北京郵電大學

计算机网络实验

实验二：协议数据的捕获和解析

姓名 张梓良

学院 计算机学院

专业 计算机科学与技术

班级 2021211304

学号 2021212484

任课教师 高占春

2023 年 6 月 9 日

目录

1 实验内容和实验步骤描述

1.1 实验内容

1. 使用 Wireshark 软件捕获在使用 ping 命令时产生的 ICMP 消息。
2. 分析网络层 IP 包头格式，理解各字段的作用，对于分段和校验和进行验证。
3. 使用 Wireshark 软件捕获在使用 ARP 消息，分析其消息格式，理解其工作原理。
4. 使用 Wireshark 捕获 DHCP 消息，分析其消息序列，理解 DHCP 的功能和操作原理。
5. 使用 Wireshark 捕获 TCP 消息，分析 TCP 报文段头格式，理解连接建立和释放的原理，差错控制原理、序号和窗口管理的原理。

1.2 实验步骤

准备工作

1. 下载 Wireshark 软件并了解其功能和使用方法。
2. 确保计算机已经连接到网络。
3. 启动 Wireshark，设置捕获接口（Interface）为本机网卡，选中混杂模式（promiscuousmode）捕获选项，设置合适的捕获过滤器（CaptureFilter）。
4. 开始捕获。

数据捕获

捕获 ICMP 协议数据

1.运行 ping 命令（c>ping www.bupt.edu.cn）。将捕获到的数据保存为文件。

2.使用 Windows 中 ping 命令的-l 选项（c>ping -l 10000 www.bupt.edu.cn），生成大于 8000 字节的 IP 包并发送，捕获后分析其分段传输的包结构。

捕获 DHCP 协议数据

1.使用 ipconfig 命令释放计算机的 IP 地址（c>ipconfig -release）。

2.使用 ipconfig 命令重新申请 IP 地址（c>ipconfig -renew）。此时 wireshark 窗口中可以捕获到完整的 DHCP 地址分配的流程，将捕获到的数据保存为文件。

捕获 ARP 协议数据

释放 IP 地址并重新申请，在 wireshark 窗口中可以捕获到 ARP 请求和响应消息，保存为文件。

捕获 TCP 协议数据

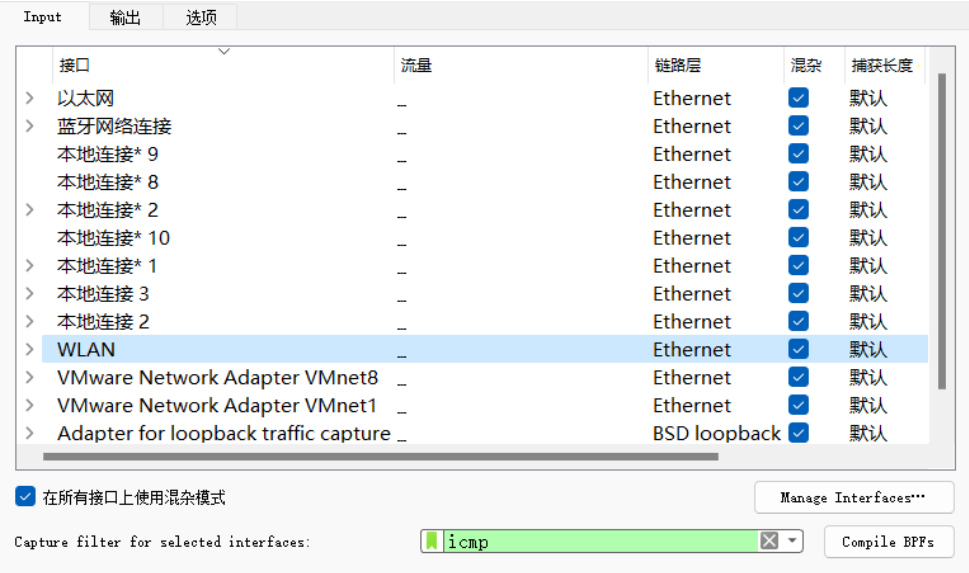
打开浏览器，输入一个页面内容较简单的网页的 URL（www.baidu.com）网页全部显示后关闭浏览器。

1.3 实验环境

- WiresharkVersion 4.0.6
- windowsSdkVersion:10.0.22000.0

2 IP 协议分析

启动 Wireshark，设置捕获接口（Interface）为 WLAN，选中混杂模式（promiscuousmode）捕获选项，设置捕获过滤器为 icmp，开始捕获 IP 数据包。



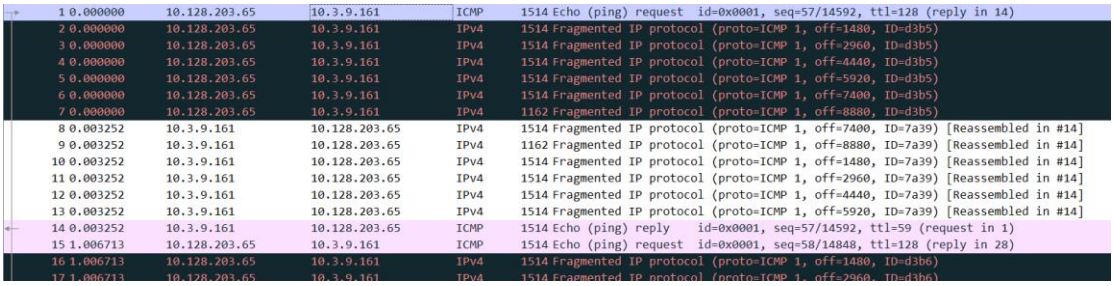
在 cmd 输入 ping -l 10000 www.bupt.edu.cn，向北邮 www 服务器发送长度为 10000 字节的 IP 数据包。

```
C:\Users\SteveZL>ping -l 10000 www.bupt.edu.cn

正在 Ping vn46.bupt.edu.cn [10.3.9.161] 具有 10000 字节的数据:
来自 10.3.9.161 的回复: 字节=10000 时间=5ms TTL=59
来自 10.3.9.161 的回复: 字节=10000 时间=3ms TTL=59
来自 10.3.9.161 的回复: 字节=10000 时间=6ms TTL=59
来自 10.3.9.161 的回复: 字节=10000 时间=5ms TTL=59

10.3.9.161 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 = 3ms, 最长 = 6ms, 平均 = 4ms
```

