

北京邮电大学

实 验 报 告



题目： 计算机网络实验二

姓 名 杨晨

学 院 计算机学院

专 业 计算机科学与技术

班 级 2021211304

学 号 2021212171

任课教师 高占春

2023 年 6 月

目 录

第一章 引言	1
1.1 实验任务和内容	1
1.2 实验环境	1
第二章 实验步骤和网络层分组分析	2
2.1 IP 协议分析	2
2.1.1 捕获 IP 数据包	2
2.1.2 IP 数据包头部校验和原理	3
2.1.3 IP 包头分段原理	3
2.2 捕获和分析 ICMP 分组	13
2.2.1 分析 PING 回显响应数据包: www.bupt.edu.cn	13
2.2.2 分析 PING 大负载数据包的回显响应	8
2.3 捕获 DHCP 协议数据	13
2.3.1 使用 ipconfig 命令释放和重新申请 IP 地址	13
2.3.2 DHCP 地址分配过程	19
2.3.3 DHCP 协议分析	20
2.4 捕获 ARP 协议数据	20
2.4.1 使用 ipconfig 命令释放和重新申请 IP 地址	20
2.4.2 ARP 各字段的功能	26
2.5 TCP 协议分析	26
2.5.1 使用浏览器打开和关闭网页, 捕获 TCP 协议数据	26
2.5.2 连接建立	27
2.5.3 数据传输过程	30
2.5.4 数据和应答报文段	33
第三章 实验结论和心得	36

第一章 引言

1.1 实验任务和内容

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，TCP 协议数据。
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。

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。

1.2 实验环境

Windows 10

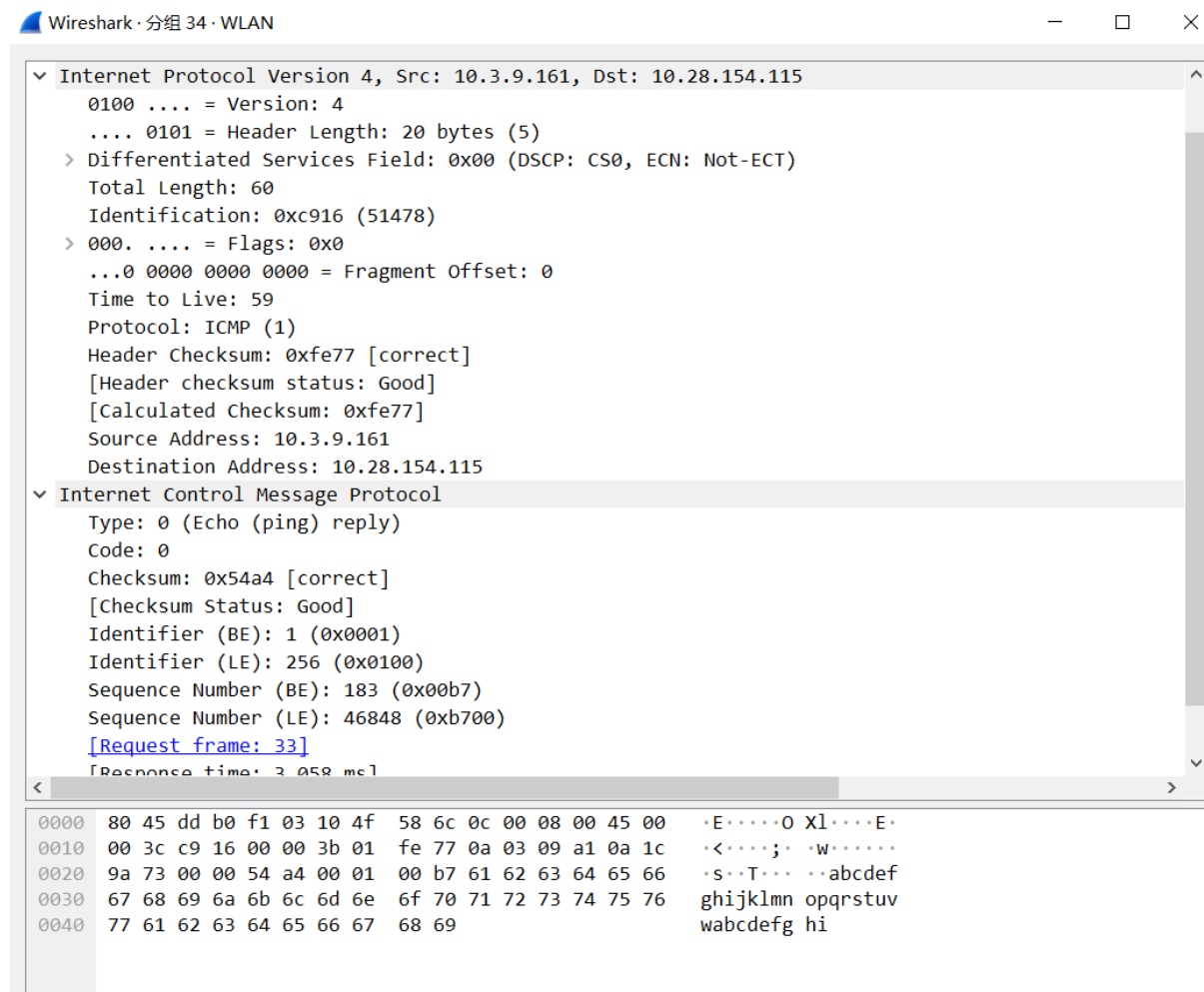
Wireshark Version 4.0.6

第二章 实验步骤和网络层分组分析

2.1 IP 协议分析

2.1.1 捕获 IP 数据包

捕获某个 IP 包



序号为 34 的分组

字段	报文 (16 进制)	内容
包头长度	45	20 字节 (5 * 32 位/8)
服务类型	00	DSCP: CS0, ECN: Not-ECT
总长度	003c	60 字节
标识	c916	51478
标志	00	不分片
片偏移	0000	0
生存周期	3b	59
协议	01	ICMP (1)
头部校验和	fe77	0xfe77 (正确)

源地址	0a03:09a1	10.3.9.161
目的地址	0a1c:9a73	10.28.154.115

序号为 34 的分组详细信息

2.1.2 IP 数据包头部校验和原理

注意，计算校验和时，需要将原始校验和字段设置为 0。

以这个 IP 头部为例，字段如下（每行 4 字节）：

```
4500 003c c916 0000 3b01 fe77 0a03 09a1 0a1c 9a73
```

将所有 16 位字段相加：

```
4500
+ 003c
+ c916
+ 0000
+ 3b01
+ 0000
+ 0a03
+ 09a1
+ 0a1c
+ 9a73
-----
20186
```

处理溢出（将溢出的 1 加到和的低 16 位上）：

```
2 + 0186 = 0188
```

最后，对结果取反：

```
~0188 = FE77
```

经过计算，得到的校验和为 0xFE77。我们得到的校验和为 0xFE77。这个结果与给出的正确校验和 0xFE77 相符

2.1.3 IP 包头分段原理

IP 包分段是将一个较大的 IP 数据报分割成多个更小的数据报段，以便在网络中传输。这种分割方式可以避免在网络中传输过程中因为 MTU 限制而导致的数据包过大无法传输的问题。

在 IP 包分段过程中，一个数据报段的大小是由 MTU（最大传输单元）决定的。如果一个 IP 数据报的大小超过了 MTU，就需要将其分割成多个数据报段，并在每个数据

IPv4 目的	10.28.154.115	10.28.154.115	10.28.154.115	10.28.154.115	10.28.154.115
片偏移	7400	1480	2960	4440	5920
更多片段	无	是	是	是	是
协议	ICMP	ICMP	ICMP	ICMP	ICMP

序号 53~57 的分组

字段	报文 (16 进制)	内容
包头长度	45	20 字节 (5)
服务类型	00	DSCP: CS0, ECN: Not-ECT
总长度	05DC	1500 字节
标识	21F8	8696
标志	2000	More fragments
片偏移	0000	0
生存周期	3B	59
协议	01	ICMP (1)
头部校验和	7FF6	[correct]
源地址	0A0309A1	10.3.9.161
目的地址	0A1C9A73	10.28.154.115

序号为 58 的分组详细信息

Frame 58 是一个分段数据包，与其他分段数据包共同组成一个完整的 ICMP 数据包。这些分段数据包的关系如下：

这个数据包是一个 IPv4 数据包，采用了 ICMP (Internet Control Message Protocol, 互联网控制消息协议) 作为传输协议。包头长度为 20 字节，服务类型为 DSCP: CS0, ECN: Not-ECT，表示这个数据包没有特定的优先级和拥塞通知。总长度为 1500 字节。

标识字段为 8696，标志字段表示这个数据包具有更多片段 (More fragments)，片偏移为 0，表示这是分片序列中的第一个片段。生存周期 (TTL) 为 59，表示这个数据包在网络中的生存时间。

头部校验和为 7FF6，源地址为 10.3.9.161，目的地址为 10.28.154.115。这些信息表明这个数据包从源地址 10.3.9.161 发送至目的地址 10.28.154.115，且经过了校验。

1. Frame 58 是分段数据包序列中的第一个，负载范围为 0-1479 (共 1480 字节)。
2. 其他分段数据包的帧编号和负载范围分别为：
 - Frame 54: 负载范围 1480-2959 (共 1480 字节)
 - Frame 55: 负载范围 2960-4439 (共 1480 字节)
 - Frame 56: 负载范围 4440-5919 (共 1480 字节)

- Frame 57: 负载范围 5920-7399 (共 1480 字节)

- Frame 53: 负载范围 7400-8007 (共 608 字节)

3. 这些分段数据包共有 6 个片段, 经过重组后, 完整的 IPv4 数据包长度为 8008 字节。

根据这些信息, 我们可以得知 Frame 58 与其他分段数据包共同构成了一个更大的 IP 数据包, 这些数据包在传输过程中被分段以满足网络层的最大传输单元 (MTU) 限制。在目的地, 这些分段数据包需要被重新组合以重建完整的 IP 数据包。

2.2 捕获和分析 ICMP 分组

2.2.1 分析 PING 回显响应数据包: www.bupt.edu.cn

选用的 ping 命令为 >ping www.bupt.edu.cn

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

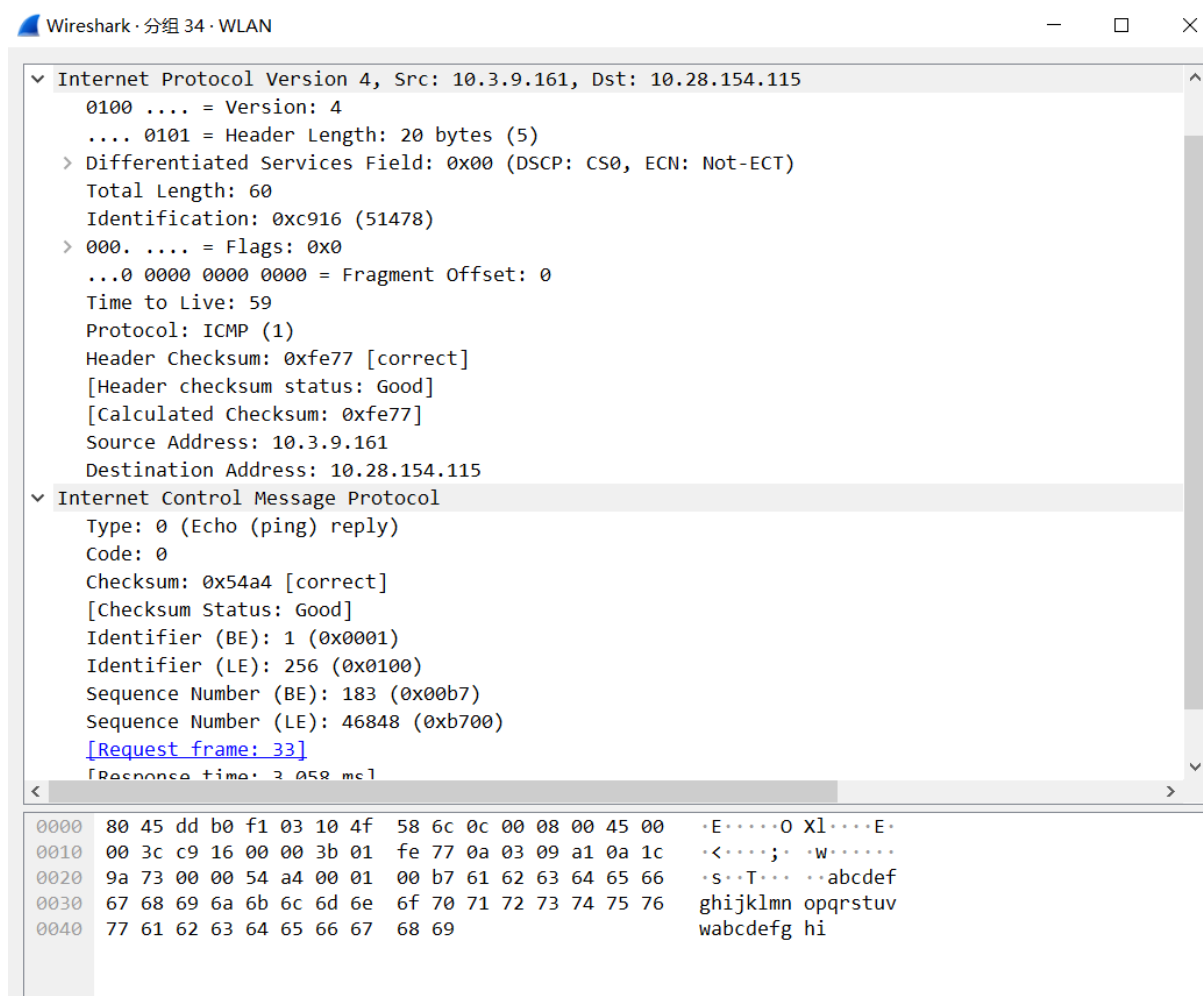
正在 Ping vn46.bupt.edu.cn [10.3.9.161] 具有 32 字节的数据:
来自 10.3.9.161 的回复: 字节=32 时间=2ms TTL=59
来自 10.3.9.161 的回复: 字节=32 时间=1ms TTL=59
来自 10.3.9.161 的回复: 字节=32 时间=3ms TTL=59
来自 10.3.9.161 的回复: 字节=32 时间=3ms TTL=59

