }
0030	d2 ca 00 00 00 00 c0 a8 00 6b 00 00 00 00 00 00 k
0040	00 00 00 00 00 00 e0 0a f6 4d ad 27 00 00 00 00 M . '
0050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

WLAN: <live capture in progress> | 分组: 12506 • 已显示: 5 (0.0%) | 配置: Default

此时，通过 Wireshark 就可以看到 DHCP 的四次握手获得 IP 地址，缺省路由

DNS 等参数的过程。

接下来就详细分析四次握手的情况：

第一次握手为discover,用于请求ip地址:

Internet Protocol Version 4, Src: 0.0.0.0, Dst: 255.255.255.255

0100 = Version: 4

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

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

Total Length: 330

Identification: 0xab75 (43893)

> Flags: 0x00

...0 0000 0000 0000 = Fragment Offset: 0

Time to Live: 128

Protocol: UDP (17)

Header Checksum: 0x8e2e [validation disabled]

[Header checksum status: Unverified]

Source Address: 0.0.0.0

Destination Address: 255.255.255.255

0000	ff ff ff ff ff e0 0a f6 4d ad 27 08 00 45 00M.'..E.
0010	01 4a ab 75 00 00 80 11 8e 2e 00 00 00 00 ff ff	.J.u....,.....
0020	ff ff 00 44 00 43 01 36 c7 7f 01 01 06 00 bd 07	...D.C.6.....
0030	1a dd 00 00 00 00 00 00 00 00 00 00 00 00 00
0040	00 00 00 00 00 00 e0 0a f6 4d ad 27 00 00 00 00M.'.....
0050	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0060	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0070	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0080	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0090	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00a0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00b0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00c0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00d0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00e0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00f0	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0100	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0110	00 00 00 00 00 00 63 82 53 63 35 01 01 3d 07 01c.Sc5..=...
0120	e0 0a f6 4d ad 27 32 04 c0 a8 00 6b 0c 0f 4c 41	...M.'2...k..LA
0130	50 54 4f 50 2d 4f 53 38 30 49 44 4d 52 3c 08 4d	PTOP-OS8 0IDMR<M
0140	53 46 54 20 35 2e 30 37 0e 01 03 06 0f 1f 21 2b	SFT 5.07!+
0150	2c 2e 2f 77 79 f9 fc ff	,./wy...

Ethernet II, Src: LiteonTe_4d:ad:27 (e0:0a:f6:4d:ad:27), Dst: Tp-LinkT_b6:17:67 (68:77:24:b6:17:67)

这里的源ip是0.0.0.0,目的ip是255.255.255.255,以广播形式来发送报文,

(e0:0a:f6:4d:ad:27),

是DHCP discover 数据包二层 mac 地址.

第二次握手,第二次握手主要实现的是向 client 提供 IP 地址。

当我们的 DHCP 服务器监听到客户端发出的广播时,他会在剩余空闲的地址池内,找出最前的空置 IP,连同其他的TCP/IP 设定,回发一个 DHCP offer 的数据包。由于客户端一开始没有相应的IP,所以在 discover 数据包内会有 mac 的地址信息,同时还会有一个 XID 编号用来判断该封包。之后 DHCP 服务器就会根据这些资料来返回给客户端。同时 offer 数据包会设定一个使用期限吗,当用户的使用期限超时时,其会自动丢弃该 IP.

<div> Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.107 </div> <div> 0100 = Version: 4 0101 = Header Length: 20 bytes (5) > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT) Total Length: 302 Identification: 0x0000 (0) > Flags: 0x00 ...0 0000 0000 0000 = Fragment Offset: 0 Time to Live: 64 Protocol: UDP (17) Header Checksum: 0xf802 [validation disabled] [Header checksum status: Unverified] Source Address: 192.168.0.1 Destination Address: 192.168.0.107 </div>

```
> Bootp flags: 0x0000 (Unicast)
Client IP address: 0.0.0.0
Your (client) IP address: 192.168.0.107
Next server IP address: 0.0.0.0
Relay agent IP address: 0.0.0.0
Client MAC address: LiteonTe_4d:ad:27 (e0:0a:f6:4d:ad:27)
Client hardware address padding: 0000000000000000000000
Server host name not given
Boot file name not given
Magic cookie: DHCP
```