Wireshark 捕获到 56 个 IP 数据包，其中 8 个是承载 ICMP 消息的 IP 数据包：

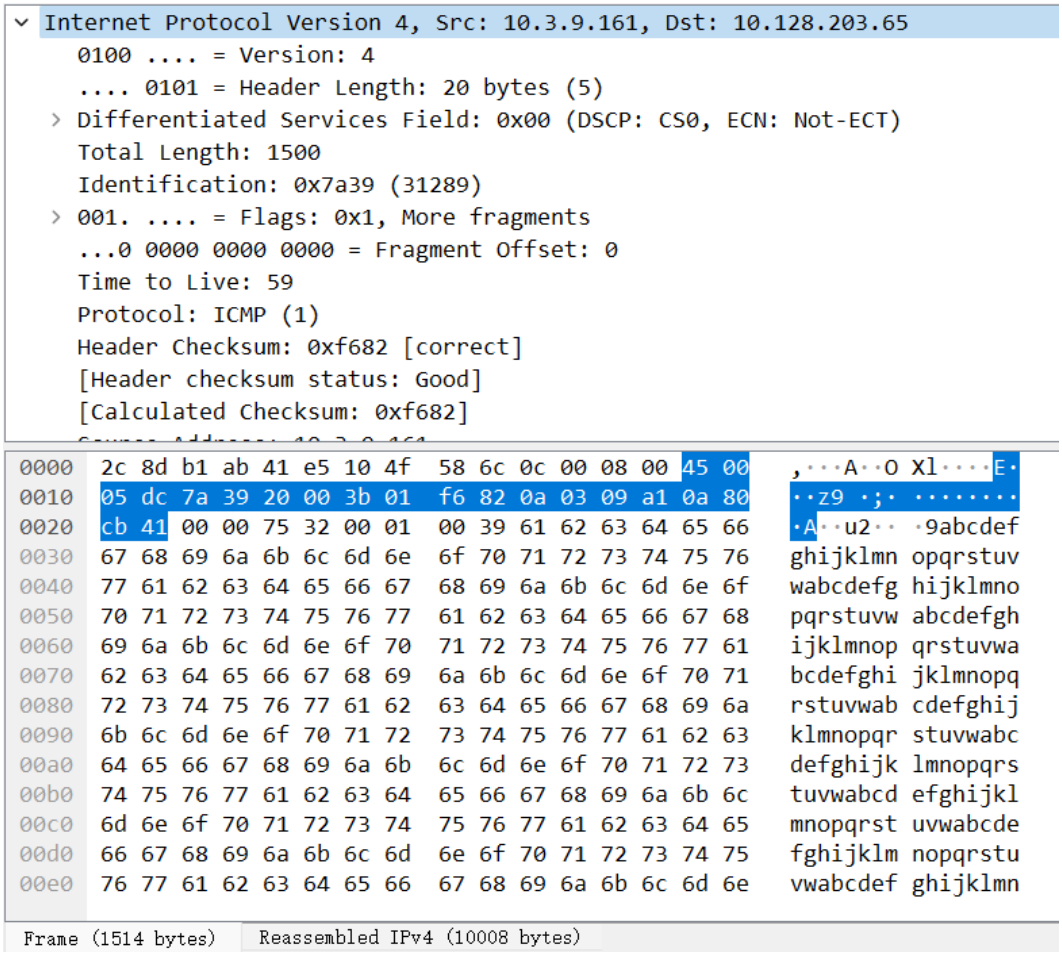


1	0.000000	10.128.203.65	10.3.9.161	ICMP	1514 Echo (ping) request id=0x0001, seq=57/14592, ttl=128 (reply in 14)
2	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=d3b5)
3	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=d3b5)
4	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=4440, ID=d3b5)
5	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=5920, ID=d3b5)
6	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=7400, ID=d3b5)
7	0.000000	10.128.203.65	10.3.9.161	IPv4	1162 Fragmented IP protocol (proto=ICMP 1, off=8880, ID=d3b5)
8	0.003252	10.3.9.161	10.128.203.65	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=7400, ID=7a39) [Reassembled in #14]
9	0.003252	10.3.9.161	10.128.203.65	IPv4	1162 Fragmented IP protocol (proto=ICMP 1, off=8880, ID=7a39) [Reassembled in #14]
10	0.003252	10.3.9.161	10.128.203.65	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=7a39) [Reassembled in #14]
11	0.003252	10.3.9.161	10.128.203.65	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=7a39) [Reassembled in #14]
12	0.003252	10.3.9.161	10.128.203.65	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=4440, ID=7a39) [Reassembled in #14]
13	0.003252	10.3.9.161	10.128.203.65	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=5920, ID=7a39) [Reassembled in #14]
14	0.003252	10.3.9.161	10.128.203.65	ICMP	1514 Echo (ping) reply id=0x0001, seq=57/14592, ttl=59 (request in 1)
15	1.006713	10.128.203.65	10.3.9.161	ICMP	1514 Echo (ping) request id=0x0001, seq=58/14848, ttl=128 (reply in 28)
16	1.006713	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=d3b6)
17	1.006713	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=d3b6)

1	0.000000	10.128.203.65	10.3.9.161	ICMP	1514 Echo (ping) request id=0x0001, seq=57/14592, ttl=128 (reply in 14)
14	0.003252	10.3.9.161	10.128.203.65	ICMP	1514 Echo (ping) reply id=0x0001, seq=57/14592, ttl=59 (request in 1)
15	1.006713	10.128.203.65	10.3.9.161	ICMP	1514 Echo (ping) request id=0x0001, seq=58/14848, ttl=128 (reply in 28)
28	1.010207	10.3.9.161	10.128.203.65	ICMP	1514 Echo (ping) reply id=0x0001, seq=58/14848, ttl=59 (request in 15)
29	2.020402	10.128.203.65	10.3.9.161	ICMP	1514 Echo (ping) request id=0x0001, seq=59/15104, ttl=128 (reply in 42)
42	2.023219	10.3.9.161	10.128.203.65	ICMP	1514 Echo (ping) reply id=0x0001, seq=59/15104, ttl=59 (request in 29)
43	3.029206	10.128.203.65	10.3.9.161	ICMP	1514 Echo (ping) request id=0x0001, seq=60/15360, ttl=128 (reply in 56)
56	3.032202	10.3.9.161	10.128.203.65	ICMP	1514 Echo (ping) reply id=0x0001, seq=60/15360, ttl=59 (request in 43)

IP 数据包包头各字段分析

下面详细分析序号为 14 的 IP 数据包：



```
Internet Protocol Version 4, Src: 10.3.9.161, Dst: 10.128.203.65
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1500
    Identification: 0x7a39 (31289)
  > 001. .... = Flags: 0x1, More fragments
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 59
    Protocol: ICMP (1)
    Header Checksum: 0xf682 [correct]
    [Header checksum status: Good]
    [Calculated Checksum: 0xf682]
```

```
0000  2c 8d b1 ab 41 e5 10 4f  58 6c 0c 00 08 00 45 00   ,···A··O Xl····E·
0010  05 dc 7a 39 20 00 3b 01  f6 82 0a 03 09 a1 0a 80   ··z9 ·;· ········
0020  cb 41 00 00 75 32 00 01  00 39 61 62 63 64 65 66   ·A··u2·· ·9abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69 6a 6b 6c 6d 6e 6f   wabcdefg hijklmno
0050  70 71 72 73 74 75 76 77  61 62 63 64 65 66 67 68   pqrstuvw abcdefgh
0060  69 6a 6b 6c 6d 6e 6f 70  71 72 73 74 75 76 77 61   ijklmnop qrstuvwa
0070  62 63 64 65 66 67 68 69  6a 6b 6c 6d 6e 6f 70 71   bcdefghi jklmnopq
0080  72 73 74 75 76 77 61 62  63 64 65 66 67 68 69 6a   rstuvwab cdefghij
0090  6b 6c 6d 6e 6f 70 71 72  73 74 75 76 77 61 62 63   klmnopqr stuvwabc
00a0  64 65 66 67 68 69 6a 6b  6c 6d 6e 6f 70 71 72 73   defghijk lmnopqrs
00b0  74 75 76 77 61 62 63 64  65 66 67 68 69 6a 6b 6c   tuvwabcd efghijkl
00c0  6d 6e 6f 70 71 72 73 74  75 76 77 61 62 63 64 65   mnopqrst uvwabcde
00d0  66 67 68 69 6a 6b 6c 6d  6e 6f 70 71 72 73 74 75   fghijklm nopqrstu
00e0  76 77 61 62 63 64 65 66  67 68 69 6a 6b 6c 6d 6e   vwabcdef ghijklmn
```

Frame (1514 bytes) Reassembled IPv4 (10008 bytes)

其头部各字段分析表如下：

字段（位数）	报文（默认 16 进制）	内容
Version(4)	4	版本：IPv4
IHL(4)	5	头部长度：20 字节
DifferentiatedServices	00	DSCP（区分服务信