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 = 1ms, 最长 = 3ms, 平均 = 2ms
```

cmd 命令端使用 ping 命令

No.	Time	Source	Destination	Protocol	Length	Info
5	1.812563	10.28.154.115	10.3.9.161	ICMP	74	Echo (ping) request id=0x0001,
6	1.814955	10.3.9.161	10.28.154.115	ICMP	74	Echo (ping) reply id=0x0001,
7	2.824906	10.28.154.115	10.3.9.161	ICMP	74	Echo (ping) request id=0x0001,
8	2.826602	10.3.9.161	10.28.154.115	ICMP	74	Echo (ping) reply id=0x0001,
24	3.836915	10.28.154.115	10.3.9.161	ICMP	74	Echo (ping) request id=0x0001,
25	3.840110	10.3.9.161	10.28.154.115	ICMP	74	Echo (ping) reply id=0x0001,
33	4.839035	10.28.154.115	10.3.9.161	ICMP	74	Echo (ping) request id=0x0001,
34	4.842093	10.3.9.161	10.28.154.115	ICMP	74	Echo (ping) reply id=0x0001,

ping 命令产生的 ICMP 分组内容



序号为 34 的分组

字段	报文 (16 进制)	内容
包头长度	45	20 字节 (5 * 32 位/8)
服务类型	00	DSCP: CS0, ECN: Not-ECT
总长度	003c	60 字节
标识	c916	51478
标志	00	不分片
片偏移	0000	0
生存周期	3b	59
协议	01	ICMP (1)
头部校验和	fe77	0xfe77 (正确)
源地址	0a03:09a1	10.3.9.161
目的地址	0a1c:9a73	10.28.154.115

序号为 34 的 ICMP 分组头分析

序号为 34 的 ICMP 分组是一个 ICMP (Internet Control Message Protocol) 数据包,

即互联网控制报文协议。具体来说，这是一个 ICMP Echo 回复，即 Ping 回复。当我们向 www.bupt.edu.cn 发送 Ping 请求时，这个数据包是作为响应被发送回来的。

2.2.2 分析 PING 大负载数据包的回显响应

使用命令 `>ping -l 8000 www.bupt.edu.cn`

向 ip 地址为 10.3.9.161 发送大于 8000 字节的数据包

```
C:\Users\Administrator>ping -l 8000 www.bupt.edu.cn

正在 Ping vn46.bupt.edu.cn [10.3.9.161] 具有 8000 字节的数据:
来自 10.3.9.161 的回复: 字节=8000 时间=3ms TTL=59
来自 10.3.9.161 的回复: 字节=8000 时间=4ms TTL=59
来自 10.3.9.161 的回复: 字节=8000 时间=5ms TTL=59
来自 10.3.9.161 的回复: 字节=8000 时间=7ms 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 = 7ms, 平均 = 4ms
```

用 ping 命令收到的 ICMP 分组回复如图

No.	Time	Source	Destination	Protocol	Length	Info
35	3.028703	10.28.154.115	10.3.9.161	ICMP	1514	Echo (ping) request id=0x0001, s
46	3.037233	10.3.9.161	10.28.154.115	ICMP	1514	Echo (ping) reply id=0x0001, s
47	4.034138	10.28.154.115	10.3.9.161	ICMP	1514	Echo (ping) request id=0x0001, s
58	4.037274	10.3.9.161	10.28.154.115	ICMP	1514	Echo (ping) reply id=0x0001, s
60	5.041579	10.28.154.115	10.3.9.161	ICMP	1514	Echo (ping) request id=0x0001, s
71	5.044773	10.3.9.161	10.28.154.115	ICMP	1514	Echo (ping) reply id=0x0001, s
72	6.050504	10.28.154.115	10.3.9.161	ICMP	1514	Echo (ping) request id=0x0001, s
83	6.055279	10.3.9.161	10.28.154.115	ICMP	1514	Echo (ping) reply id=0x0001, s

用 ping 命令收到的 ICMP 分组回复

53	4.037274	10.3.9.161	10.28.154.115	IPv4	642	Fragmented IP protocol (proto=ICMP 1, c
54	4.037274	10.3.9.161	10.28.154.115	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, c
55	4.037274	10.3.9.161	10.28.154.115	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, c
56	4.037274	10.3.9.161	10.28.154.115	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, c
57	4.037274	10.3.9.161	10.28.154.115	IPv4	1514	Fragmented IP protocol (proto=ICMP 1, c
58	4.037274	10.3.9.161	10.28.154.115	ICMP	1514	Echo (ping) reply id=0x0001, seq=18:

用 ping 命令收到的第一个 ICMP 分组回复

Wireshark · 分组 53 · WLAN

```

> Frame 53: 642 bytes on wire (5136 bits), 642 bytes captured (5136 bits) on interface \Device\NPF_{6...}
> Ethernet II, Src: ArubaaHe_6c:0c:00 (10:4f:58:6c:0c:00), Dst: IntelCor_b0:f1:03 (80:45:dd:b0:f1:03)
  Internet Protocol Version 4, Src: 10.3.9.161, Dst: 10.28.154.115
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 628
    Identification: 0xecdc (60636)
  > 000. .... = Flags: 0x0
    ...0 0011 1001 1101 = Fragment Offset: 7400
    Time to Live: 59
    Protocol: ICMP (1)
    Header Checksum: 0xd4dc [correct]
    [Header checksum status: Good]
    [Calculated Checksum: 0xd4dc]
    Source Address: 10.3.9.161
    Destination Address: 10.28.154.115
    [Reassembled IPv4 in frame: 58]
  Data (608 bytes)
    Data: 6a6b6c6d6e6f70717273747576776162636465666768696a6b6c6d6e6f70717273747576...
    [Length: 608]

```

0000	80 45 dd b0 f1 03 10 4f	58 6c 0c 00 08 00 45 00	·E· · · · · O X1 · · · · · E·
0010	02 74 ec dc 03 9d 3b 01	d4 dc 0a 03 09 a1 0a 1c	·t· · · · · ; · · · · · · ·
0020	9a 73 6a 6b 6c 6d 6e 6f	70 71 72 73 74 75 76 77	·s·j·k·l·m·n·o·p·q·r·s·t·u·v·w·
0030	61 62 63 64 65 66 67 68	69 6a 6b 6c 6d 6e 6f 70	a·b·c·d·e·f·g·h·i·j·k·l·m·n·o·p·
0040	71 72 73 74 75 76 77 61	62 63 64 65 66 67 68 69	q·r·s·t·u·v·w·a·b·c·d·e·f·g·h·i·
0050	6a 6b 6c 6d 6e 6f 70 71	72 73 74 75 76 77 61 62	j·k·l·m·n·o·p·q·r·s·t·u·v·w·a·b·
0060	63 64 65 66 67 68 69 6a	6b 6c 6d 6e 6f 70 71 72	c·d·e·f·g·h·i·j·k·l·m·n·o·p·q·r·
0070	73 74 75 76 77 61 62 63	64 65 66 67 68 69 6a 6b	s·t·u·v·w·a·b·c·d·e·f·g·h·i·j·k·
0080	6c 6d 6e 6f 70 71 72 73	74 75 76 77 61 62 63 64	l·m·n·o·p·q·r·s·t·u·v·w·a·b·c·d·
0090	65 66 67 68 69 6a 6b 6c	6d 6e 6f 70 71 72 73 74	e·f·g·h·i·j·k·l·m·n·o·p·q·r·s·t·
00a0	75 76 77 61 62 63 64 65	66 67 68 69 6a 6b 6c 6d	u·v·w·a·b·c·d·e·f·g·h·i·j·k·l·m·
00b0	6e 6f 70 71 72 73 74 75	76 77 61 62 63 64 65 66	n·o·p·q·r·s·t·u·v·w·a·b·c·d·e·f·

序号为 53 的分组

Wireshark · 分组 54 · WLAN

```

> Frame 54: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NP...
> Ethernet II, Src: ArubaaHe_6c:0c:00 (10:4f:58:6c:0c:00), Dst: IntelCor_b0:f1:03 (80:45:dd:b0:f1:03)
  Internet Protocol Version 4, Src: 10.3.9.161, Dst: 10.28.154.115
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1500
    Identification: 0xecdc (60636)
  > 001. .... = Flags: 0x1, More fragments
    ...0 0000 1011 1001 = Fragment Offset: 1480
    Time to Live: 59
    Protocol: ICMP (1)
    Header Checksum: 0xb458 [correct]
    [Header checksum status: Good]
    [Calculated Checksum: 0xb458]
    Source Address: 10.3.9.161
    Destination Address: 10.28.154.115
    [Reassembled IPv4 in frame: 58]
  Data (1480 bytes)
    Data: 6162636465666768696a6b6c6d6e6f70717273747576776162636465666768696a6b6c6d...
    [Length: 1480]

```

0000	80 45 dd b0 f1 03 10 4f	58 6c 0c 00 08 00 45 00	·E· · · · · O X1 · · · · · E·
0010	05 dc ec dc 20 b9 3b 01	b4 58 0a 03 09 a1 0a 1c	· · · · · ; · · · · · X · · · · ·
0020	9a 73 61 62 63 64 65 66	67 68 69 6a 6b 6c 6d 6e	·s·a·b·c·d·e·f·g·h·i·j·k·l·m·n·
0030	6f 70 71 72 73 74 75 76	77 61 62 63 64 65 66 67	o·p·q·r·s·t·u·v·w·a·b·c·d·e·f·g·
0040	68 69 6a 6b 6c 6d 6e 6f	70 71 72 73 74 75 76 77	h·i·j·k·l·m·n·o·p·q·r·s·t·u·v·w·
0050	61 62 63 64 65 66 67 68	69 6a 6b 6c 6d 6e 6f 70	a·b·c·d·e·f·g·h·i·j·k·l·m·n·o·p·
0060	71 72 73 74 75 76 77 61	62 63 64 65 66 67 68 69	q·r·s·t·u·v·w·a·b·c·d·e·f·g·h·i·
0070	6a 6b 6c 6d 6e 6f 70 71	72 73 74 75 76 77 61 62	j·k·l·m·n·o·p·q·r·s·t·u·v·w·a·b·
0080	63 64 65 66 67 68 69 6a	6b 6c 6d 6e 6f 70 71 72	c·d·e·f·g·h·i·j·k·l·m·n·o·p·q·r·
0090	73 74 75 76 77 61 62 63	64 65 66 67 68 69 6a 6b	s·t·u·v·w·a·b·c·d·e·f·g·h·i·j·k·
00a0	6c 6d 6e 6f 70 71 72 73	74 75 76 77 61 62 63 64	l·m·n·o·p·q·r·s·t·u·v·w·a·b·c·d·
00b0	65 66 67 68 69 6a 6b 6c	6d 6e 6f 70 71 72 73 74	e·f·g·h·i·j·k·l·m·n·o·p·q·r·s·t·

No.: 54 · Time: 4.037274 · Source: 10.3.9.161 · Destination: 10.28.154.115 · Fragmented IP protocol (proto=ICMP 1, offset=1480, ID=ecdc) [Reassembled in #58]

序号为 54 的分组

Wireshark · 分组 55 · WLAN

```
> Frame 55: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF...
> Ethernet II, Src: ArubaaHe_6c:0c:00 (10:4f:58:6c:0c:00), Dst: IntelCor_b0:f1:03 (80:45:dd:b0:f1:03)
> Internet Protocol Version 4, Src: 10.3.9.161, Dst: 10.28.154.115
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 1500
  Identification: 0xecdc (60636)
> 001. .... = Flags: 0x1, More fragments
  ...0 0001 0111 0010 = Fragment Offset: 2960
  Time to Live: 59
  Protocol: ICMP (1)
  Header Checksum: 0xb39f [correct]
  [Header checksum status: Good]
  [Calculated Checksum: 0xb39f]
  Source Address: 10.3.9.161
  Destination Address: 10.28.154.115
  [Reassembled IPv4 in frame: 58]
> Data (1480 bytes)
  Data: 696a6b6c6d6e6f707172737475767776162636465666768696a6b6c6d6e6f707172737475...
  [Length: 1480]
```

0000	80 45 dd b0 f1 03 10 4f	58 6c 0c 00 08 00 45 00	·E·...·O Xl·...·E·
0010	05 dc ec dc 21 72 3b 01	b3 9f 0a 03 09 a1 0a 1c	·...·!r;· ...
0020	9a 73 69 6a 6b 6c 6d 6e	6f 70 71 72 73 74 75 76	·sijklmn opqrstuv
0030	77 61 62 63 64 65 66 67	68 69 6a 6b 6c 6d 6e 6f	wabcde fghijklmno
0040	70 71 72 73 74 75 76 77	61 62 63 64 65 66 67 68	pqrstuvw abcdefgh
0050	69 6a 6b 6c 6d 6e 6f 70	71 72 73 74 75 76 77 61	ijklmnop qrstuvw
0060	62 63 64 65 66 67 68 69	6a 6b 6c 6d 6e 6f 70 71	bcdefghi jklmnopq
0070	72 73 74 75 76 77 61 62	63 64 65 66 67 68 69 6a	rstuvwab cdefghij
0080	6b 6c 6d 6e 6f 70 71 72	73 74 75 76 77 61 62 63	klmnopqr stuvwabc
0090	64 65 66 67 68 69 6a 6b	6c 6d 6e 6f 70 71 72 73	defghijk lmnopqrs
00a0	74 75 76 77 61 62 63 64	65 66 67 68 69 6a 6b 6c	tuvwabcd e fghijkl
00b0	6d 6e 6f 70 71 72 73 74	75 76 77 61 62 63 64 65	mnopqrst uvwabcde

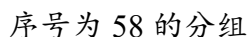
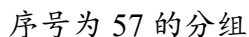
序号为 55 的分组

Wireshark · 分组 56 · WLAN