通过上图我们分析 offer 数据包之后可以得出：客户端的地址为 0.0.0.0，应答信息也为广播应答，通过 discover 数据包之后客户端的地址为 192.168.0.107。其中，Client IP address = 0.0.0.0(0.0.0.0)表示客户机还没有使用该地址。

第三次握手：

- √ Option: (53) DHCP Message Type (Request)
 - Length: 1
 - DHCP: Request (3)
- √ Option: (61) Client identifier
 - Length: 7
 - Hardware type: Ethernet (0x01)
 - Client MAC address: LiteonTe_4d:ad:27 (e0:0a:f6:4d:ad:27)
- √ Option: (50) Requested IP Address (192.168.0.107)
 - Length: 4
 - Requested IP Address: 192.168.0.107
- √ Option: (54) DHCP Server Identifier (192.168.0.1)
 - Length: 4
 - DHCP Server Identifier: 192.168.0.1

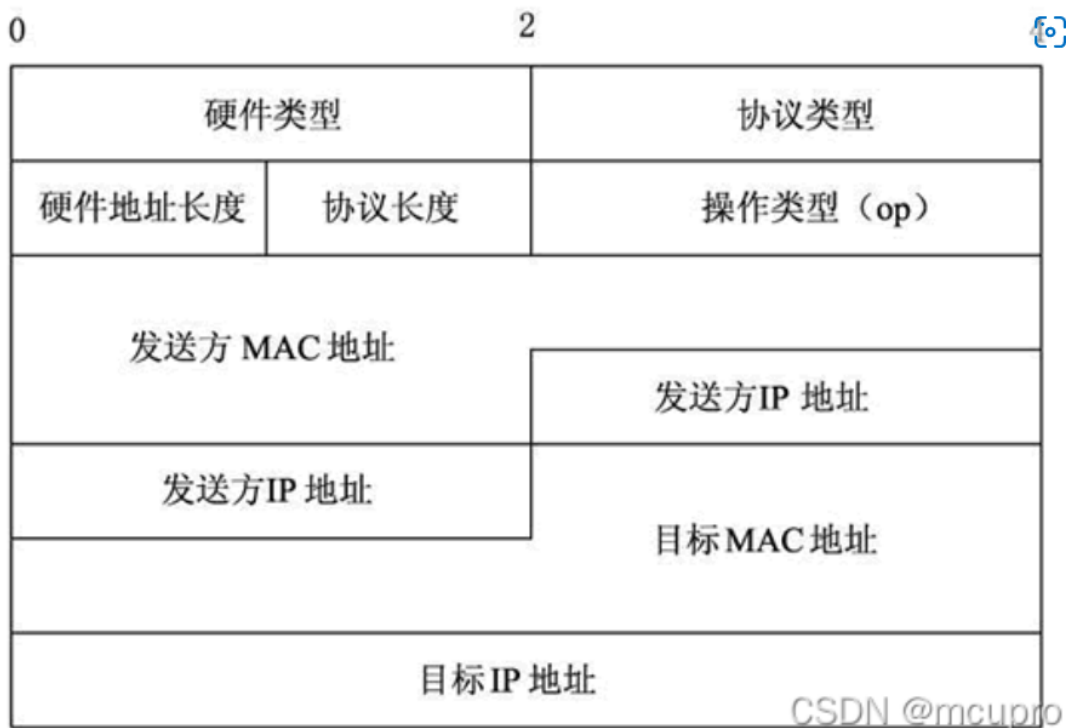
第三次握手主要的功能就是发送一个响应报文。用户会发送 request 报文，以广播形式发送，主要是通过 Message type=Boot Request 来判断的。其中在选项字段中会加入选中的 DHCP Server 的 IP 地址 192.168.0.1 和需要的 IP 地址 192.168.0.107。