（8）		息）：000000B（默认） 显式拥塞通知（ECN）： 00B（否）
TotalLength(16)	05 dc	数据包总长度：1500 字节（以太网）
Identification(16)	7a 39	数据包标识： 31289
Flags(3)	001B	（第一位 0 是 Reservedbit） Don’tFragment：0（可以分段） MoreFragments：1（有更多的段）
FragmentOffset(13)	0000000000000B	分段偏移量：0
TTL(8)	3b	生存期：59 跳
Protocol(8)	01	协议：ICMP
HeaderCheckSum(16)	f6 82	头部校验和：f682
SourceIPAddress(32)	0a 03 09 a1	源地址：10.3.9.161
DestimationIPAddress(32)	0a 80 cb 41	目的地 址:10.128.203.65

IP 包头校验和校验原理

发送时的校验和：

1. 把校验和字段清零。
2. 然后对首部每 16 位(2 字节)进行二进制反码求和,反码求和的意思是先对每 16 位求和，再将得到的和的高 16 位（不够则补 0）加上低 16 位的结果转为反码。
3. 把得到的结果存入校验和字段中。

接收时的校验和

1. 对首部每 16 位(2 字节)进行二进制反码求和（包括校验和字段）。
2. 检查计算出的结果是否等于 0x0000。等于说明数据正确，否则丢弃。

验证上述序号为 14 的 IP 包头的校验和：

上述 IP 包头为：45 00 05 dc 7a 39 20 00 3b 01 f6 82 0a 03 09 a1 0a 80 cb 41。

对首部每 16 位(2 字节)进行二进制反码求和得：

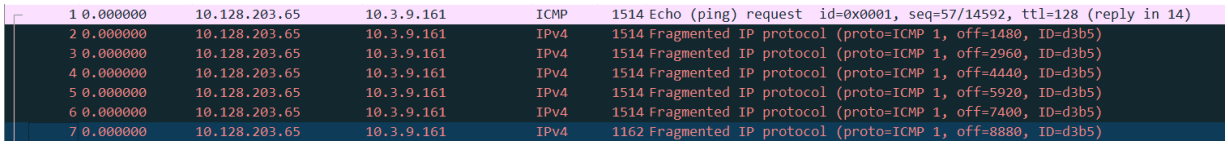
4500+05dc+7a39+2000+3b01+f682+0a03+09a1+0a80+cb41=2fffd，

0002+fffd=ffff，~ffff=0000。

说明 IP 包头正确。

IP 包分段原理

下面以序号 1-7 的 IP 包来讨论 IP 包分段原理。



1	0.000000	10.128.203.65	10.3.9.161	ICMP	1514 Echo (ping) request id=0x0001, seq=57/14592, ttl=128 (reply in 14)
2	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=1480, ID=d3b5)
3	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=2960, ID=d3b5)
4	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=4440, ID=d3b5)
5	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=5920, ID=d3b5)
6	0.000000	10.128.203.65	10.3.9.161	IPv4	1514 Fragmented IP protocol (proto=ICMP 1, off=7400, ID=d3b5)
7	0.000000	10.128.203.65	10.3.9.161	IPv4	1162 Fragmented IP protocol (proto=ICMP 1, off=8880, ID=d3b5)

它们的 MF 标志位以及分段偏移量如下表所示：

序号	MF 标志位	分段偏移量
1	1	0
2	1	1480
3	1	2960
4	1	4440
5	1	5920
6	1	7400
7	0	8880

由于我们选用的接口 WLAN 的链路层为以太网。以太网数据链路层的 MTU 为 1500 字节，除去 IP 头之后剩余 1480 字节，正好是 8 的倍数，因此长度为 10000 字节的数据被拆分成 7 个分段，前 6 个的长度为 1480 字节（其中第一个包中包含 8 个字节的 ICMP 报文头部），最后一个的长度为 1128 字节，分段偏移量表示该分段在原数据段中的位置，通过 MF 标志位为 0 确定数据的结束。

3 ICMP 协议分析

在 cmd 输入 ping www.bupt.edu.cn。

```
C:\Users\SteveZL>ping www.bupt.edu.cn

正在 Ping vn46.bupt.edu.cn [10.3.9.161] 具有 32 字节的数据:
来自 10.3.9.161 的回复: 字节=32 时间=2ms TTL=59
来自 10.3.9.161 的回复: 字节=32 时间=4ms TTL=59
来自 10.3.9.161 的回复: 字节=32 时间=2ms TTL=59
来自 10.3.9.161 的回复: 字节=32 时间=3ms TTL=59

10.3.9.161 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 = 4ms, 平均 = 2ms
```

Wireshark 捕获到 8 个承载 ICMP 消息的 IP 数据包：

1	0.000000	10.128.203.65	10.3.9.161	ICMP	74	Echo (ping) request id=0x0001, seq=53/13568, ttl=128 (reply in 2)
2	0.002163	10.3.9.161	10.128.203.65	ICMP	74	Echo (ping) reply id=0x0001, seq=53/13568, ttl=59 (request in 1)
3	1.009857	10.128.203.65	10.3.9.161	ICMP	74	Echo (ping) request id=0x0001, seq=54/13824, ttl=128 (reply in 4)
4	1.014405	10.3.9.161	10.128.203.65	ICMP	74	Echo (ping) reply id=0x0001, seq=54/13824, ttl=59 (request in 3)
5	2.014642	10.128.203.65	10.3.9.161	ICMP	74	Echo (ping) request id=0x0001, seq=55/14080, ttl=128 (reply in 6)
6	2.017395	10.3.9.161	10.128.203.65	ICMP	74	Echo (ping) reply id=0x0001, seq=55/14080, ttl=59 (request in 5)
7	3.018598	10.128.203.65	10.3.9.161	ICMP	74	Echo (ping) request id=0x0001, seq=56/14336, ttl=128 (reply in 8)
8	3.022384	10.3.9.161	10.128.203.65	ICMP	74	Echo (ping) reply id=0x0001, seq=56/14336, ttl=59 (request in 7)

ICMP 包格式

下面详细分析序号为 1 的 IP 数据包中 ICMP 报文格式：

```
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x4d26 [correct]
    [Checksum Status: Good]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence Number (BE): 53 (0x0035)
    Sequence Number (LE): 13568 (0x3500)
    [Response frame: 2]
  > Data (32 bytes)
```

```
0000  10 4f 58 6c 0c 00 2c 8d  b1 ab 41 e5 08 00 45 00   ·OXl··,· ··A···E·
0010  00 3c d3 b1 00 00 80 01  00 00 0a 80 cb 41 0a 03   ·<······ ·····A··
0020  09 a1 08 00 4d 26 00 01  00 35 61 62 63 64 65 66   ····M&·· ·5abcdef
0030  67 68 69 6a 6b 6c 6d 6e  6f 70 71 72 73 74 75 76   ghijklmn opqrstuv
0040  77 61 62 63 64 65 66 67  68 69                    wabcdefg hi
```

分析如下表所示：

字段（字节数）	报文（默认 16 进制）	内容
Type(1)	08	ICMP 类型为 8，说明这是一个 ICMP Echo 请求报文
Code(1)	00	表示是请求
Checksum(2)	4d 26	校验和为：4d26
Identifier(2)	00 01	标识符，用于区分不同

		进程 ping 消息 在 linux 下是大端 （0x0001）为 1; 在 windows 下是小端（0x0100）为 256
Sequence Number(2)	00 35	ping 请求序列号，在 linux 下是大端 （0x0035）为 53; 在 windows 下是小端（0x3500）为 13568