```
> Frame 56: 1514 bytes on wire (12112 bits), 1514 bytes captured (12112 bits) on interface \Device\NPF...
> Ethernet II, Src: ArubaaHe_6c:0c:00 (10:4f:58:6c:0c:00), Dst: IntelCor_b0:f1:03 (80:45:dd:b0:f1:03)
> Internet Protocol Version 4, Src: 10.3.9.161, Dst: 10.28.154.115
  0100 .... = Version: 4
  .... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
  Total Length: 1500
  Identification: 0xecdc (60636)
> 001. .... = Flags: 0x1, More fragments
  ...0 0010 0010 1011 = Fragment Offset: 4440
  Time to Live: 59
  Protocol: ICMP (1)
  Header Checksum: 0xb2e6 [correct]
  [Header checksum status: Good]
  [Calculated Checksum: 0xb2e6]
  Source Address: 10.3.9.161
  Destination Address: 10.28.154.115
  [Reassembled IPv4 in frame: 58]
> Data (1480 bytes)
  Data: 7172737475767776162636465666768696a6b6c6d6e6f70717273747576777616263646566...
  [Length: 1480]
```

0000	80 45 dd b0 f1 03 10 4f	58 6c 0c 00 08 00 45 00	·E·...·O Xl·...·E·
0010	05 dc ec dc 22 2b 3b 01	b2 e6 0a 03 09 a1 0a 1c	·...·"+;· ...
0020	9a 73 71 72 73 74 75 76	77 61 62 63 64 65 66 67	·sqrstuv wabcde f
0030	68 69 6a 6b 6c 6d 6e 6f	70 71 72 73 74 75 76 77	hijklmno pqrstuvw
0040	61 62 63 64 65 66 67 68	69 6a 6b 6c 6d 6e 6f 70	abcde fghijklmnop
0050	71 72 73 74 75 76 77 61	62 63 64 65 66 67 68 69	qrstuvwab cdefgh
0060	6a 6b 6c 6d 6e 6f 70 71	72 73 74 75 76 77 61 62	ijklmnopq rstuvw
0070	63 64 65 66 67 68 69 6a	6b 6c 6d 6e 6f 70 71 72	cdefghij klmnopqr
0080	73 74 75 76 77 61 62 63	64 65 66 67 68 69 6a 6b	stuvwabc defghijk
0090	6c 6d 6e 6f 70 71 72 73	74 75 76 77 61 62 63 64	lmnopqrs tuvwabcd
00a0	65 66 67 68 69 6a 6b 6c	6d 6e 6f 70 71 72 73 74	efghijkl mnopqrst

序号为 56 的分组



属性	Frame 53	Frame 54	Frame 55	Frame 56	Frame 57
帧编号	53	54	55	56	57
帧大小	628 字节	1500 字节	1500 字节	1500 字节	1500 字节
数据部分	608 字节	1480 字节	1480 字节	1480 字节	1480 字节
IPv4 源	10.3.9.161	10.3.9.161	10.3.9.161	10.3.9.161	10.3.9.161
IPv4 目的	10.28.154.115	10.28.154.115	10.28.154.115	10.28.154.115	10.28.154.115
片偏移	7400	1480	2960	4440	5920
更多片段	无	是	是	是	是
协议	ICMP	ICMP	ICMP	ICMP	ICMP

序号 53~57 的分组

字段	报文 (16 进制)	内容
包头长度	45	20 字节 (5)
服务类型	00	DSCP: CS0, ECN: Not-ECT
总长度	05DC	1500 字节
标识	21F8	8696
标志	2000	More fragments
片偏移	0000	0
生存周期	3B	59
协议	01	ICMP (1)
头部校验和	7FF6	[correct]
源地址	0A0309A1	10.3.9.161
目的地址	0A1C9A73	10.28.154.115

序号为 58 的 IP 分组详细信息

Frame 58 是一个分段数据包，与其他分段数据包共同组成一个完整的 ICMP 数据包。这些分段数据包的关系如下：

这个数据包是一个 IPv4 数据包，采用了 ICMP (Internet Control Message Protocol, 互联网控制消息协议) 作为传输协议。包头长度为 20 字节，服务类型为 DSCP: CS0, ECN: Not-ECT，表示这个数据包没有特定的优先级和拥塞通知。总长度为 1500 字节。

标识字段为 8696，标志字段表示这个数据包具有更多片段 (More fragments)，片偏移为 0，表示这是分片序列中的第一个片段。生存周期 (TTL) 为 59，表示这个数据包

在网络中的生存时间。

头部校验和为 7FF6，源地址为 10.3.9.161，目的地址为 10.28.154.115。这些信息表明这个数据包从源地址 10.3.9.161 发送至目的地址 10.28.154.115，且经过了校验。

1. Frame 58 是分段数据包序列中的第一个，负载范围为 0-1479（共 1480 字节）。
2. 其他分段数据包的帧编号和负载范围分别为：
 - Frame 54: 负载范围 1480-2959（共 1480 字节）
 - Frame 55: 负载范围 2960-4439（共 1480 字节）
 - Frame 56: 负载范围 4440-5919（共 1480 字节）
 - Frame 57: 负载范围 5920-7399（共 1480 字节）
 - Frame 53: 负载范围 7400-8007（共 608 字节）
3. 这些分段数据包共有 6 个片段，经过重组后，完整的 IPv4 数据包长度为 8008 字节。

根据这些信息，我们可以得知 Frame 58 与其他分段数据包共同构成了一个更大的 ICMP 数据包，这些数据包在传输过程中被分段以满足网络层的最大传输单元（MTU）限制。在目的地，这些分段数据包需要被重新组合以重建完整的 ICMP 数据包。

2.3 捕获 DHCP 协议数据

2.3.1 使用 ipconfig 命令释放和重新申请 IP 地址

在命令行窗口中使用命令 >ipconfig -release

命令提示符

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

Windows IP 配置

不能在 以太网 上执行任何操作，它已断开媒体连接。
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 蓝牙网络连接 上执行任何操作，它已断开媒体连接。

未知适配器 本地连接:

    连接特定的 DNS 后缀 . . . . . :
    本地链接 IPv6 地址. . . . . : fe80::c007:8480:8412:836e%15
    IPv4 地址 . . . . . : 2.0.0.1
    子网掩码 . . . . . : 255.255.255.0
    默认网关. . . . . :

以太网适配器 以太网:

    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* 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 . . . . . :
```


用 wireshark 捕获到的数据包

No.	Time	Source	Destination	Protocol	Length	Info
2168	43.607878	10.28.154.115	10.3.9.2	DHCP	342	DHCP Release - Transaction ID 0xb0ea7819

捕获到的 DHCP Release

Wireshark · 分组 2168 · WLAN

Internet Protocol Version 4, Src: 10.28.154.115, Dst: 10.3.9.2

- 0100 = Version: 4
- 0101 = Header Length: 20 bytes (5)
- > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- Total Length: 328
- Identification: 0xd04c (53324)
- > 000. = Flags: 0x0
- ...0 0000 0000 0000 = Fragment Offset: 0
- Time to Live: 128
- Protocol: UDP (17)
- > Header Checksum: 0x0000 incorrect, should be 0xb1c4(may be caused by "IP checksum offload"?)
- > [Expert Info (Error/Checksum): Bad checksum [should be 0xb1c4]]
- [Bad checksum [should be 0xb1c4]]
- [Severity level: Error]
- [Group: Checksum]
- [Header checksum status: Bad]
- [Calculated Checksum: 0xb1c4]
- Source Address: 10.28.154.115
- Destination Address: 10.3.9.2
- > User Datagram Protocol, Src Port: 68, Dst Port: 67
- Source Port: 68
- Destination Port: 67
- Length: 308
- > Checksum: 0xb8d9 incorrect, should be 0xd072 (maybe caused by "UDP checksum offload"?)
- [Checksum Status: Bad]
- [Stream index: 13]
- > [Timestamps]
- UDP payload (300 bytes)
- > Dynamic Host Configuration Protocol (Release)

0000 10 4f 58 6c 0c 00 80 45 dd b0 f1 03 08 00 45 00 ·OXl···E·····E·

0010 01 48 d0 4c 00 00 80 11 00 00 0a 1c 9a 73 0a 03 ·H·L·····s··

0020 09 02 00 44 00 43 01 34 b8 d9 01 01 06 00 b0 ea ···D·C·4·····

0030 78 19 00 00 00 00 0a 1c 9a 73 00 00 00 00 00 00 x·····s·····

0040 00 00 00 00 00 00 80 45 dd b0 f1 03 00 00 00 00 ······E·····

序号为 2168 的分组


层级	字段名称	值	描述
数据链路层	协议类型	Ethernet II	数据链路层协议
	源 MAC 地址	IntelCor_b0:f1:03 (80:45:dd:b0:f1:03)	发送此数据包的设备的 MAC 地址
	目标 MAC 地址	ArubaaHe_6c:0c:00 (10:4f:58:6c:0c:00)	目标设备（可能是本地网络中的 DHCP 服务器或网关）的 MAC 地址
网络层	协议类型	IPv4	网络层协议

	源 IP 地址	10.28.154.115	发送此数据包的设备的 IP 地址
	目标 IP 地址	10.3.9.2	目标设备（DHCP 服务器或网关）的 IP 地址
	TTL (Time to Live)	128	数据包在网络中的最大跳数
传输层	协议类型	UDP	传输层协议
	源端口	68	客户端使用的源端口
	目标端口	67	服务器的目标端口
应用层	协议类型	DHCP	应用层协议
	op	1 (BOOTREQUEST)	操作代码，用于标识消息类型（请求或响应）
	htype	1 (Ethernet)	硬件类型，用于指定客户端网络接口的物理类型
	hlen	6	硬件地址长度，用于指定客户端物理地址（如 MAC 地址）的长度
	hops	0	跳数，用于记录中继代理转发请求的次数
	xid	（随机生成的值）	事务 ID，客户端生成的随机数，用于将请求和响应匹配起来
	secs	0	从客户端开始地址获取过程到发送此消息经过的秒数
	flags	0	标志位，用于指示是否允许广播响应
	ciaddr	10.28.154.115	客户端 IP 地址
	yiaddr	0.0.0.0	'Your'（您的）IP 地址，用于服务器分配给客户端的 IP 地址
	siaddr	0.0.0.0	服务器 IP 地址，用于指定分配 IP 地址的服务器
	giaddr	0.0.0.0	网关 IP 地址，用于指定 DHCP/BOOTP 中继代理的地址
	chaddr	IntelCor_b0:f1:03 (80:45:dd:b0:f1:03)	客户端硬件地址，通常为 MAC 地址
	sname	空字符串	服务器主机名，可选字段，用于指定 DHCP 服务器的主机名
	file	空字符串	引导文件名，可选字段，用于指定引导文件的名称

	options	53: (DHCPRELEASE)	7	选项，用于提供其他配置参数和特定的 DHCP 操作
--	---------	----------------------	---	---------------------------

序号为 2168 的 DHCP Release 数据包中各个层级

在命令行窗口中使用 >ipconfig -renew

 命令提示符

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

Windows IP 配置

不能在 以太网 上执行任何操作，它已断开媒体连接。
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。

未知适配器 本地连接:

    连接特定的 DNS 后缀 . . . . . :
    本地链接 IPv6 地址. . . . . : fe80::c007:8480:8412:836e%15
    IPv4 地址 . . . . . : 2.0.0.1
    子网掩码 . . . . . : 255.255.255.0
    默认网关. . . . . :

以太网适配器 以太网:

    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* 1:

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

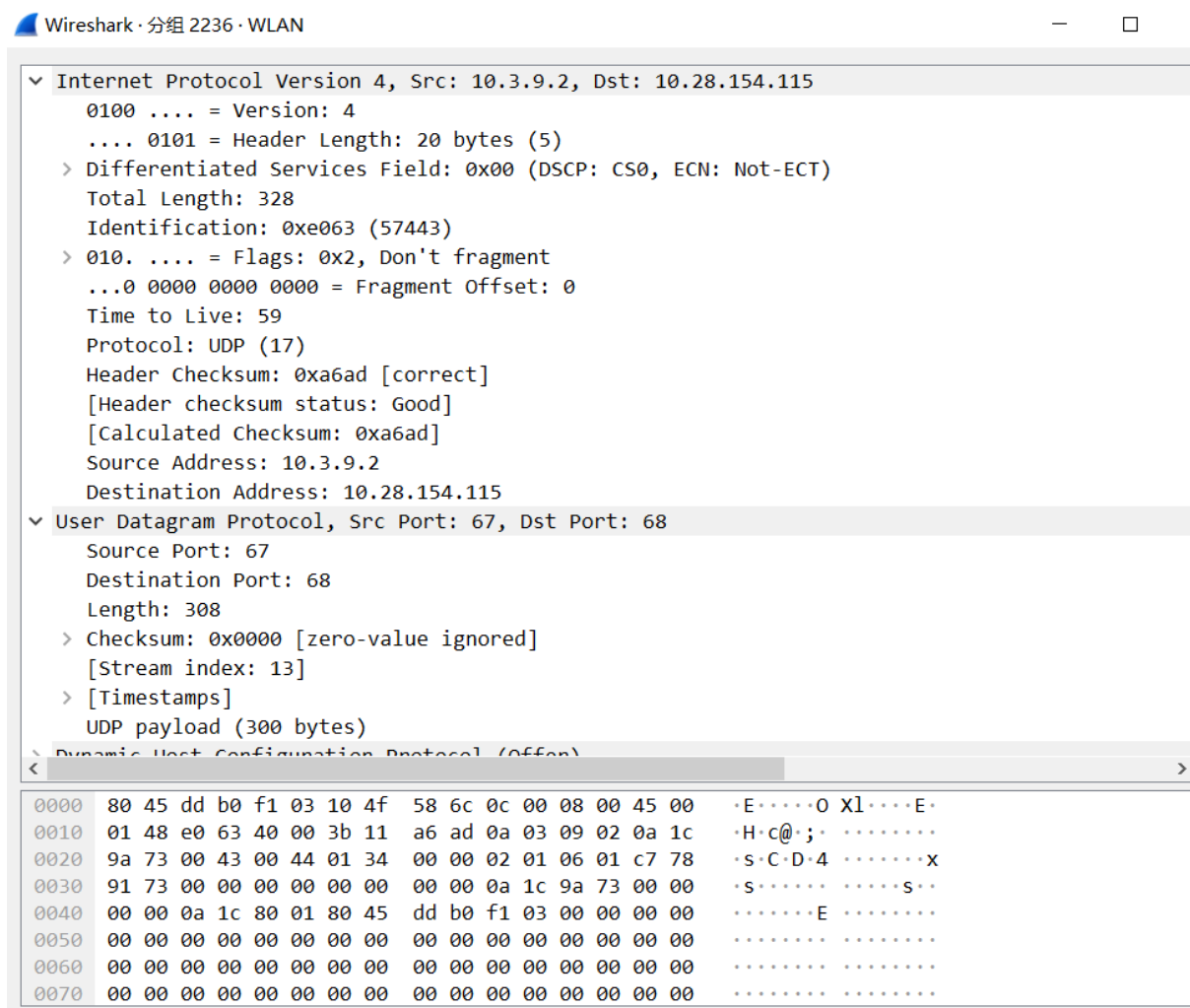
无线局域网适配器 本地连接* 2:

    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:
```

No.	Time	Source	Destination	Protocol	Length	Info
2168	43.607878	10.28.154.115	10.3.9.2	DHCP	342	DHCP Release - Transaction ID 0xb0ea7819
2235	52.073880	0.0.0.0	255.255.255.255	DHCP	344	DHCP Discover - Transaction ID 0xc7789173
2236	52.077624	10.3.9.2	10.28.154.115	DHCP	342	DHCP Offer - Transaction ID 0xc7789173
2237	52.077983	0.0.0.0	255.255.255.255	DHCP	370	DHCP Request - Transaction ID 0xc7789173
2238	52.080770	10.3.9.2	10.28.154.115	DHCP	342	DHCP ACK - Transaction ID 0xc7789173

捕获到的 DHCP Offer 和 DHCP ACK



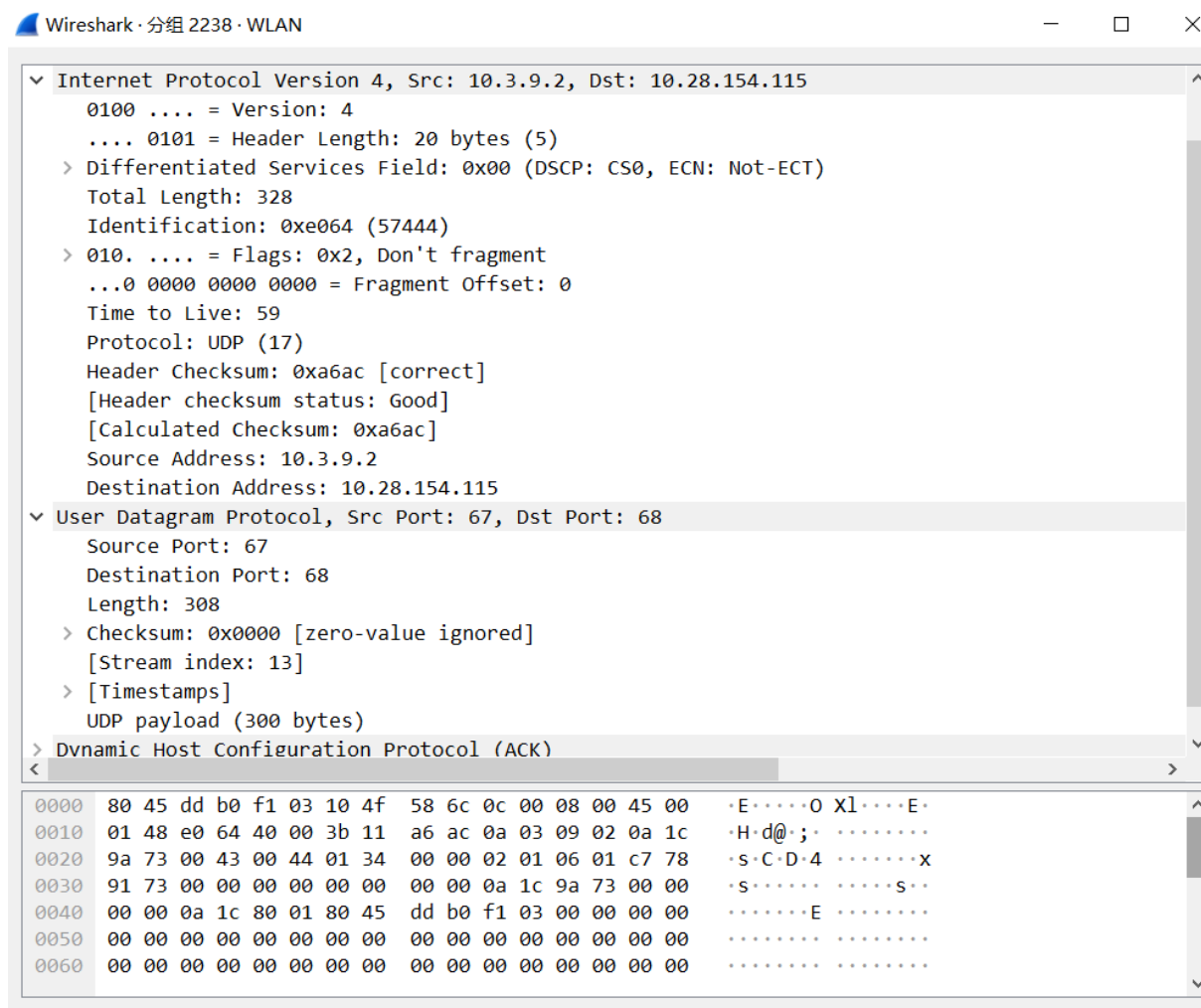
序号为 2236 的分组

以下是对 DHCP Offer 数据包的详细分析，具体地，此数据包中的 DHCP 选项及相关参数包括：

字段(字节数)	内容 (16 进制)	解释
OP (1)	02	消息类型：引导回复
HOPS (1)	00	经过的 DHCP 中继的数目：0
YIADDR (4)	0a 1c 9a 73	客户端 IP 地址（服务器分配的地址）： 10.28.154.115
GIADDR (4)	00 00 00 00	客户端发出请求分组后经过的第一个 中继的地址：0.0.0.0
OPTION (3)	35 01 02	DHCP 消息类型：Offer
OPTION (6)	36 04 0a 03 09 02	DHCP 服务器标识符：10.3.9.2
OPTION (6)	51 04 1c 20 00 00	IP 地址租约时间：7200 秒

OPTION (6)	01 04 ff ff ff 00	子网掩码: 255.255.255.0
OPTION (6)	03 04 0a 03 09 02	路由器: 10.3.9.2
OPTION (6)	06 08 0a 03 09 02 0a 03 09 02	域名服务器: 10.3.9.2
OPTION (1)	ff	选项字段结束

序号为 2236 的 DHCP Offer 数据包



序号为 2238 的分组

2238 数据包是一种 DHCP (Dynamic Host Configuration Protocol) ACK 数据包, 用于回应客户端请求并确认分配的 IP 地址、子网掩码、默认网关、DNS 服务器等网络配置信息。该数据包的源 IP 地址为 10.3.9.2, 目的 IP 地址为 10.28.154.115。

具体地, 该数据包中的 DHCP 选项及相关参数包括:

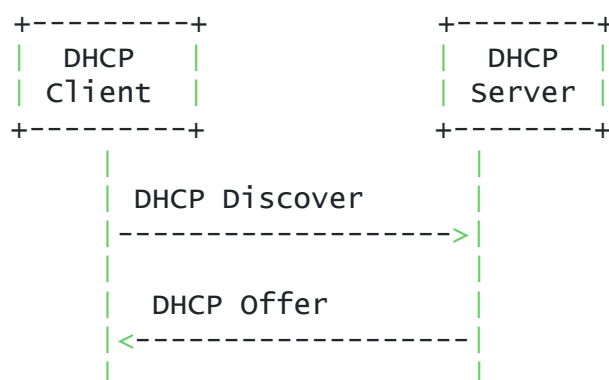
字段(字节数)	内容 (16 进制)	解释
---------	------------	----

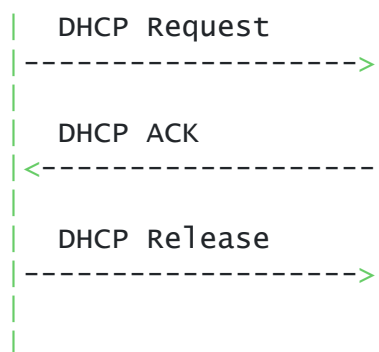
OP (1)	02	消息类型：引导回复
HOPS (1)	00	经过的 DHCP 中继的数目：0
XID (4)	7c 80 64 44	事务 ID
SECS (2)	00 00	从客户端发送 DHCP 请求经过的秒数（一般为 0）
FLAGS (2)	00 00	标记位
CIADDR (4)	0a 1c 9a 73	客户端 IP 地址
YIADDR (4)	0a 1c 9a 73	你的（客户端）IP 地址（服务器分配的地址）
SIADDR (4)	00 00 00 00	DHCP 服务器 IP 地址
GIADDR (4)	00 00 00 00	客户端发出请求分组后经过的第一个中继的地址：0.0.0.0
CHADDR (16)	48 65 6c 6c 6f 57 6f 72	客户端 MAC 地址
	6c 64 21 00 00 00 00 00	
OPTION (3)	35 01 05	DHCP 消息类型：ACK
OPTION (6)	36 04 0a 03 09 02	DHCP 服务器标识符：10.3.9.2
OPTION (6)	01 04 ff ff ff 00	子网掩码：255.255.255.0
OPTION (6)	03 04 0a 03 09 02	路由器：10.3.9.2
OPTION (6)	06 08 0a 03 09 02	域名服务器：10.3.9.2
OPTION (2)	1c 02 00 50	MTU 选项：1500
OPTION (1)	ff	选项字段结束

序号为 2238 的 DHCPACK 数据包

2.3.2 DHCP 地址分配的过程

根据的 DHCP 数据包，绘制的 DHCP 地址分配过程的消息序列图，如下所示：





DHCP 地址分配过程的消息序列图

在上述消息序列图中，DHCP Client 通过 DHCP Discover 消息向广播地址请求分配 IP 地址。DHCP Server 收到该消息并通过 DHCP Offer 消息提供可用的 IP 地址等网络配置信息给 DHCP Client。DHCP Client 确认接受所提供的网络配置信息后，向 DHCP Server 发送 DHCP Request 消息确认分配 IP 地址等网络配置信息，并通过 DHCP ACK 消息向 DHCP Client 发送确认。

需要注意的是，在 DHCP Request 数据包中，DHCP Request 消息是从 DHCP Client 的端口 68 发送到 DHCP Server 的端口 67，并且该消息的目标地址是广播地址。如果存在 DHCP Relay，则 DHCP Relay 会将广播消息转发到 DHCP Server。DHCP Server 在收到 DHCP Request 消息后，向 DHCP Client 的 IP 地址发送了 DHCP ACK 消息，该消息被 DHCP Relay 转发给了 DHCP Client。

在 DHCP ACK 数据包中，DHCP ACK 数据包是 DHCP 服务器向 DHCP 客户端发送的确认消息，以确认 DHCP 客户端分配的 IP 地址和其他网络配置信息。该数据包的源端口是 67，目标端口是 68，且使用了单播地址。

当 DHCP Client 不再需要分配的 IP 地址时，可以通过 DHCP Release 消息通知 DHCP Server 释放该 IP 地址。DHCP Server 收到 DHCP Release 消息后，将该 IP 地址标记为可用状态，以便重新分配给其他 DHCP Client 使用。

2.3.3 DHCP 协议分析

根据抓包数据，可以看到 DHCP Server 的 IP 地址为 10.3.9.2，而客户端的 IP 地址为 10.28.154.115，这两个 IP 地址不在同一个子网中，因此可以推断出 DHCP Server 不在客户端所在的子网内，需要通过 DHCP Relay 转发 DHCP 请求和响应。

此外，抓包数据中还可以看到 Relay agent IP address 字段为 10.28.128.1，这个 IP 地址可以被解释为 DHCP Relay 的 IP 地址。因此可以确认这里的 DHCP 服务是由路由器充当 DHCP Relay 转发的。

2.4 捕获 ARP 协议数据

2.4.1 使用 ipconfig 命令释放和重新申请 IP 地址

在命令行窗口中，使用 >ipconfig -release 和 >ipconfig -renew 命令

No.	Time	Source	Destination	Protocol	Length	Info
101	28.348220	IntelCor_b0:f1:03	Broadcast	ARP	42	Who has 169.254.46.40? (ARP Probe)
105	29.347919	IntelCor_b0:f1:03	Broadcast	ARP	42	Who has 169.254.46.40? (ARP Probe)
120	30.362403	IntelCor_b0:f1:03	Broadcast	ARP	42	Who has 169.254.46.40? (ARP Probe)
126	31.347742	IntelCor_b0:f1:03	Broadcast	ARP	42	ARP Announcement for 169.254.46.40
227	41.853921	IntelCor_b0:f1:03	Broadcast	ARP	42	Who has 10.28.154.115? (ARP Probe)
241	41.995655	IntelCor_b0:f1:03	Broadcast	ARP	42	Who has 10.28.128.1? Tell 10.28.154.115
242	41.997988	ArubaaHe_6c:0c:00	IntelCor_b0:f1:03	ARP	56	10.28.128.1 is at 10:4f:58:6c:0c:00