第四次握手：

- ✓ Option: (53) DHCP Message Type (ACK)
 - Length: 1
 - DHCP: ACK (5)
- ✓ Option: (54) DHCP Server Identifier (192.168.0.1)
 - Length: 4
 - DHCP Server Identifier: 192.168.0.1
- ✓ Option: (51) IP Address Lease Time
 - Length: 4
 - IP Address Lease Time: (7200s) 2 hours
- ✓ Option: (6) Domain Name Server
 - Length: 4
 - Domain Name Server: 192.168.0.1
- ✓ Option: (1) Subnet Mask (255.255.255.0)
 - Length: 4
 - Subnet Mask: 255.255.255.0
- ✓ Option: (3) Router
 - Length: 4
 - Router: 192.168.0.1
- ✓ Option: (255) End
 - Option End: 255

第四次握手主要是发送一个 ack 报文。其类似于 offer 报文，不同之处在于其是以单播的方式发送的，当服务器接收到 request 报文时，ack 报文就会被发送。这里代表DHCP 服务器端给出了domain name=192.168.0.1，代表此时服务器的域名为 192.168.0.1。

②观察 ARP 和 PING 命令的执行过程：



ARP 报文总长度为 28 字节，MAC 地址长度为 6 字节，IP 地址长度为 4 字节。

其中，每个字段的含义如下。

硬件类型：指明了发送方想知道的硬件接口类型，以太网的值为 1。

协议类型：表示要映射的协议地址类型。它的值为 0x0800，表示 IP 地址。

硬件地址长度和协议长度：分别指出硬件地址和协议的长度，以字节为单位。对于以太网上 IP 地址的 ARP 请求或应答来说，它们的值分别为 6 和 4。

操作类型：用来表示这个报文的类型，ARP 请求为 1，ARP 响应为 2，RARP 请求为 3，RARP 响应为 4。

发送方 MAC 地址：发送方设备的硬件地址。

发送方 IP 地址：发送方设备的 IP 地址。

目标 MAC 地址：接收方设备的硬件地址。

目标 IP 地址：接收方设备的 IP 地址。

如上所示，我们打开 cmd，输入命令“ping 192.168.0.1”，之后我们打开 Wireshark

```
C:\Users\16062>ping 192.168.0.1
```

```
正在 Ping 192.168.0.1 具有 32 字节的数据:
```

```
来自 192.168.0.1 的回复: 字节=32 时间=2ms TTL=64
```

```
来自 192.168.0.1 的回复: 字节=32 时间=3ms TTL=64
```

```
来自 192.168.0.1 的回复: 字节=32 时间=12ms TTL=64
```

```
来自 192.168.0.1 的回复: 字节=32 时间=5ms TTL=64
```

```
192.168.0.1 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 = 2ms, 最长 = 12ms, 平均 = 5ms
```

调好过滤器设置，可以看到如下信息。

205	17.219418	HuaweiTe_c8:7d:ff	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.102
359	29.120199	Tp-LinkT_b6:17:67	Broadcast	ARP	60	Who has 192.168.0.101? Tell 192.168.0.1

- ✓ Ethernet II, Src: HuaweiTe_c8:7d:ff (94:0e:6b:c8:7d:ff), Dst: Broadcast
 - Destination: Broadcast (ff:ff:ff:ff:ff:ff)
 - Source: HuaweiTe_c8:7d:ff (94:0e:6b:c8:7d:ff)
 - Type: ARP (0x0806)
- ✓ Address Resolution Protocol (request)
 - Hardware type: Ethernet (1)
 - Protocol type: IPv4 (0x0800)
 - Hardware size: 6
 - Protocol size: 4
 - Opcode: request (1)
 - Sender MAC address: HuaweiTe_c8:7d:ff (94:0e:6b:c8:7d:ff)
 - Sender IP address: 192.168.0.102
 - Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
 - Target IP address: 192.168.0.1

其中目标 ip 地址的是 192.168.0.1，源 ip 为 192.168.0.102。由上图可知报文长度为 42 字节，MAC 地址为 dc:1b:a1:2d:d8:3b，目标 MAC 地址为 ff:ff:ff:ff:ff:ff。这里的目标地局域网中所有设备都会收到该数据包。