各字段功能

- Type：占一字节，标识 ICMP 报文的类型，目前已定义了 14 种，从类型值来看 ICMP 报文可以分为两大类。第一类是取值为 1~127 的差错报文，第 2 类是取值 128 以上的信息报文。
- Code：占一字节，标识对应 ICMP 报文的代码。它与类型字段一起共同标识了 ICMP 报文的详细类型。
- Checksum：这是对包括 ICMP 报文数据部分在内的整个 ICMP 数据报的校验和，以检验报文在传输过程中是否出现了差错。其计算方法与 IP 报头中的校验和计算方法一致。
- Identifier：占两字节，用于标识本 ICMP 进程，但仅适用于回显请求和应答 ICMP 报文，对于目标不可达 ICMP 报文和超时 ICMP 报文等，该字段的值为 0。
- Sequence Number：占两个字节，用于匹配回显应答和回显请求。具体来说，标识符确定该报文的发送者，而序列号则关联对应的请求报文和应答报文。

经查阅资料，Type 和 Code 字段的具体组合如下表所示

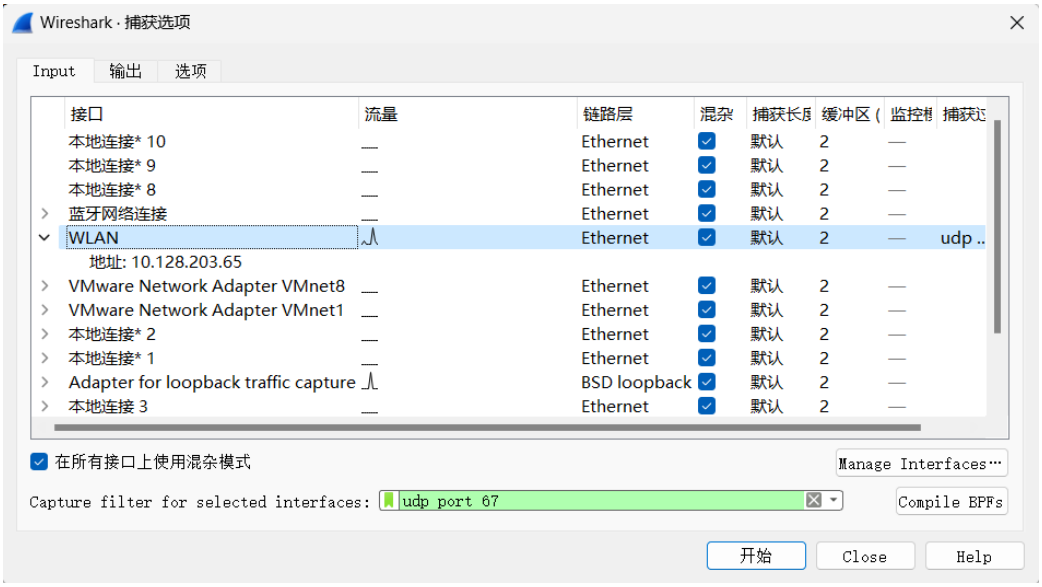
Type	Code	描述	查询/差错
0-Echo 响应	0	Echo 响应报文	查询
3-目的不可达	0	目标网络不可达报文	差错
	1	目标主机不可达报文	差错
	2	目标协议不可达报文	差错
	3	目标端口不可达报文	差错
	4	要求分段并设置 DF flag 标志报文	差错
	5	源路由失败报文	差错

Type	Code	描述	查询/差错
	6	未知的目标网络报文	差错
	7	未知的目标主机报文	差错
	8	源主机隔离报文	差错
	9	禁止访问的网络报文	差错
	10	禁止访问的主机报文	差错
	11	对特定的 TOS 网络不可达报文	差错
	12	对特定的 TOS 主机不可达报文	差错
	13	由于过滤 网络流量被禁止报文	差错
	14	主机越权报文	差错
	15	优先权终止生效报文	差错
5-重定向	0	重定向网络报文	差错
	1	重定向主机报文	差错
	2	基于 TOS 的网络重定向报文	差错
	3	基于 TOS 的主机重定向报文	差错
8-Echo 请求	0	Echo 请求报文	查询
9-路由器通告	0	路由通告报文	查询
10-路由器请求	0	路由器的发现/选择/请求报文	查询
11-ICMP 超时	0	TTL 超时报文	差错
	1	分片重组超时报文	差错
12-参数问题	0	IP 报首部参数错误报文	差错

Type	Code	描述	查询/差错
	1	丢失必要选项报文	差错
	2	不支持的长度报文	差错
13-时间戳请求	0	时间戳请求报文	查询
14-时间戳应答	0	时间戳应答报文	查询
15-信息请求	0	信息请求报文	查询
16-信息应答	0	信息应答报文	查询

4 DHCP 协议分析

启动 Wireshark，设置捕获接口（Interface）为 WLAN，选中混杂模式（promiscuousmode）捕获选项，设置捕获过滤器为 udp port 67，开始捕获 IP 数据包。



在 cmd 输入 ipconfig -release 和 ipconfig -renew。

```
C:\Users\SteveZL>ipconfig -release
```

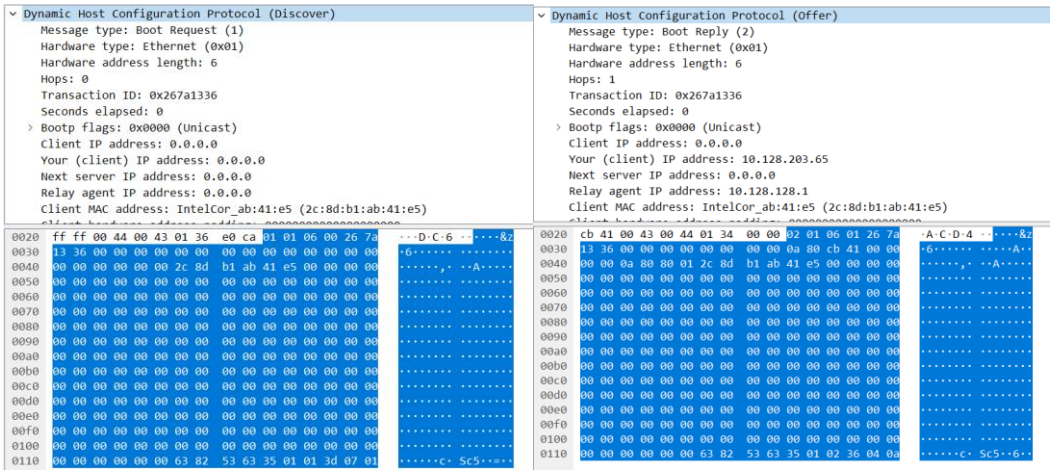
```
C:\Users\SteveZL>ipconfig -renew
```