273	42.154931	IntelCor_b0:f1:03	Broadcast	ARP	42	Who has 10.28.128.1? Tell 10.28.154.115
274	42.160554	ArubaaHe_6c:0c:00	IntelCor_b0:f1:03	ARP	56	10.28.128.1 is at 10:4f:58:6c:0c:00
333	42.851483	IntelCor_b0:f1:03	Broadcast	ARP	42	Who has 10.28.154.115? (ARP Probe)
371	43.852969	IntelCor_b0:f1:03	Broadcast	ARP	42	Who has 10.28.154.115? (ARP Probe)
453	44.854496	IntelCor_b0:f1:03	Broadcast	ARP	42	ARP Announcement for 10.28.154.115
1506	47.449192	IntelCor_4e:1b:d2	IntelCor_b0:f1:03	ARP	56	Who has 10.28.154.115? Tell 10.28.157.17
1507	47.449202	IntelCor_b0:f1:03	IntelCor_4e:1b:d2	ARP	42	10.28.154.115 is at 80:45:dd:b0:f1:03

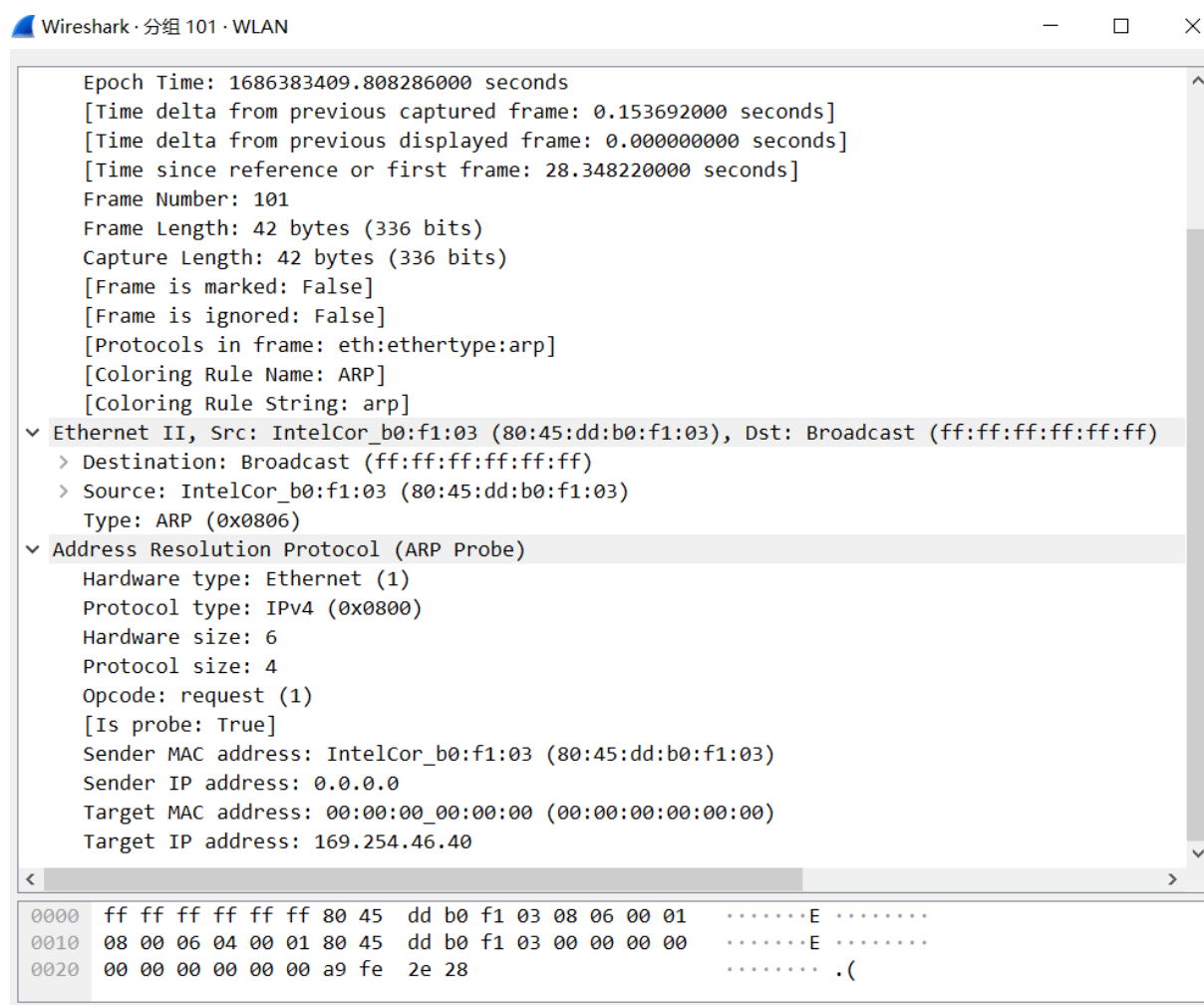
捕获到的 ARP 数据包

从捕获到的数据包中，可以看到有多个 ARP（Address Resolution Protocol）消息，这些消息包含了 ARP 的功能和操作原理

1. ARP Probe 消息（101、105、120）
2. ARP Announcement 消息（126、453）
3. ARP 请求和响应消息（241、242、273、274）
4. ARP 请求和响应消息（1506、1507）

这些数据包包含了 ARP 协议在局域网环境中通信时的基本交互流程，可以用来解释 ARP 的功能和操作原理，以及 ARP 的包格式和各字段的功能。

下面是对每个数据包的简要说明：



序号为 101 的 ARP Probe 数据包

下面是数据包 101 的详细信息，表格：

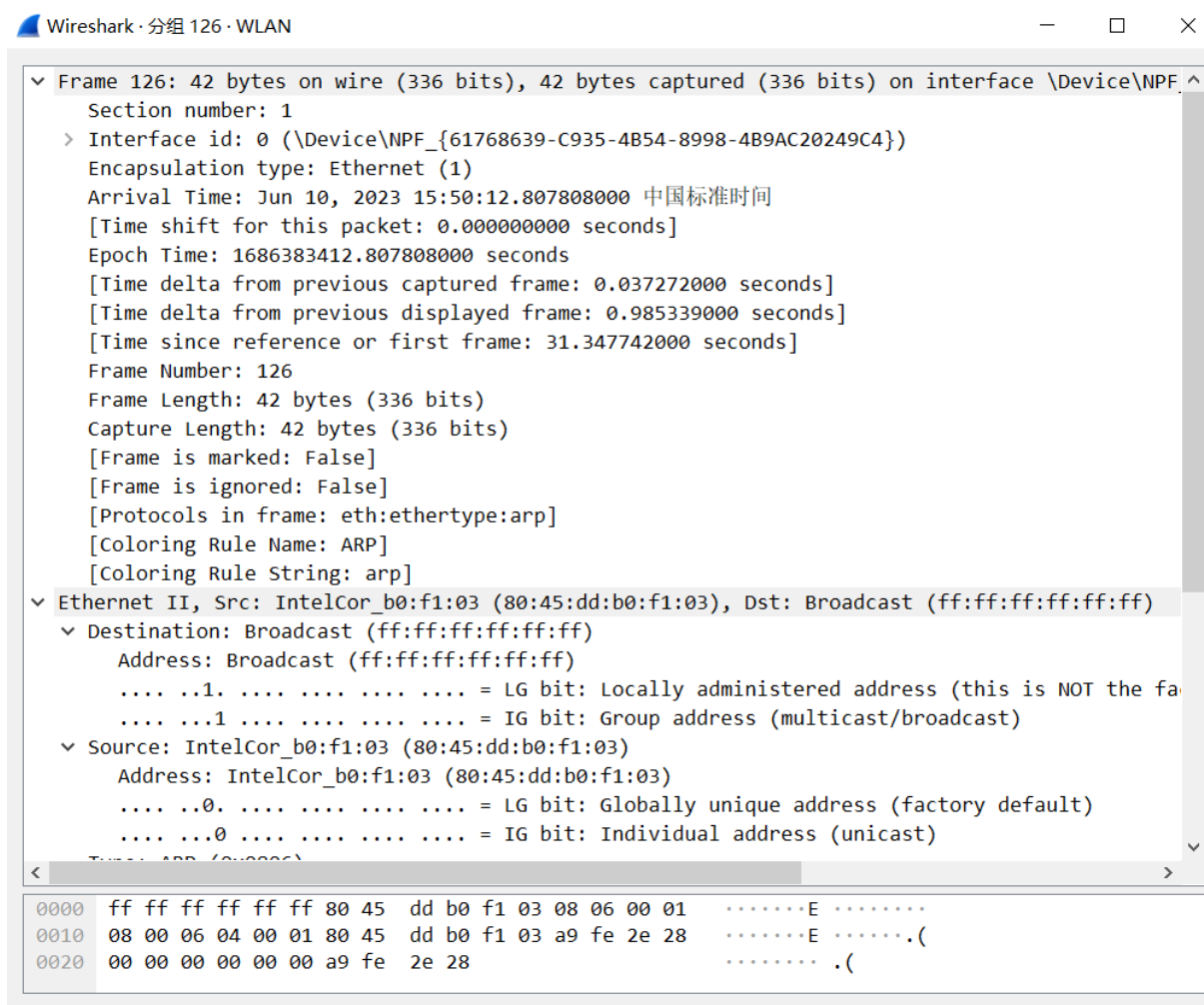
字段(字节数)	内容 (16 进制)	解释
HTYPE (2)	00 01	硬件类型：以太网
PTYPE (2)	08 00	协议类型：IPv4
HLEN (1)	06	硬件地址长度：6
PLEN (1)	04	协议地址长度：4
OPER (2)	00 01	ARP 消息类型：request
SHA (6)	80 45 dd b0 f1 03	发送方 MAC 地址：80:45:dd:b0:f1:03
SPA (4)	00 00 00 00	发送方 IP 地址：0.0.0.0
THA (6)	ff ff ff ff ff ff	目标 MAC 地址：11:11:11:11:11:11
TPA (4)	a9 fe 2e 28	目标 IP 地址：169.254.46.40

序号为 101 的数据包详细信息

ARP Announcement 消息（序号 453 数据包同理），它是由源 MAC 地址为

IntelCor_b0:f1:03 (80:45:dd:b0:f1:03) 的主机广播到本地网络的。它的目的是通知其他主机，它的 IP 地址为 169.254.46.40，并将其与自己的 MAC 地址绑定。这个消息中的 Sender MAC 地址和 Sender IP 地址字段指定了消息发送者的 MAC 和 IP 地址，而 Target MAC 地址和 Target IP 地址字段则设置为 11:11:11:11:11:11 和 169.254.46.40，分别用于指定消息中所提到的 IP 地址的 MAC 地址和广播地址。

需要注意的是，该消息中的 Opcode 字段设置为请求 (request)，表明这是一个 ARP 请求消息，它在本地网络上向所有主机广播，以便获取目标 IP 地址的 MAC 地址。同时，该消息的 Is gratuitous 和 Is announcement 字段都被设置为 True，表明这是一个无偿和广播的 ARP Announcement 消息。



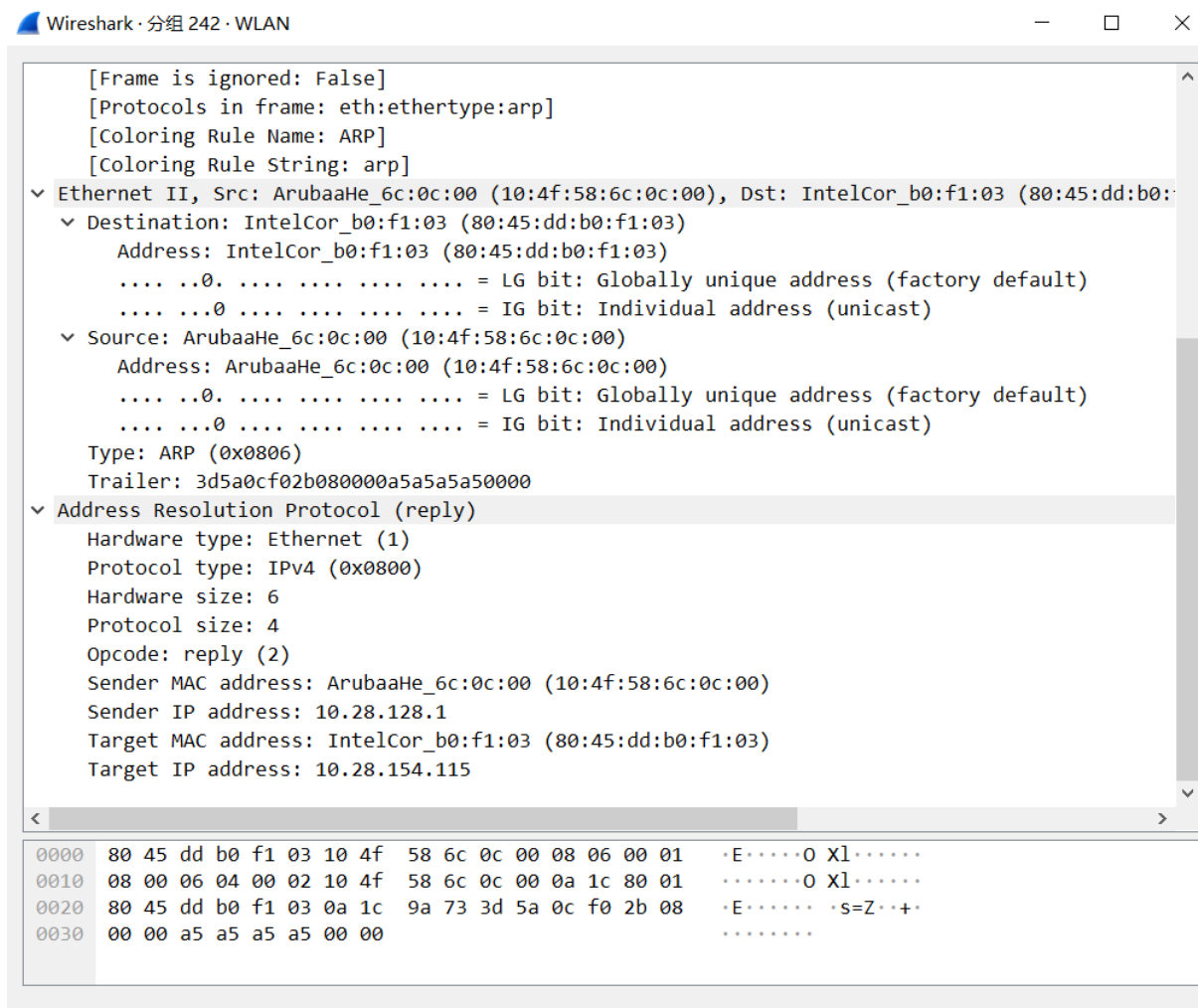
序号为 126 的 ARP Reply 数据包

字段 (字节数)	内容 (16 进制)	解释
HTYPE (2)	0001	硬件类型字段，值为 0001 表示硬件类型为 Ethernet。

PTYPE (2)	0800	协议类型字段, 值为 0800 表示协议类型为 IPv4。
HLEN (1)	06	硬件地址长度字段, 值为 06 表示 MAC 地址长度为 6 字节。
PLEN (1)	04	协议地址长度字段, 值为 04 表示 IPv4 地址长度为 4 字节。
OPER (2)	0002	操作码字段, 值为 0002 表示该数据包为 ARP 响应消息。
SHA (6)	10:4f:58:6c:0c:00	发送方 MAC 地址字段, 值为 10:4f:58:6c:0c:00, 表示发送方的 MAC 地址为 10:4f:58:6c:0c:00。
SPA (4)	10.28.128.1	发送方 IP 地址字段, 值为 10.28.128.1, 表示发送方的 IP 地址为 10.28.128.1。
THA (6)	80:45:dd:b0:f1:03	目标 MAC 地址字段, 值为 80:45:dd:b0:f1:03, 表示目标的 MAC 地址为 80:45:dd:b0:f1:03。
TPA (4)	10.28.154.115	目标 IP 地址字段, 值为 10.28.154.115, 表示目标的 IP 地址为 10.28.154.115。

序号为 126 的数据包详细信息

该数据包是一个 ARP Announcement 消息 (序号 453 同理), 它是由主机在获得自己的 IP 地址后, 向本地网络广播的消息, 用于通知其他主机自己已经获得该 IP 地址, 并将其与自己的 MAC 地址绑定。ARP Announcement 消息的作用是更新网络中其他设备的 ARP 缓存, 以便其他设备可以快速地找到发送方的 MAC 地址。该数据包的以太网帧头部分表示该数据包是一个广播消息, 目的地 MAC 地址为 ff:ff:ff:ff:ff:ff, 表示该数据包应该被发送到本地网络中的所有设备。ARP Announcement 消息的操作码为 1, 表示该数据包是一个 ARP 请求消息。Sender MAC 地址为 IntelCor_b0:f1:03 (80:45:dd:b0:f1:03), Sender IP 地址为 169.254.46.40, 表示发送方的 MAC 地址为 80:45:dd:b0:f1:03, IP 地址为 169.254.46.40。Target MAC 地址为 00:00:00_00:00:00 (00:00:00:00:00:00), Target IP 地址为 169.254.46.40, 表示该数据包是一个 ARP Announcement 消息, 目的地 MAC 地址和 IP 地址均为广播地址, 表示该消息不针对特定的设备, 而是应该被发送到本地网络中的所有设备。



序号为 242 的 ARP Reply 数据包

字段(字节数)	内容(16 进制)	解释
HTYPE (2)	0001	硬件类型为 Ethernet
PTYPE (2)	0800	协议类型为 IPv4
HLEN (1)	06	MAC 地址长度为 6 字节
PLEN (1)	04	IPv4 地址长度为 4 字节
OPER (2)	0002	ARP 操作码为响应 (2)
SHA (6)	104f586c0c00	发送方 MAC 地址为 10:4f:58:6c:0c:00
SPA (4)	0a208001	发送方 IP 地址为 10.28.128.1
THA (6)	8045ddb0f103	目标 MAC 地址为 80:45:dd:b0:f1:03
TPA (4)	0a1c9a73	目标 IP 地址为 10.28.154.115

序号为 242 的数据包详细信息

这个数据包是 ARP（地址解析协议）响应消息（序号 274 数据包同理），用于回复源主机发送的 ARP 请求消息，告知源主机目标 IP 地址对应的 MAC 地址。该数据包

的源 MAC 地址为 10:4f:58:6c:0c:00，目的 MAC 地址为 80:45:dd:b0:f1:03，表示该数据包是由源主机发送给目标主机的。ARP 请求和响应消息的目的是用于查询目标 IP 地址对应的 MAC 地址。

2.4.2 ARP 各字段的功能

ARP (Address Resolution Protocol) 协议是一种用于将 IP 地址转换为 MAC 地址的协议，它使用请求和响应消息来确定网络上的主机的 MAC 地址。下面是 ARP 消息中各个字段的功能

字段	功能
Hardware type	指定网络上使用的硬件类型（如以太网）
Protocol type	指定网络层协议类型（如 IPv4）
Hardware size	指定硬件地址的长度（通常为 6 个字节，即 MAC 地址的长度）
Protocol size	指定协议地址的长度（通常为 4 个字节，即 IPv4 地址的长度）