- ✓ Ethernet II, Src: Tp-LinkT_b6:17:67 (68:77:24:b6:17:67), Dst: Broadcast
 - Destination: Broadcast (ff:ff:ff:ff:ff:ff)
 - Source: Tp-LinkT_b6:17:67 (68:77:24:b6:17:67)
 - Type: ARP (0x0806)
 - Padding: 00
- ✓ Address Resolution Protocol (request)
 - Hardware type: Ethernet (1)
 - Protocol type: IPv4 (0x0800)
 - Hardware size: 6
 - Protocol size: 4
 - Opcode: request (1)
 - Sender MAC address: Tp-LinkT_b6:17:67 (68:77:24:b6:17:67)
 - Sender IP address: 192.168.0.1
 - Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
 - Target IP address: 192.168.0.101

这个报文作为一个回应报文，应答包长度为60个字节头6个字节是本地的 mac 地址接着的6个字节

(68:77:24:b6:17:67)

就是对方的mac 地址，这样我们就知道了对

方地址 (00:00:00:00:00:00) 加进缓存表。

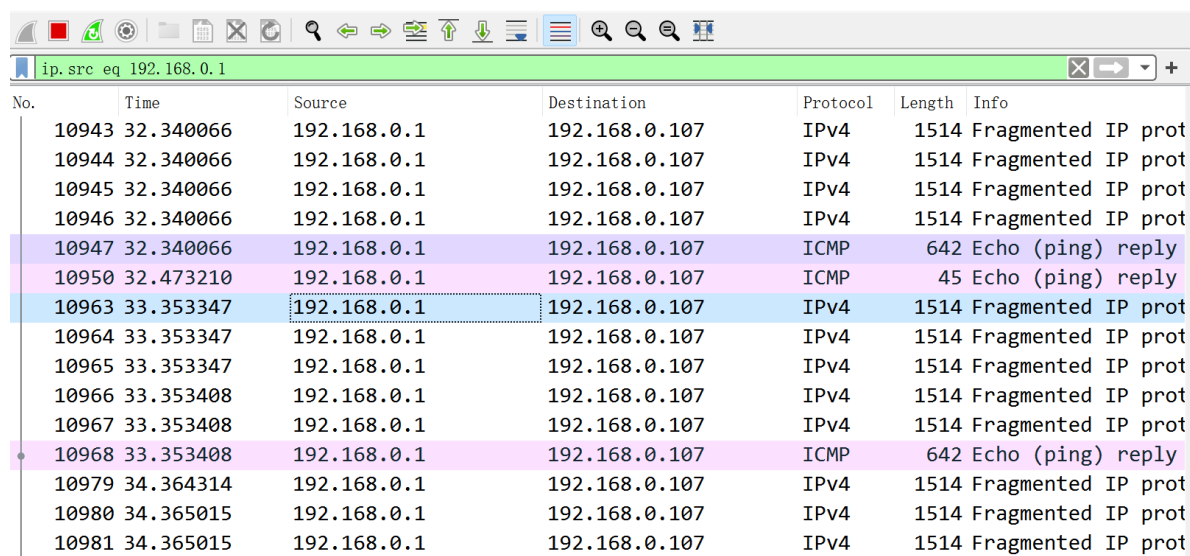
③分析数据分组的分片传输过程：

打开cmd 窗口，输入指令“ping -l 8000 192.168.0.1”：

```
正在 Ping 192.168.0.1 具有 8000 字节的数据：
来自 192.168.0.1 的回复: 字节=8000 时间=2ms TTL=64
来自 192.168.0.1 的回复: 字节=8000 时间=8ms TTL=64
来自 192.168.0.1 的回复: 字节=8000 时间=3ms TTL=64
来自 192.168.0.1 的回复: 字节=8000 时间=6ms TTL=64

192.168.0.1 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 = 2ms, 最长 = 8ms, 平均 = 4ms
```

然后打开Wireshark，将过滤器设置为ip.src eq 192.168.0.1，结果如下：



No.	Time	Source	Destination	Protocol	Length	Info
10943	32.340066	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10944	32.340066	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10945	32.340066	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10946	32.340066	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10947	32.340066	192.168.0.1	192.168.0.107	ICMP	642	Echo (ping) reply
10950	32.473210	192.168.0.1	192.168.0.107	ICMP	45	Echo (ping) reply
10963	33.353347	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10964	33.353347	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10965	33.353347	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10966	33.353408	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10967	33.353408	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10968	33.353408	192.168.0.1	192.168.0.107	ICMP	642	Echo (ping) reply
10979	34.364314	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10980	34.365015	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot
10981	34.365015	192.168.0.1	192.168.0.107	IPv4	1514	Fragmented IP prot