Wireshark 捕获到 5 个 DHCP 消息。

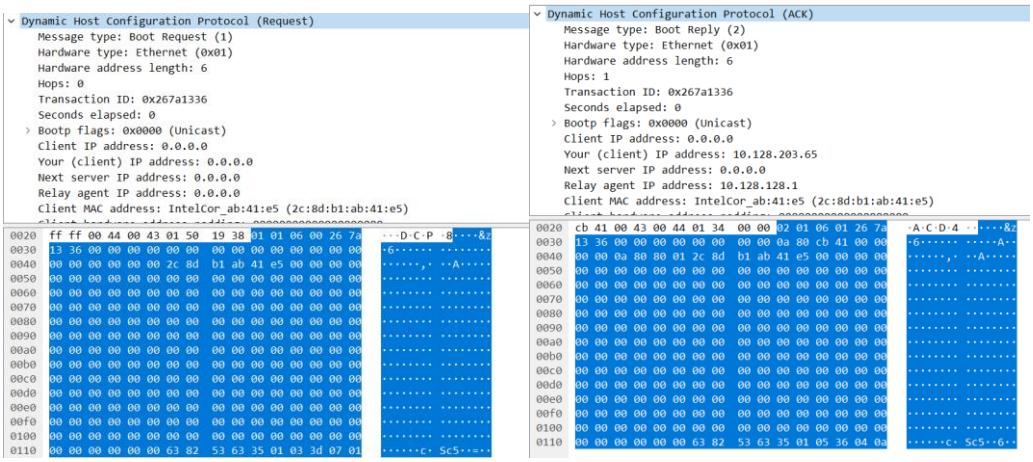
1	0.000000	10.128.203.65	10.3.9.2	DHCP	342 DHCP Release - Transaction ID 0x6ed1e6b5
2	29.887175	0.0.0.0	255.255.255.255	DHCP	344 DHCP Discover - Transaction ID 0x267a1336
3	30.891817	10.3.9.2	10.128.203.65	DHCP	342 DHCP Offer - Transaction ID 0x267a1336
4	30.892559	0.0.0.0	255.255.255.255	DHCP	370 DHCP Request - Transaction ID 0x267a1336
5	30.907621	10.3.9.2	10.128.203.65	DHCP	342 DHCP ACK - Transaction ID 0x267a1336

其中序号 2-4 的 DHCP 消息 Transaction ID 相同，为一次地址分配过程，下面详细分析其通信流程：

DHCP Discover 和 DHCP Offer：



DHCP Request 和 DHCP ACK:



各字段分析

下面对 DHCP ACK 消息各字段进行详细分析：

字段（字节数）	报文（默认 16 进制）	内容
OP (1)	02	报文的操作类型：应答报文
HTYPE (1)	01	DHCP 客户端的 MAC 地址类型:以太网
HLEN (1)	06	DHCP 客户端的 MAC 地址长度：6 个字节
HOPS (1)	01	DHCP 报文经过的 DHCP 中

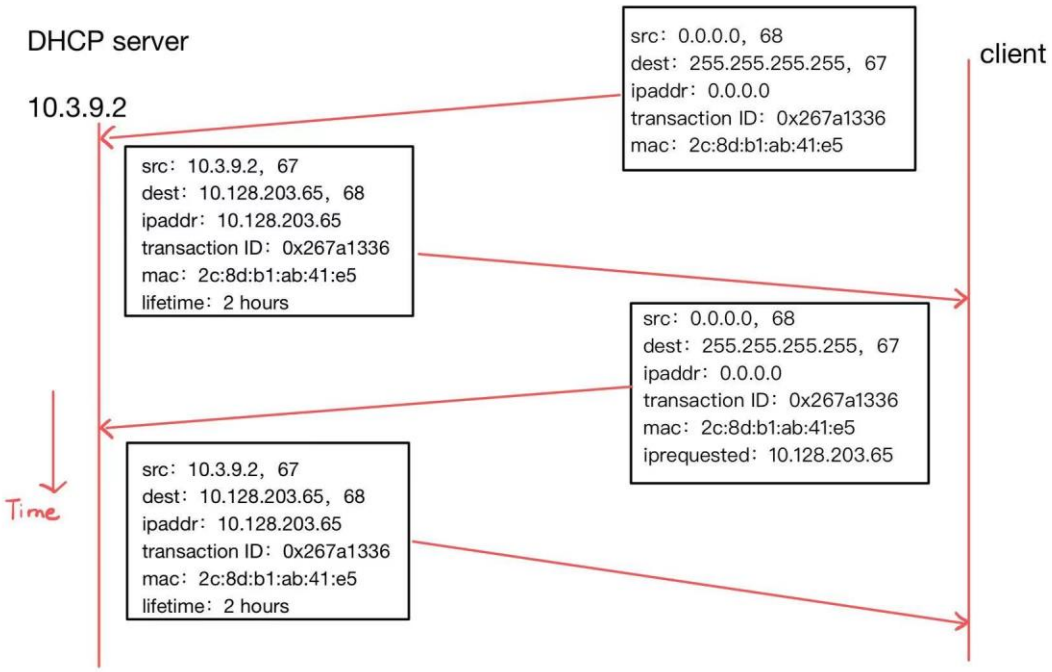
		继的数目：1
XID (4)	26 7a 13 36	请求标识，用来标识一次 IP 地址请求过程。在一次请求中所有报文的 XID 都是一样的。
SECS (2)	00 00	DHCP 客户端从获取到 IP 地址或者续约过程开始到现在所消耗的时间：0s
FLAGS (2)	00 00	标志位，只使用第 1 比特位：单播
CIADDR (4)	00 00 00 00	DHCP 客户端的 IP 地址：0.0.0.0
YIADDR (4)	0a 80 cb 41	DHCP 服务器分配给客户端的 IP 地址：10.128.203.65
SIADDR (4)	00 00 00 00	下一个为 DHCP 客户端分配 IP 地址等信息的 DHCP 服务器 IP 地址：0.0.0.0（DHCP 服务器未知）
GIADDR (4)	0a 80 80 01	DHCP 客户端发出请求报文后经过的第一个 DHCP 中继的 IP 地址：10.128.128.1
CHADDR (16)	2c 8d b1 ab 41 e5…	DHCP 客户端的 MAC 地址：2c:8d:b1:ab:41:e5（后接 10 个字节的 0 的填充）
SNAME (64)	全 0	为 DHCP 客户端分配 IP 地址的 DHCP 服务器名称：未给出
FILE (128)	全 0	DHCP 服务器为 DHCP 客户端指定的启动配置文件名称及路径信息：未给出
magic cookie(4)	63 82 53 63	可选字段的格式：DHCP
OPTION (3)	35 01 05	DHCP 消息类型：ACK
OPTION (6)	36 04 0a 03 09 02	DHCP 服务器 ID:10.3.9.2
OPTION (6)	33 04 00 00 1c 20	IP 地址有效租约时间：2hours
OPTION (6)	01 04 ff ff 80 00	子网掩码：255.255.128.0

OPTION (6)	03 04 0a 80 80 01	默认网关路由器地址：10.128.128.1
OPTION (10)	06 08 0a…	域名服务器地址
OPTION (1)	ff	选项字段结束

可以发现存在 DHCP 中继，且中继路由器的 IP 地址为：10.128.128.1。同时 DHCP 可以配置 IP 地址、子网掩码、DNS 服务器地址、默认网关地址等参数。

消息序列图

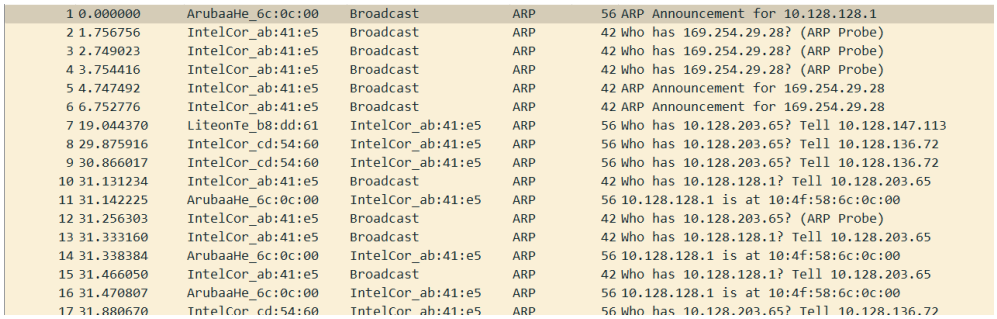
DHCP 地址分配过程的消息序列图如下所示：



1. 客户端连入局域网后，向局域网内发送广播，发送一个 DHCP Discover 信息，报文内容中有本机的 MAC 地址，并在 OPTION 字段中附带一个请求参数的列表。
2. 服务器为客户端分配 IP 地址，并向之前的 MAC 地址发送 DHCP Offer 分组，分组内容中有分配的 IP 地址，分组的 OPTION 字段中附带服务器标识符、IP 有效租约时长、子网掩码等信息。
3. 客户端再次发送广播，发送一个 DHCP Request 信息，在 OPTIONS 字段中指明请求的 IP 地址和服务器标识符。
4. 服务器向客户端发送 DHCP ACK 信息进行确认。