Opcode	指定 ARP 消息的类型，如 ARP 请求或 ARP 响应
Sender MAC address	指定发送 ARP 消息的主机的 MAC 地址
Sender IP address	指定发送 ARP 消息的主机的 IP 地址
Target MAC address	指定 ARP 消息中所带的目标 MAC 地址，只有在 ARP 响应消息中才有数据
Target IP address	指定 ARP 消息中所带的目标 IP 地址

ARP 各字段的详细信息

ARP 协议通过使用这些字段来确定网络上的主机的 MAC 地址，以便进行通信。当一个主机需要发送数据到另一个主机时，它可以使用 ARP 协议来查询目标主机的 MAC 地址，然后将数据包发送到目标主机的 MAC 地址。

2.5 TCP 协议分析

2.5.1 用浏览器打开和关闭网页，捕获 TCP 协议数据

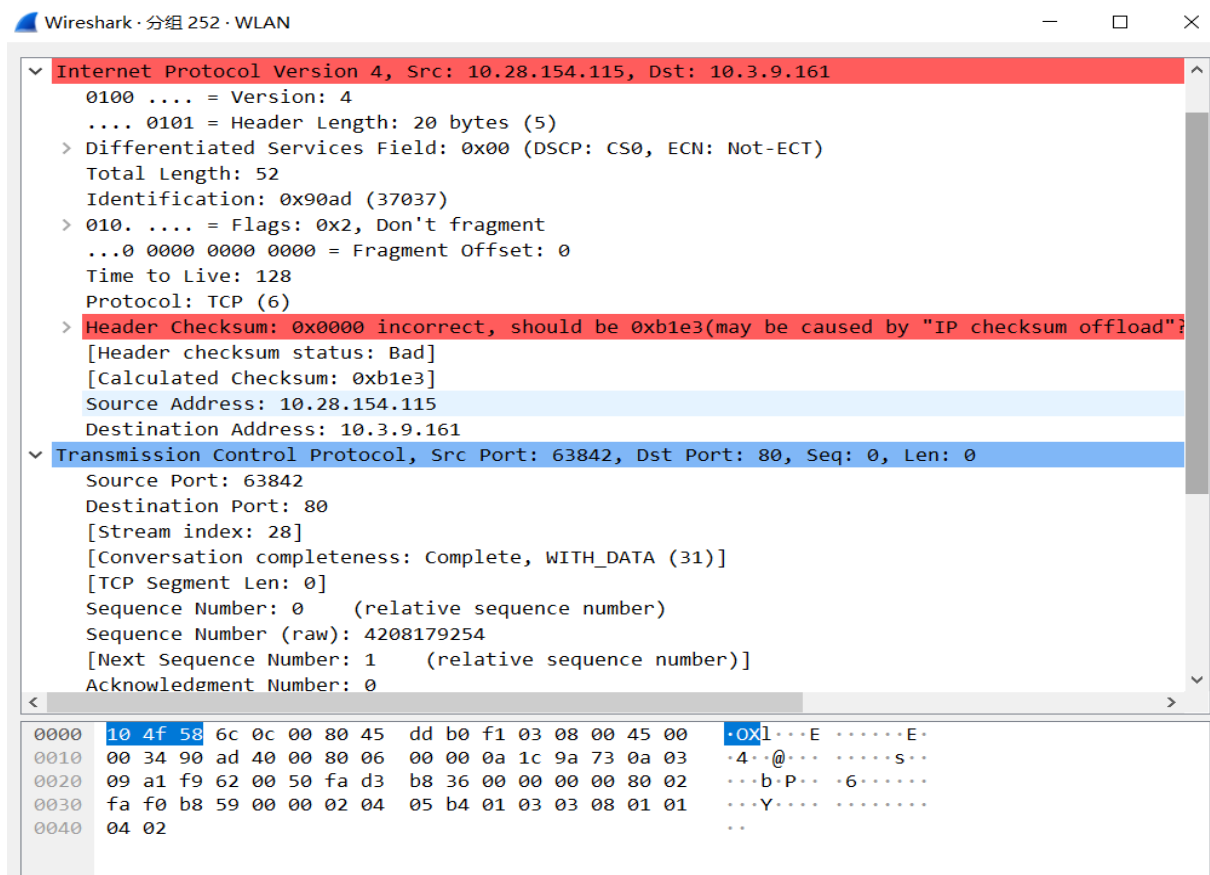
用火狐浏览器打开 www.bupt.edu.cn，在页面完全显示后，关闭浏览器

No.	Time	Source	Destination	Protocol	Length	Info
252	10.195484	10.28.154.115	10.3.9.161	TCP	66	63842 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
253	10.198810	10.3.9.161	10.28.154.115	TCP	66	80 → 63842 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1386 SACK_PERM WS...
254	10.198875	10.28.154.115	10.3.9.161	TCP	54	63842 → 80 [ACK] Seq=1 Ack=1 Win=131584 Len=0
255	10.199007	10.28.154.115	10.3.9.161	HTTP	501	GET / HTTP/1.1
256	10.201797	10.3.9.161	10.28.154.115	TCP	60	80 → 63842 [ACK] Seq=1 Ack=448 Win=30720 Len=0
257	10.202965	10.3.9.161	10.28.154.115	TCP	1440	80 → 63842 [ACK] Seq=1 Ack=448 Win=30720 Len=1386 [TCP segment of a rea...
258	10.204189	10.3.9.161	10.28.154.115	TCP	208	80 → 63842 [PSH, ACK] Seq=1387 Ack=448 Win=30720 Len=154 [TCP segment o...
259	10.204189	10.3.9.161	10.28.154.115	TCP	1325	80 → 63842 [PSH, ACK] Seq=1541 Ack=448 Win=30720 Len=1271 [TCP segment ...
260	10.204189	10.3.9.161	10.28.154.115	TCP	1177	80 → 63842 [PSH, ACK] Seq=2812 Ack=448 Win=30720 Len=1123 [TCP segment ...
261	10.204189	10.3.9.161	10.28.154.115	TCP	998	80 → 63842 [PSH, ACK] Seq=3935 Ack=448 Win=30720 Len=944 [TCP segment o...
262	10.204189	10.3.9.161	10.28.154.115	TCP	920	80 → 63842 [PSH, ACK] Seq=4879 Ack=448 Win=30720 Len=866 [TCP segment o...
263	10.204223	10.28.154.115	10.3.9.161	TCP	54	63842 → 80 [ACK] Seq=448 Ack=5745 Win=131584 Len=0
264	10.205209	10.3.9.161	10.28.154.115	TCP	815	80 → 63842 [PSH, ACK] Seq=5745 Ack=448 Win=30720 Len=761 [TCP segment o...
265	10.205209	10.3.9.161	10.28.154.115	TCP	1119	80 → 63842 [PSH, ACK] Seq=6506 Ack=448 Win=30720 Len=1065 [TCP segment ...
266	10.205209	10.3.9.161	10.28.154.115	TCP	900	80 → 63842 [PSH, ACK] Seq=7571 Ack=448 Win=30720 Len=846 [TCP segment o...
267	10.205233	10.28.154.115	10.3.9.161	TCP	54	63842 → 80 [ACK] Seq=448 Ack=8417 Win=131584 Len=0
268	10.206387	10.3.9.161	10.28.154.115	TCP	871	80 → 63842 [PSH, ACK] Seq=8417 Ack=448 Win=30720 Len=817 [TCP segment o...
269	10.206387	10.3.9.161	10.28.154.115	TCP	1440	80 → 63842 [ACK] Seq=9234 Ack=448 Win=30720 Len=1386 [TCP segment of a ...
270	10.206387	10.3.9.161	10.28.154.115	TCP	1440	80 → 63842 [ACK] Seq=10620 Ack=448 Win=30720 Len=1386 [TCP segment of a ...

捕获的 TCP 数据包

2.5.2 连接建立

序号 252-254: 客户端向服务器发送 SYN 请求建立连接, 服务器返回 SYN+ACK 应答, 客户端返回 ACK 完成连接建立。

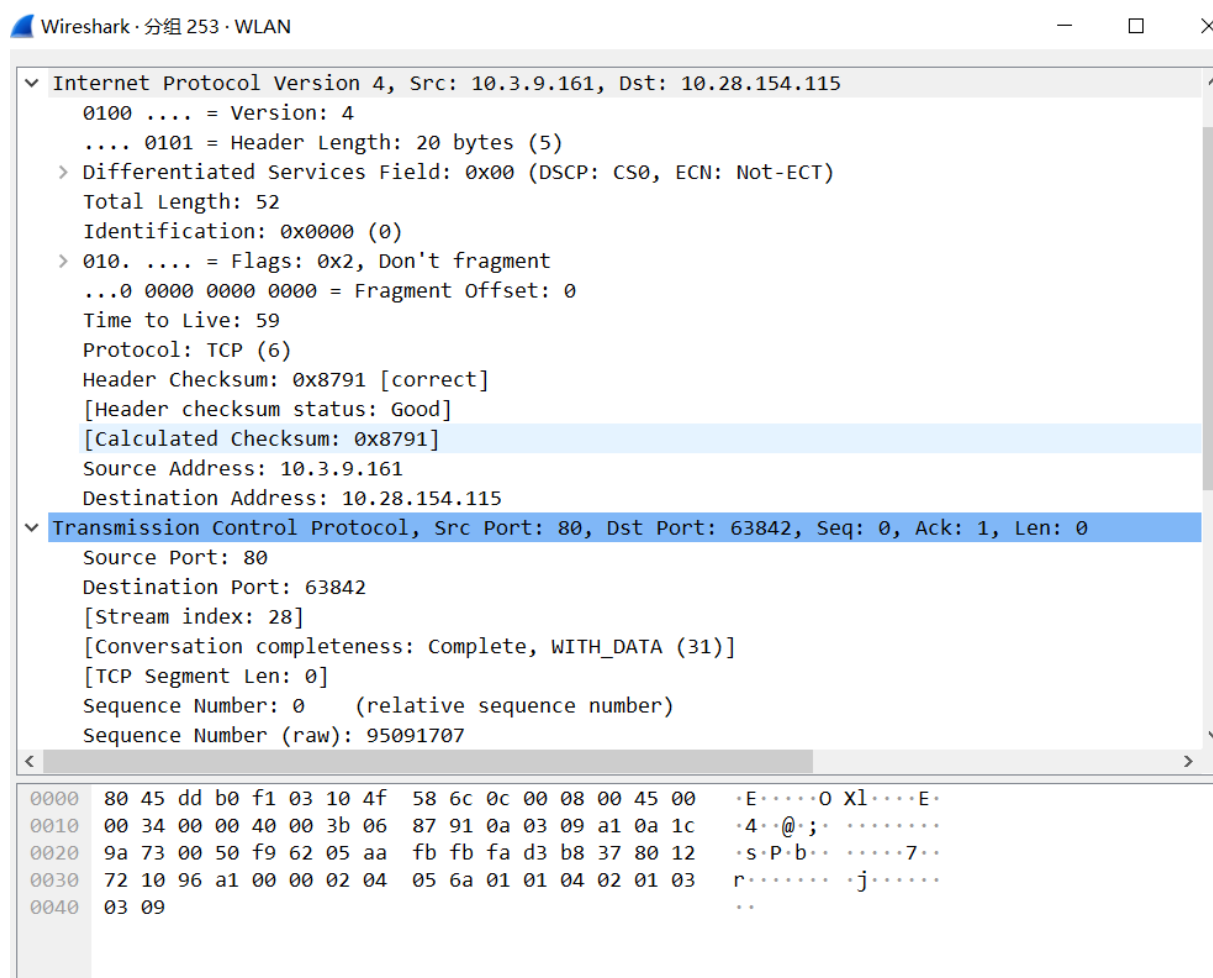


序号为 252 的 SYN 请求信息

详细信息如下标

字段 (字节数)	内格(16 进制)	解释
源端口 (2)	10 0B	源端口号为 2766 = 0x010B
目的端口(2)	00 64	目的端口号为 80 = 0x0064
序列号(4)	00 00 00 00	序列号为 0
接收序列号(4)	00 00 00 00	接收序列号为 0
首部长度(4)	80 00 00	首部长度为 32 字节 = 0x20
标志位 (1)	02	仅 SYN 位为 1
窗口大小(2)	D8 40	窗口大小为 64240 = 0xD840
校验和 (2)	B8 59	校验和值为 0xB859
紧急指针(2)	00 00	紧急指针为 0
选项(可变)	0C 00	含 MSS、WSALE、SACK 等选项

序号为 252 的数据包详细信息



序号为 253 的 SYN 和 ACK 应答信息

字段 (字节数)	内容 (16 进制)	解释
源端口 (2)	00 50	源端口号为 80 (十进制)
目的端口 (2)	0b 0a	目的端口号为 63842 (十进制)
序列号 (4)	00 00 00 00	序列号为 0
确认号 (4)	00 00 00 01	确认号为 1
首部长度 (4)	00 80 00 00	首部长度为 8 个 32 位字, 即 32 字节
标志位 (1)	0x12	SYN 和 ACK 位均为 1
窗口大小 (2)	0x7210	窗口大小为 29200 (十进制)
校验和 (2)	0x96a1	校验和值为 0x96a1
紧急指针 (2)	0x0000	紧急指针为 0
选项 (可变)	0x020405b40401 b080	包含 MSS、SACK permitted 和 Window scale 选项, 总长度为 12 字节, 即 0x0c

序号为 253 的分组详细信息

根据以上表格中的各个字段描述, 该数据包是一个 TCP SYN-ACK 报文段, 用于建立一个新的 TCP 连接。其中, SYN 和 ACK 位均为 1, 窗口大小为 29200, 校验和为 0x96a1, 选项中包含了 MSS、SACK permitted 和 Window scale 选项。

TCP 协议中的标志位和序号的作用如下:

标志位:

SYN (Synchronize): 用于建立连接, 表示请求建立一条连接。

ACK (Acknowledgment): 确认标志位, 表示已经收到对方发送的数据。

FIN (Finish): 用于释放连接, 表示数据传输完成, 请求释放连接。

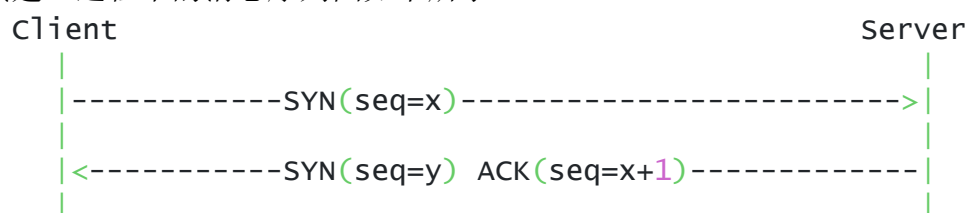
RST (Reset): 表示重置连接, 一般用于异常情况, 如连接超时或出现错误。

序号:

序列号: 表示发送方发送的数据的第一个字节的序号。

确认号: 表示接收方期望收到的下一个字节的序号。

连接建立过程中的消息序列图如下所示