以上我们任选一个分组1进行分析，如下所示：

- Internet Protocol Version 4, Src: 192.168.0.1, Dst: 192.168.0.107
 - 0100 = Version: 4
 - 0101 = Header Length: 20 bytes (5)
 - > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
 - Total Length: 1500
 - Identification: 0x6806 (26630)
 - > Flags: 0x20, More fragments
 - ...0 0000 0000 0000 = Fragment Offset: 0
 - Time to Live: 64
 - Protocol: ICMP (1)
 - Header Checksum: 0x6b5e [validation disabled]
[Header checksum status: Unverified]
 - Source Address: 192.168.0.1
 - Destination Address: 192.168.0.107
 - [\[Reassembled IPv4 in frame: 10968\]](#)

Version：代表版本信息为IPv4。所有的设备运行的协议版本必须相同，否则可能会出现数 据包拒收的情况。长度为 4 字节。

Header length IP：报文头部字段长度为 20 字节。

1105 19.508624 192.168.0.107 39.156.66.18 TCP 66 62243 → 80 [SYN] Seq=0 Win=64240 Le
Internet Protocol Version 4, Src: 192.168.0.107, Dst: 39.156.66.18
0100 = Version: 4
.... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 52
Identification: 0x93e4 (37860)
> Flags: 0x40, Don't fragment
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 128
Protocol: TCP (6)
Header Checksum: 0xc31e [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.0.107
Destination Address: 39.156.66.18

```

Transmission Control Protocol, Src Port: 62243, Dst Port: 80, Seq: 0, Len: 0
  Source Port: 62243
  Destination Port: 80
  [Stream index: 63]
  [Conversation completeness: Complete, NO_DATA (55)]
  [TCP Segment Len: 0]
  Sequence Number: 0 (relative sequence number)
  Sequence Number (raw): 307628967
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 0
  Acknowledgment number (raw): 0
  1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x002 (SYN)
  Window: 64240
  [Calculated window size: 64240]
  Checksum: 0x37ed [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation
  > [Timestamps]

```

客户端发送一个 TCP，标志位为 SYN=1，序号 seq 为 Sequence number=0，

62243 -> 80，代表客户端请求建立连接；

第二次握手：

```

1106 19.540711 39.156.66.18 192.168.0.107 TCP 66 80 -> 62243 [SYN, ACK] Seq=0 Ack=1 W

```

```

Internet Protocol Version 4, Src: 39.156.66.18, Dst: 192.168.0.107
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x04 (DSCP: LE, ECN: Not-ECT)
  Total Length: 52
  Identification: 0x93e4 (37860)
  > Flags: 0x40, Don't fragment
  ...0 0000 0000 0000 = Fragment Offset: 0
  Time to Live: 51
  Protocol: TCP (6)
  Header Checksum: 0x891a [validation disabled]
  [Header checksum status: Unverified]
  Source Address: 39.156.66.18
  Destination Address: 192.168.0.107

```

```

Transmission Control Protocol, Src Port: 80, Dst Port: 62243, Seq: 0, Ack: 1, Len: 0
  Source Port: 80
  Destination Port: 62243
  [Stream index: 63]
  [Conversation completeness: Complete, NO_DATA (55)]
  [TCP Segment Len: 0]
  Sequence Number: 0 (relative sequence number)
  Sequence Number (raw): 2232698205
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 307628968
  1000 .... = Header Length: 32 bytes (8)
  > Flags: 0x012 (SYN, ACK)
  Window: 8192
  [Calculated window size: 8192]
  Checksum: 0x489a [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation
  > [Timestamps]
  > [SEQ/ACK analysis]

```

此时SEQ=0,ACK=1，收到请求后，发送确认数据包返回,同时seq+1;

第三次握手:

1107 19.540813 192.168.0.107 39.156.66.18 TCP 54 62243 → 80 [ACK] Seq=1 Ack=1 Win=26