5 ARP 协议分析

同 DHCP 消息的捕获操作类似，在 Wireshark 中捕获到 32 个 ARP 消息。



1	0.000000	ArubaaHe_6c:0c:00	Broadcast	ARP	56 ARP Announcement for 10.128.128.1
2	1.756756	IntelCor_ab:41:e5	Broadcast	ARP	42 Who has 169.254.29.28? (ARP Probe)
3	2.749023	IntelCor_ab:41:e5	Broadcast	ARP	42 Who has 169.254.29.28? (ARP Probe)
4	3.754416	IntelCor_ab:41:e5	Broadcast	ARP	42 Who has 169.254.29.28? (ARP Probe)
5	4.747492	IntelCor_ab:41:e5	Broadcast	ARP	42 ARP Announcement for 169.254.29.28
6	6.752776	IntelCor_ab:41:e5	Broadcast	ARP	42 ARP Announcement for 169.254.29.28
7	19.044370	LiteonTe_b8:dd:61	IntelCor_ab:41:e5	ARP	56 Who has 10.128.203.65? Tell 10.128.147.113
8	29.875916	IntelCor_cd:54:60	IntelCor_ab:41:e5	ARP	56 Who has 10.128.203.65? Tell 10.128.136.72
9	30.866017	IntelCor_cd:54:60	IntelCor_ab:41:e5	ARP	56 Who has 10.128.203.65? Tell 10.128.136.72
10	31.131234	IntelCor_ab:41:e5	Broadcast	ARP	42 Who has 10.128.128.1? Tell 10.128.203.65
11	31.142225	ArubaaHe_6c:0c:00	IntelCor_ab:41:e5	ARP	56 10.128.128.1 is at 10:4f:58:6c:0c:00
12	31.256303	IntelCor_ab:41:e5	Broadcast	ARP	42 Who has 10.128.203.65? (ARP Probe)
13	31.333160	IntelCor_ab:41:e5	Broadcast	ARP	42 Who has 10.128.128.1? Tell 10.128.203.65
14	31.338384	ArubaaHe_6c:0c:00	IntelCor_ab:41:e5	ARP	56 10.128.128.1 is at 10:4f:58:6c:0c:00
15	31.466050	IntelCor_ab:41:e5	Broadcast	ARP	42 Who has 10.128.128.1? Tell 10.128.203.65
16	31.470807	ArubaaHe_6c:0c:00	IntelCor_ab:41:e5	ARP	56 10.128.128.1 is at 10:4f:58:6c:0c:00
17	31.880670	IntelCor_cd:54:60	IntelCor_ab:41:e5	ARP	56 Who has 10.128.203.65? Tell 10.128.136.72

下面对序号为 11 的 ARP 消息进行分析：

```
Address Resolution Protocol (reply)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: reply (2)
    Sender MAC address: ArubaaHe_6c:0c:00 (10:4f:58:6c:0c:00)
    Sender IP address: 10.128.128.1
    Target MAC address: IntelCor_ab:41:e5 (2c:8d:b1:ab:41:e5)
    Target IP address: 10.128.203.65
```

```
0000  2c 8d b1 ab 41 e5 10 4f  58 6c 0c 00 08 06 00 01   ,···A··O Xl··••··
0010  08 00 06 04 00 02 10 4f  58 6c 0c 00 0a 80 80 01   ·•••••·O Xl••••••
0020  2c 8d b1 ab 41 e5 0a 80  cb 41 db b4 8a 15 00 00   ,···A··· ·A••••••
0030  00 0f 08 00 12 0b 00 00                           ••••••••
```

各字段分析

该 ARP 包的各字段分析如下表：

字段（字节数）	报文（默认 16 进制）	内容
Hardware type(2)	0001	硬件类型：以太网
Protocol type(2)	0800	协议类型：IPv4
Hardware size(1)	06	硬件地址长度：6
Protocol size(1)	04	协议地址长度：4
Opcode(2)	0002	ARP 消息类型：reply
Sender MAC address (6)	104f586c0c00	发送方 MAC 地址：10:4f:58:6c:0c:00
Sender IP address(4)	0a808001	发送方 IP 地址：10.128.128.1
Target MAC address (6)	2c8db1ab41e5	接收方 MAC 地址：2c:8d:b1:ab:41:e5
Target IP address(6)	0a80cb41	接收方 IP 地址：10.128.203.65

ARP 的工作流程

下面通过序号为 1、2、7、11 的 ARP 消息对 ARP 工作流程进行分析。

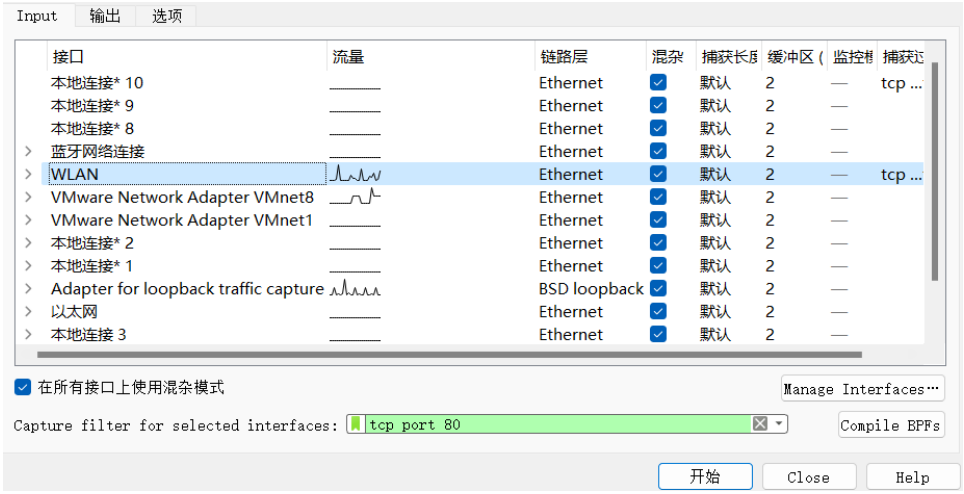
1. 序号为 1 的 ARP 消息是一个 ARP Announcement（ARP 声明），通过该 ARP 消

息可以在该局域网中声明这个 IP 地址，其发送地址被设置为本机 IP，其他主机在收到该消息时，可以用消息中的发送方 IP 地址和发送方 MAC 地址在 ARP cache 中建立映射关系。

2. 序号为 2 的 ARP 消息是一个 ARP Probe（ARP 探测），通过该 ARP 消息可以在局域网中探测 IP 地址是否已经被使用，将接收方 IP 地址设置为想要探测的 IP 地址即可进行探测，若没有回复，则可以为本机分配该 IP 地址。
3. 序号为 7 的 ARP 消息是一个 ARP Request（ARP 请求），通过该 ARP 消息可以在局域网中请求一个特定的 IP 地址所对应的 MAC 地址，并要求目标主机通过单播将该信息发送给自己。
4. 序号为 11 的 ARP 消息是一个 ARP Reply（ARP 回复），通过该 ARP 消息可以将本机的 IP 地址与 MAC 地址的映射关系传递给发起相应 ARP Request 的主机。例如此处是将 IP 地址为 10.128.128.1 的主机的地址映射关系告诉了 IP 地址为 10.128.203.65 的主机。