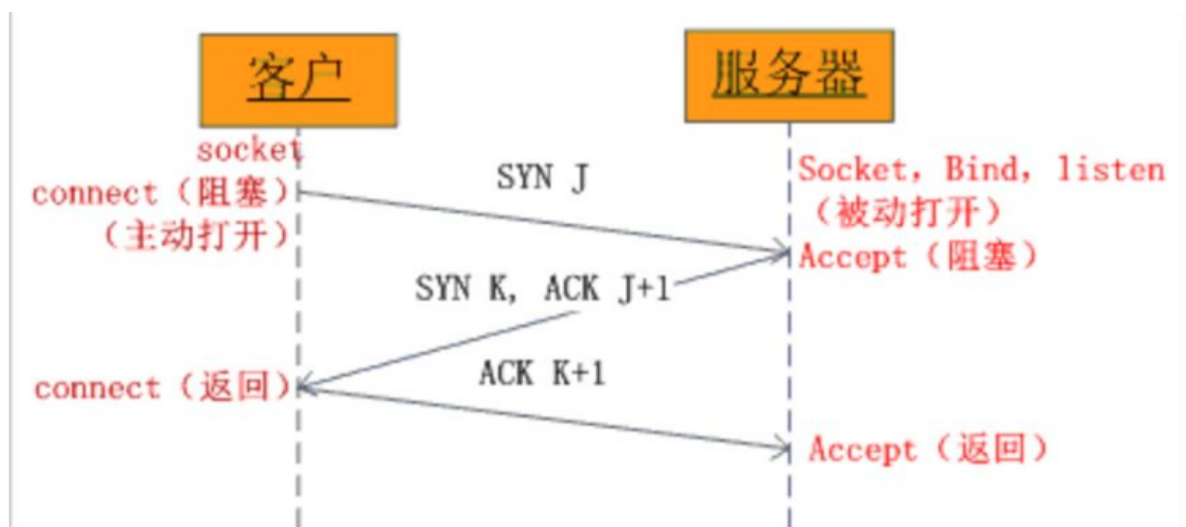
Internet Protocol Version 4, Src: 192.168.0.107, Dst: 39.156.66.18

0100 = Version: 4
.... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
Total Length: 40
Identification: 0x93e5 (37861)
> Flags: 0x40, Don't fragment
...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 128
Protocol: TCP (6)
Header Checksum: 0x3c29 [validation disabled]
[Header checksum status: Unverified]
Source Address: 192.168.0.107
Destination Address: 39.156.66.18

Transmission Control Protocol, Src Port: 62243, Dst Port: 80, Seq: 1, Ack: 1, Len: 0

Source Port: 62243
Destination Port: 80
[Stream index: 63]
[Conversation completeness: Complete, NO_DATA (55)]
[TCP Segment Len: 0]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 307628968
[Next Sequence Number: 1 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)
Acknowledgment number (raw): 2232698206
0101 = Header Length: 20 bytes (5)
> Flags: 0x010 (ACK)
Window: 1028
[Calculated window size: 263168]
[Window size scaling factor: 256]
Checksum: 0xa52a [unverified]
[Checksum Status: Unverified]
Urgent Pointer: 0
> [Timestamps]
> [SEQ/ACK analysis]

此时SEQ=1, ACK=1, 服务器向客户端发送确认信息,客户端收到后确, 一次 TCP 连接就此建立, 可以传送数据了。



断开挥手:

接下来观察 tcp 四次挥手断开链接: TCP 断开连接时, 会有四次挥手过程, 标志位是 FIN,

213	1.329864	39.156.66.18	192.168.0.107	TCP	54	80 → 62208 [FIN, ACK] Seq=98 Ack=161 Win=30336 Len=0
214	1.329916	192.168.0.107	39.156.66.18	TCP	54	62208 → 80 [ACK] Seq=161 Ack=99 Win=262912 Len=0
215	1.329986	192.168.0.107	39.156.66.18	TCP	54	62208 → 80 [FIN, ACK] Seq=161 Ack=99 Win=262912 Len=0
216	1.357682	39.156.66.18	192.168.0.107	TCP	54	80 → 62208 [ACK] Seq=99 Ack=162 Win=30336 Len=0

(和前面的握手通信不是同一次,报文太多,一划过去那次的找不到了)