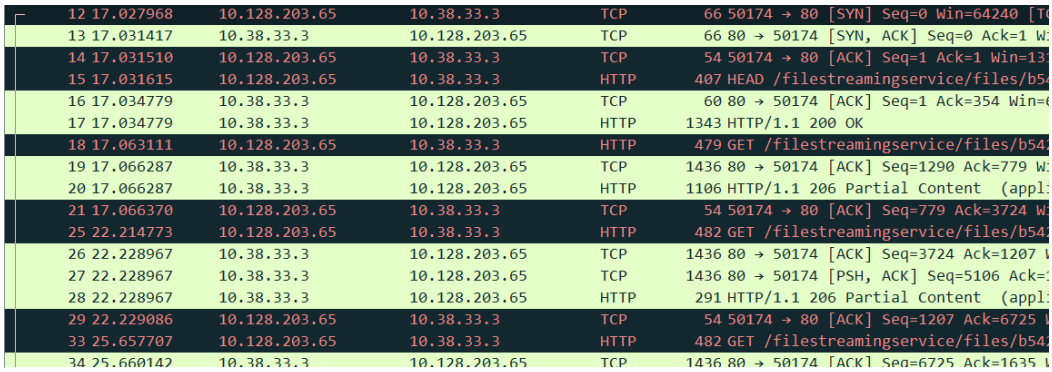
6 TCP 协议分析

启动 Wireshark，设置捕获接口（Interface）为 WLAN，选中混杂模式（promiscuousmode）捕获选项，设置捕获过滤器为 tcp port 80，开始捕获 IP 数据包。



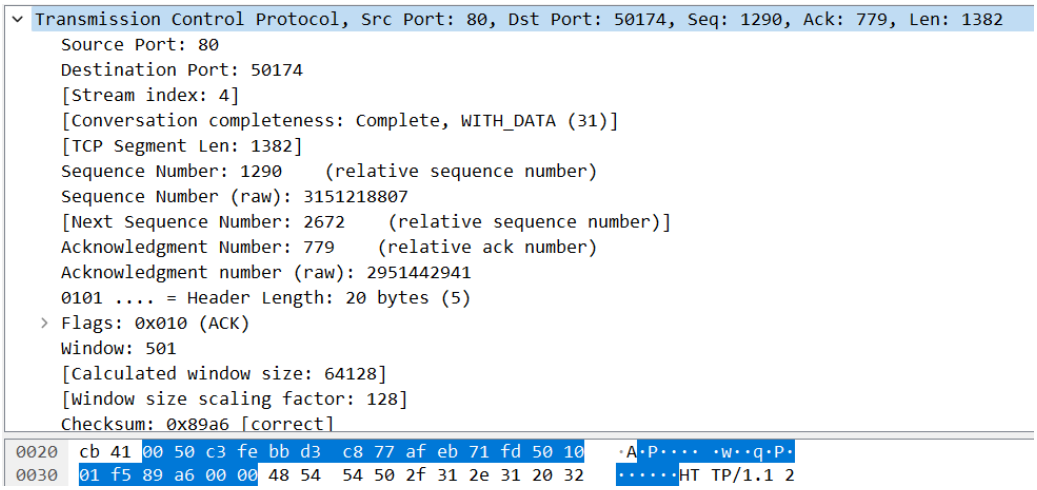
打开浏览器，输入一个页面内容较简单的网页的 URL（www.baidu.com）网页全部显示后关闭浏览器。

Wireshark 捕获到数个 TCP 报文，通过条件 ip.src eq 10.38.33.3 or ip.dst eq 10.38.33.3 进行筛选。



12	17.027968	10.128.203.65	10.38.33.3	TCP	66 50174 → 80 [SYN] Seq=0 Win=64240 [T
13	17.031417	10.38.33.3	10.128.203.65	TCP	66 80 → 50174 [SYN, ACK] Seq=0 Ack=1 W
14	17.031510	10.128.203.65	10.38.33.3	TCP	54 50174 → 80 [ACK] Seq=1 Ack=1 Win=13
15	17.031615	10.128.203.65	10.38.33.3	HTTP	407 HEAD /filestreamingservice/files/b5
16	17.034779	10.38.33.3	10.128.203.65	TCP	60 80 → 50174 [ACK] Seq=1 Ack=354 Win=
17	17.034779	10.38.33.3	10.128.203.65	HTTP	1343 HTTP/1.1 200 OK
18	17.063111	10.128.203.65	10.38.33.3	HTTP	479 GET /filestreamingservice/files/b54
19	17.066287	10.38.33.3	10.128.203.65	TCP	1436 80 → 50174 [ACK] Seq=1290 Ack=779 W
20	17.066287	10.38.33.3	10.128.203.65	HTTP	1106 HTTP/1.1 206 Partial Content (appl
21	17.066370	10.128.203.65	10.38.33.3	TCP	54 50174 → 80 [ACK] Seq=779 Ack=3724 W
25	22.214773	10.128.203.65	10.38.33.3	HTTP	482 GET /filestreamingservice/files/b54
26	22.228967	10.38.33.3	10.128.203.65	TCP	1436 80 → 50174 [ACK] Seq=3724 Ack=1207
27	22.228967	10.38.33.3	10.128.203.65	TCP	1436 80 → 50174 [PSH, ACK] Seq=5106 Ack=
28	22.228967	10.38.33.3	10.128.203.65	HTTP	291 HTTP/1.1 206 Partial Content (appl
29	22.229086	10.128.203.65	10.38.33.3	TCP	54 50174 → 80 [ACK] Seq=1207 Ack=6725
33	25.657707	10.128.203.65	10.38.33.3	HTTP	482 GET /filestreamingservice/files/b54
34	25.660142	10.38.33.3	10.128.203.65	TCP	1436 80 → 50174 [ACK] Seq=6725 Ack=1635

下面对序号为 19 的 IP 数据包进行详细分析：



```
Transmission Control Protocol, Src Port: 80, Dst Port: 50174, Seq: 1290, Ack: 779, Len: 1382
    Source Port: 80
    Destination Port: 50174
    [Stream index: 4]
    [Conversation completeness: Complete, WITH_DATA (31)]
    [TCP Segment Len: 1382]
    Sequence Number: 1290    (relative sequence number)
    Sequence Number (raw): 3151218807
    [Next Sequence Number: 2672    (relative sequence number)]
    Acknowledgment Number: 779    (relative ack number)
    Acknowledgment number (raw): 2951442941
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
    Window: 501
    [Calculated window size: 64128]
    [Window size scaling factor: 128]
    Checksum: 0x89a6 [correct]
0020  cb 41 00 50 c3 fe bb d3  c8 77 af eb 71 fd 50 10   ·A·P···· ·w··q·P·
0030  01 f5 89 a6 00 00 48 54  54 50 2f 31 2e 31 20 32   ·····HT TP/1.1 2
```