在封包列表中找到对应位置, 但是因为服务器端在给客户端传回的过程中, 将两个连续发送的包进行了合并, 因此只能抓到 3 个数据包。

第一次挥手:

213	1.329864	39.156.66.18	192.168.0.107	TCP	54	80 → 62208 [FIN, ACK] Seq=98 Ack=161 Win=30336 Len=0
Transmission Control Protocol, Src Port: 80, Dst Port: 62208, Seq: 98, Ack: 161, Len: 0						
Source Port: 80						
Destination Port: 62208						
[Stream index: 18]						
[Conversation completeness: Complete, WITH_DATA (63)]						
[TCP Segment Len: 0]						
Sequence Number: 98 (relative sequence number)						
Sequence Number (raw): 2907929677						
[Next Sequence Number: 99 (relative sequence number)]						
Acknowledgment Number: 161 (relative ack number)						
Acknowledgment number (raw): 3981828111						
0101 = Header Length: 20 bytes (5)						
Flags: 0x011 (FIN, 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						
>1 = Fin: Set						

一方发送 TCP 包给另一方, 用来关闭客户端到服务器的数据传送。将标志位 FIN 和 ACK 置为

1, 序号 seq=X=98, 确认序号 ack=161, 80-> 62208;

第二次挥手:

215	1.329986	192.168.0.107	39.156.66.18	TCP	54	62208 → 80 [FIN, ACK] Seq=161 Ack=99 Win=262912 Len=0
Transmission Control Protocol, Src Port: 62208, Dst Port: 80, Seq: 161, Ack: 99, Len: 0						
Source Port: 62208						
Destination Port: 80						
[Stream index: 18]						
[Conversation completeness: Complete, WITH_DATA (63)]						
[TCP Segment Len: 0]						
Sequence Number: 161 (relative sequence number)						
Sequence Number (raw): 3981828111						
[Next Sequence Number: 162 (relative sequence number)]						
Acknowledgment Number: 99 (relative ack number)						
Acknowledgment number (raw): 2907929678						
0101 = Header Length: 20 bytes (5)						
Flags: 0x011 (FIN, 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						
>1 = Fin: Set						

发回一个 FIN 和 ACK(标志位 FIN=1,ACK=1), 确认序号 ack 为收到的序号加 1, 即 X=X+1=99。

序号 seq 为收到的确认序号=1013, 62208 -> 80;