TCP 报文头各字段分析

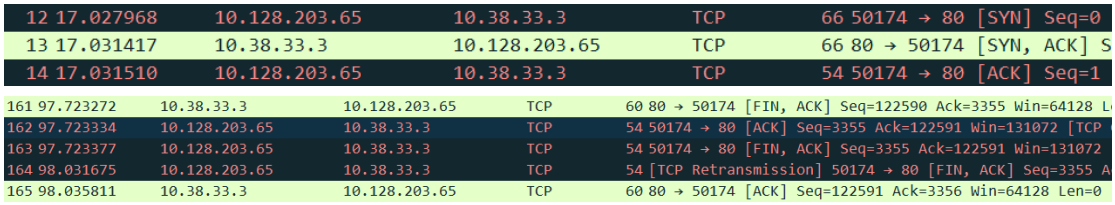
该 IP 包的 TCP 头的各字段分析如下表：

字段（字节数）	报文（默认 16 进制）	内容
Source Port(2)	00 50	源端口：80
Destination Port(2)	c3 fe	目的端口：50174
Sequence Number(4)	bb d3 c8 77	序列号：1290 (relative sequence number)；3151218807(raw)
Acknowledgment Number(4)	af eb 71 fd	确认号：779 (relative ack number)；2951442941(raw)
Header Length(4bits)	0101B	TCP 头长度：20 bytes
Flags(9bits)	50 10	（前 3 位未使用） AEC：精确 ECN 0 CWR：拥塞窗口减小 0 ECE：拥塞信息回响 0 URG：紧急标志 0

		ACK：启用 ACK 标志 1 PSH：推送标志 0 RST：复位标志 0 SYN：建立连接标志 0 FIN：释放连接标志 0
Window Size(2)	01 f5	窗口大小：501*128（scaling factor）=64128
Checksum(2)	89 a6	校验和：0x89a6
Urgent Pointer(2)	00 00	紧急指针：0x0000（正的偏移量，和序号字段的值相加表示最后一个紧急数据的下一字节的序号）

连接建立和连接释放

首先找到连接建立和连接释放的报文：



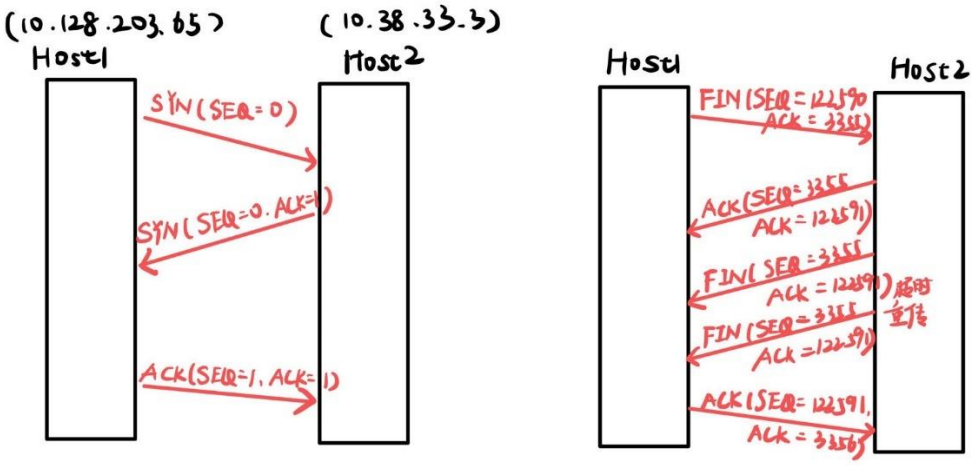
连接建立时的标志位和序号如下表所示：

	标志位	SEQ	ACK
1	SYN	0（2951442162）	0（0）
2	SYN ACK	0（3151217517）	1（2951442163）
3	ACK	1（2951442163）	1（3151217518）

释放建立时的标志位和序号如下表所示：

	标志位	SEQ（此处只给出相对序号）	ACK
1	FIN ACK	122590	3355
2	ACK	3355	122591
3	FIN ACK	3355	122591
4	FIN ACK	3355	122591
5	ACK	122591	3356

连接建立和连接释放的消息序列图为：



数据传输过程

挑选序号 84-88 的数据报作为数据传输过程分析：

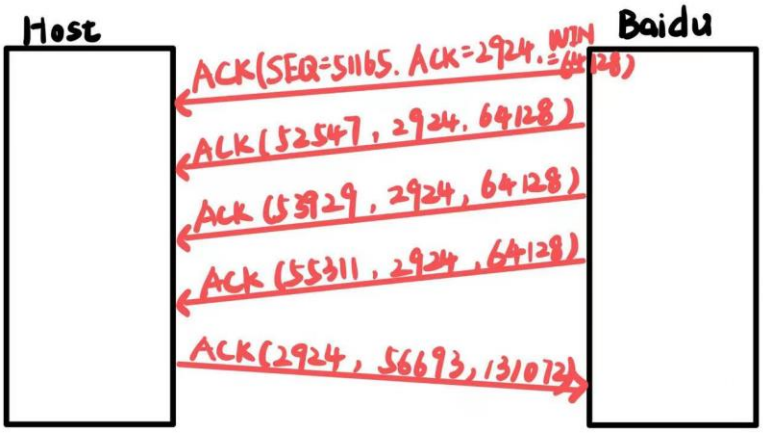
84	31.705015	10.38.33.3	10.128.203.65	TCP	1436	80 → 50174 [ACK] Seq=51165 Ack=2924 Win=64128
85	31.705015	10.38.33.3	10.128.203.65	TCP	1436	80 → 50174 [ACK] Seq=52547 Ack=2924 Win=64128
86	31.705015	10.38.33.3	10.128.203.65	TCP	1436	80 → 50174 [ACK] Seq=53929 Ack=2924 Win=64128
87	31.705015	10.38.33.3	10.128.203.65	TCP	1436	80 → 50174 [ACK] Seq=55311 Ack=2924 Win=64128
88	31.705090	10.128.203.65	10.38.33.3	TCP	54	50174 → 80 [ACK] Seq=2924 Ack=56693 Win=131072

具体分析如下表：

序号	SEQ	ACK	标志位	WIN	LEN(TCP payload)
84	51165	2924	ACK	64128	1382
85	52547	2924	ACK	64128	1382
86	53929	2924	ACK	64128	1382
87	55311	2924	ACK	64128	1382
88	2924	56693	ACK	131072	0

84-87 号数据报是远程 www 服务器向本机传输的数据，数据报的 TCP payload 部分长度始终为 1382 字节，由此可以推出 MSS 应为 1382 字节。可以发现采取了累计确认的机制，当 84-87 号数据报都成功传输到本机后，本地主机才对 87 号数据报进行了确认，确认数据报的 TCP payload 长度为 0。

数据传输过程的消息序列图如下：



7 实验结论和实验心得

在完成本实验的过程中，我遇到了一些问题并成功解决，总结如下：

1. Wireshark 捕获的数据包比较多，需要根据需求筛选，并对筛选条件进行优化，避免捕获到的数据包过多导致分析效率低下。
2. 在分析 IP 包头格式时，需要理解各字段的作用，如版本号、头部长度、服务类型、标识、标志位、偏移量等，并手动计算校验和来验证数据完整性。
3. 在分析 ARP 消息格式和工作原理时，需要了解 ARP 协议的功能和操作过程，如地址解析、请求和响应等，并能够通过 Wireshark 提供的详细信息进行分析和验证。
4. 在分析 DHCP 协议时，需要了解 DHCP 的功能和操作原理，如地址分配、掩码设置、网关设置等，并能够分析 DHCP 消息序列，了解客户端和服务器之间的交互过程。
5. 在分析 TCP 消息时，需要了解 TCP 的连接建立和释放过程，差错控制原理、序号和窗口管理原理等，对于 TCP 报文段头格式的各字段，需要深入理解并手动计算校验和以验证数据完整性。

通过本次实验，我深入理解了分层体系结构和 TCP/IP 协议栈代表协议的重要性，掌握了 IP、TCP、UDP、ICMP、ARP 和 DHCP 协议的要点，并能够使用 Wireshark 工具进行数据包分析和验证。同时，我也认识到网络安全至关重要，需要加强对网络攻击的防范和应对能力。