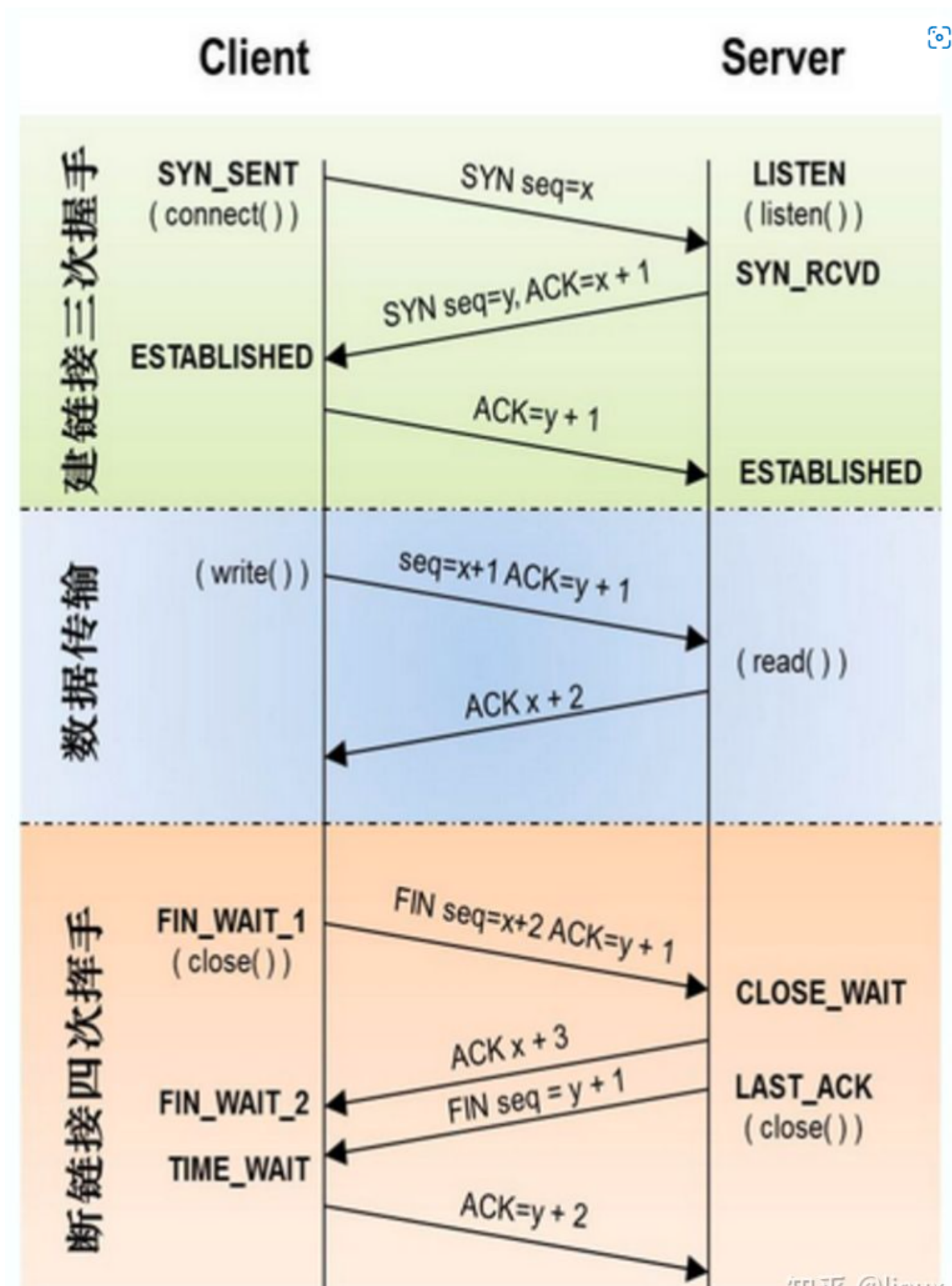
第三次握手:

216.1.357682	39.156.66.18	192.168.0.107	TCP	54 80 → 62208 [ACK] Seq=99 Ack=162 Win=30336 Len=0
Transmission Control Protocol, Src Port: 80, Dst Port: 62208, Seq: 99, Ack: 162, Len: 0				
Source Port: 80				
Destination Port: 62208				
[Stream index: 18]				
[Conversation completeness: Complete, WITH_DATA (63)]				
[TCP Segment Len: 0]				
Sequence Number: 99 (relative sequence number)				
Sequence Number (raw): 2907929678				
[Next Sequence Number: 99 (relative sequence number)]				
Acknowledgment Number: 162 (relative ack number)				
Acknowledgment number (raw): 3981828112				
0101 = Header Length: 20 bytes (5)				
Flags: 0x010 (ACK)				
000. = Reserved: Not set				
...0 = Nonce: Not set				
.... 0... = Congestion Window Reduced (CWR): Not set				
.... .0.. = ECN-Echo: Not set				
.... ..0. = Urgent: Not set				
.... ...1 = Acknowledgment: Set				
.... 0... = Push: Not set				
....0.. = Reset: Not set				
....0. = Syn: Not set				
....0 = Fin: Not set				
[TCP Flags:A.....]				

向服务器发送拆除连接请求, 此时 FIN=1:

发回 ACK 确认(标志位 ACK=1), 确认序号为收到的序号加 1, 即 Y+1=162。序号为收到的确认

序号 X=99, 80 -> 62208。至此, 整个 TCP 通信过程完毕。



⑤实验心得体会：

这此实验大大提高了我对于计算机网络这门课程的兴趣。我花了近一天的时间仔细学习了相关知识，了解了Wireshark软件的使用。之后就开始着手于实验的操作。在一开始我遇到了许多问题，Wireshark使用不熟练、cmd指令格式不清楚、报文的分析不熟悉等等，但好在自己没有放弃，最终克服了这些困难。这些宝贵的经验也会成为我未来学习的基础，让我未来对于计算机网络这门课程的学习更加得心应手。