```
|-----ACK(seq=y+1, ack=x+1)----->|
|
```

连接建立消息序列图

由图，下面是连接建立过程的描述：

客户端向服务器发送一个 SYN 报文，其中序列号为 x，SYN 标志位为 1，表示客户端请求建立连接。

服务器收到 SYN 报文后，向客户端发送一个 SYN+ACK 报文，其中序列号为 y，确认号为 x+1，SYN 和 ACK 标志位均为 1，表示服务器确认收到了客户端的请求，并请求客户端确认连接。

客户端收到服务器发送的 SYN+ACK 报文后，向服务器发送一个 ACK 报文，其中序列号为 y+1，确认号为 x+1，ACK 标志位为 1，表示客户端确认连接已建立。

在连接建立的过程中，SYN 报文用于请求建立连接，SYN+ACK 报文用于确认收到连接请求并请求确认，ACK 报文用于确认连接建立。序列号用于标识数据流中的每个报文段，确认号用于确认对方已经收到了自己发送的报文段。在 SYN 和 SYN+ACK 报文中，序列号用于初始化数据流的起始序号，确认号则用于确认收到的报文序列号。在 ACK 报文中，序列号用于确认收到的报文序列号，确认号则用于确认对方已经收到了自己发送的报文序列号。

2.5.3 数据传输过程

序号 258 - 277 展示了服务器向客户端传输 HTTP 响应数据，发送多个带有 PSH 和 ACK 标志的 TCP 段，客户端返回 ACK 应答。

No.	Time	Source	Destination	Protocol	Length	Info
258	10.204189	10.3.9.161	10.28.154.115	TCP	208	80 → 63842 [PSH, ACK] Seq=1387 Ack=448 Win=30720 Len=0
259	10.204189	10.3.9.161	10.28.154.115	TCP	1325	80 → 63842 [PSH, ACK] Seq=1541 Ack=448 Win=30720 Len=0
260	10.204189	10.3.9.161	10.28.154.115	TCP	1177	80 → 63842 [PSH, ACK] Seq=2812 Ack=448 Win=30720 Len=0
261	10.204189	10.3.9.161	10.28.154.115	TCP	998	80 → 63842 [PSH, ACK] Seq=3935 Ack=448 Win=30720 Len=0
262	10.204189	10.3.9.161	10.28.154.115	TCP	920	80 → 63842 [PSH, ACK] Seq=4879 Ack=448 Win=30720 Len=0
263	10.204223	10.28.154.115	10.3.9.161	TCP	54	63842 → 80 [ACK] Seq=448 Ack=5745 Win=131584 Len=0
264	10.205209	10.3.9.161	10.28.154.115	TCP	815	80 → 63842 [PSH, ACK] Seq=5745 Ack=448 Win=30720 Len=0
265	10.205209	10.3.9.161	10.28.154.115	TCP	1119	80 → 63842 [PSH, ACK] Seq=6506 Ack=448 Win=30720 Len=0
266	10.205209	10.3.9.161	10.28.154.115	TCP	900	80 → 63842 [PSH, ACK] Seq=7571 Ack=448 Win=30720 Len=0
267	10.205233	10.28.154.115	10.3.9.161	TCP	54	63842 → 80 [ACK] Seq=448 Ack=8417 Win=131584 Len=0
268	10.206387	10.3.9.161	10.28.154.115	TCP	871	80 → 63842 [PSH, ACK] Seq=8417 Ack=448 Win=30720 Len=0
269	10.206387	10.3.9.161	10.28.154.115	TCP	1440	80 → 63842 [ACK] Seq=9234 Ack=448 Win=30720 Len=1386
270	10.206387	10.3.9.161	10.28.154.115	TCP	1440	80 → 63842 [ACK] Seq=10620 Ack=448 Win=30720 Len=1386
271	10.206418	10.28.154.115	10.3.9.161	TCP	54	63842 → 80 [ACK] Seq=448 Ack=12006 Win=131584 Len=0
272	10.207447	10.3.9.161	10.28.154.115	TCP	1440	80 → 63842 [ACK] Seq=12006 Ack=448 Win=30720 Len=1386
273	10.207447	10.3.9.161	10.28.154.115	TCP	1440	80 → 63842 [ACK] Seq=13392 Ack=448 Win=30720 Len=1386
274	10.207447	10.3.9.161	10.28.154.115	TCP	1440	80 → 63842 [ACK] Seq=14778 Ack=448 Win=30720 Len=1386
275	10.207492	10.28.154.115	10.3.9.161	TCP	54	63842 → 80 [ACK] Seq=448 Ack=16164 Win=131584 Len=0
276	10.208490	10.3.9.161	10.28.154.115	HTTP	1432	HTTP/1.1 200 OK (text/html)
277	10.208512	10.28.154.115	10.3.9.161	TCP	54	63842 → 80 [ACK] Seq=448 Ack=17542 Win=130048 Len=0

Wireshark · 分组 258 · WLAN

Internet Protocol Version 4, Src: 10.3.9.161, Dst: 10.28.154.115

- 0100 = Version: 4
- 0101 = Header Length: 20 bytes (5)
- > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- Total Length: 194
- Identification: 0x0c9a (3226)
- > 010. = Flags: 0x2, Don't fragment
- ...0 0000 0000 0000 = Fragment Offset: 0
- Time to Live: 59
- Protocol: TCP (6)
- Header Checksum: 0x7a69 [correct]
- [Header checksum status: Good]
- [Calculated Checksum: 0x7a69]
- Source Address: 10.3.9.161
- Destination Address: 10.28.154.115

Transmission Control Protocol, Src Port: 80, Dst Port: 63842, Seq: 1387, Ack: 448, Len: 154

- Source Port: 80
- Destination Port: 63842
- [Stream index: 28]
- [Conversation completeness: Complete, WITH_DATA (31)]
- [TCP Segment Len: 154]

< >

0000	80 45 dd b0 f1 03 10 4f	58 6c 0c 00 08 00 45 00	E.....O Xl.....E.
0010	00 c2 0c 9a 40 00 3b 06	7a 69 0a 03 09 a1 0a 1c@.; zi.....
0020	9a 73 00 50 f9 62 05 ab	01 66 fa d3 b9 f6 50 18	.s.P.b... f....P.
0030	00 3c a2 a4 00 00 d1 7e	9c b0 b9 bd 3a 0d bc 9c	.<.....~;
0040	59 f5 76 28 aa d0 5b 36	65 63 e4 fe 2d 26 dd e1	Y.v(..[6 ec...&..
0050	61 8b 51 31 77 72 92 e2	a4 43 48 ce 9e 3d ec f5	a.Q1wr... CH...=...
0060	0e ba 07 2d fc 75 7b ad	76 ab dd ec 41 ef b0 d3-u{ v...A...
0070	eb 76 ba ed 6e ab db e9	74 9a ed e6 21 de ab 36	.v..n... t...!...6
0080	79 f7 7a 90 6a 61 d6 98	c8 e1 76 0d a5 dd b5 55	y.z.ja... ..v....U
0090	84 2d 71 6d 41 68 e3 94	a2 34 71 c1 c6 fd 80 4e	..qmAh... 4q.....N
00a0	70 13 00 88 6c 39 e9 b9	71 cf a1 83 95 98 92 7c	p...l9... q.....

序号为 258 的 PSH 和 ACK 应答

字段 (字节数)	内格(16 进制)	解释
源端口 (2)	00 50	源端口号为 80 = 0x0050
目的端口(2)	F9 22	目的端口号为 63842 = 0xF922
序列号(4)	00 00 38 43	序列号为 1387
接收序列号(4)	00 00 01 C0	接收序列号为 448
首部长度的(4)	50 12 0C 53	首部长度的为 20 字节 = 0x14
标志位 (1)	00 18	PSH 和 ACK 位均为 1
窗口大小(2)	00 3C	窗口大小为 60 = 0x003C
校验和 (2)	A2 A4	校验和值为 0xA2A4
紧急指针(2)	00 00	紧急指针为 0
选项(可变)	无	无

序号为 258 的数据包详细信息

根据数据包中的信息，可以看出这是一个 TCP 报文，源端口号为 80，目的端口号为 63842。标志位中的 PSH 和 ACK 位均为 1，表示这是一个有数据的确认报文。窗口大小为 60，校验和为 0xA2A4。此外，数据包中没有包含任何 TCP 选项。

从 TCP 报文的内容中，还可以看出该报文是对先前的 SYN 报文的回应。其中，序列号为 1387，表示该报文中的数据是从这个序列号处开始的。确认号为 448，表示收到了序列号为 448 的数据包，并期望收到下一个序列号为 448 的数据包。此外，数据包中还包含了 TCP 数据段，长度为 154 字节。

对于 TCP 数据传输过程中的数据报文段和应答报文段，以下是各个字段的作用：

发送序号：指发送方发送数据的第一个字节的序号。

应答序号：指接收方期望接收的下一个字节的序号。

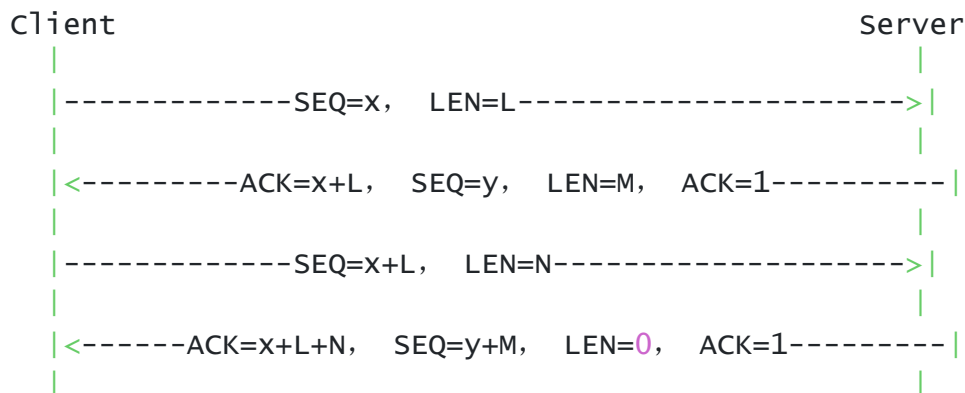
应答标志位：用于表示 TCP 数据报文段的类型，例如 SYN 表示连接请求，FIN 表示连接终止，ACK 表示确认报文等。

窗口大小：指接收方当前可以接收的最大字节数，用于流量控制。

数据长度：指 TCP 数据报文段中数据的长度，不包括 TCP 首部。

MSS：最大分段大小，即 TCP 传输时每个数据段的最大长度，由 TCP 连接的双方协商得出。

以下是 TCP 数据传输过程的消息序列图，包括了数据校验错和数据丢失导致的数据重传情形，标出了对应的序号、标志位和窗口大小。



数据传输过程的消息序列图

在这个消息序列图中，客户端向服务器发送一个长度为 L 的 TCP 数据报文，其中发送序号为 x 。服务器收到数据后，向客户端发送一个确认报文，其中确认号为 $x+L$ ，期望接收序号为 y ，长度为 M ，并设置 ACK 标志位为 1。客户端收到确认报文后，向服务器发送另一个长度为 N 的数据报文，其中发送序号为 $x+L$ ，长度为 N 。服务器收到数据后，向客户端发送一个确认报文，其中确认号为 $x+L+N$ ，期望接收序号为 $y+M$ ，长度为 0，并设置 ACK 标志位为 1。

如果数据传输过程中发生数据校验错或者数据丢失，那么接收方会发送一个重传请求，请求发送方重新发送数据。在这种情况下，发送方会重新发送对应的数据报文段，但是序列号和确认号可能会有所变化，以确保接收方可以正确地接收数据。在重传的数

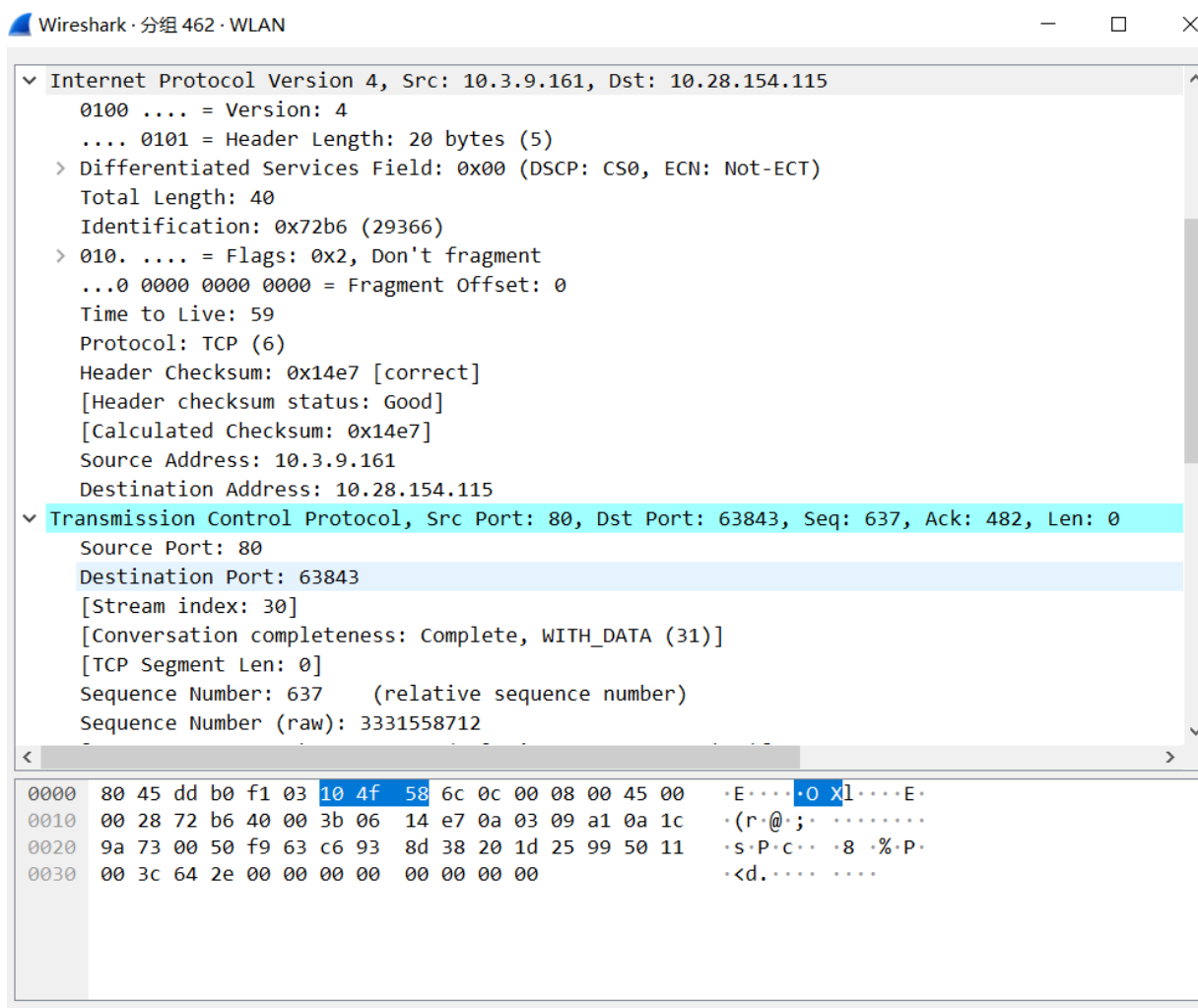
据报文段中，序列号和确认号应该与第一次发送时的不同。

另外，确认报文中设置 ACK 标志位为 1 是表示确认接收到的数据报文，而不是表示确认消息本身。在消息序列图中，ACK 标志位应该在服务器向客户端发送的确认报文中设置为 1，而不是在客户端向服务器发送的数据报文中。因为 ACK 标志位表示确认接收到的数据报文，而不是确认收到的消息。

2.5.4 数据和应答报文段

序号 426 – 465 数据包展示了客户端和服务端分别发送 FIN 请求，对方返回 FIN+ACK 应答，最后发送 ACK 完成连接释放。

426	21.962717	10.28.154.115	10.3.9.161	TCP	54 63843 → 80 [FIN, ACK] Seq=481 Ack=637 Win=130816 Len=0
427	21.962744	10.28.154.115	10.3.9.161	TCP	54 63842 → 80 [FIN, ACK] Seq=966 Ack=18096 Win=131584 Len=0
462	21.966382	10.3.9.161	10.28.154.115	TCP	60 80 → 63843 [FIN, ACK] Seq=637 Ack=482 Win=30720 Len=0
463	21.966382	10.3.9.161	10.28.154.115	TCP	60 80 → 63842 [FIN, ACK] Seq=18096 Ack=967 Win=31744 Len=0
464	21.966471	10.28.154.115	10.3.9.161	TCP	54 63843 → 80 [ACK] Seq=482 Ack=638 Win=130816 Len=0
465	21.966553	10.28.154.115	10.3.9.161	TCP	54 63842 → 80 [ACK] Seq=967 Ack=18097 Win=131584 Len=0



序号为 462 的 FIN+ACK 应答

以下是对序号为 462 的数据包进行分析

字段 (字节数)	内格(16 进制)	解释
----------	-----------	----

源端口 (2)	50 00	源端口号为 80 = 0x0050
目的端口(2)	FC 33	目的端口号为 63843 = 0xFC33
序列号(4)	0000027D	序列号为 637 = 0x27D
接收序列号(4)	208B5DD9	接收序列号为 538781081 = 0x208B5DD9
首部长度(4)	50 18 00 35	首部长度为 20 字节，标志位为 0x500x18
标志位 (1)	11	ACK 和 FIN 标志位均为 1
窗口大小(2)	003C	窗口大小为 60 = 0x003C
校验和 (2)	642E	校验和值为 0x642E
紧急指针(2)	0000	紧急指针为 0
选项(可变)	None	该数据包没有包含任何选项字段

序号为 462 分组的详细信息

这个数据包是一个 TCP 报文，它的源端口号为 80，目的端口号为 63843。根据 TCP 协议，数据包的标志位为 0x11，其中 ACK 和 FIN 标志位均为 1。这表示服务器接收到了客户端发送的数据，并且不再向客户端发送任何数据。根据数据包的内容和标志位，可以看出该数据包是一个 FIN+ACK 应答报文，用于响应之前由客户端发送的 FIN 请求报文。服务器已完成数据传输并准备关闭连接。最后的 ACK 报文用于确认对方已经关闭连接，从而完成连接释放。

在连接释放的过程中，序列号和确认号的作用与连接建立过程中相同，用于确认对方是否确认了自己发送的关闭连接请求，从而完成连接的释放。

连接释放部分的消息序列图如下所示：



连接释放的消息序列图

客户端和服务端之间的连接释放过程如下：

客户端向服务器发送一个 FIN 报文，其中序列号为 a，FIN 标志位为 1，表示客户端

请求关闭连接。

服务器收到 FIN 报文后向客户端发送一个 FIN+ACK 报文，其中序列号为 c ，确认号为 $b+1$ ，FIN 标志位为 1，ACK 标志位为 1，表示服务器请求关闭连接并确认客户端先前的 FIN 报文。

在数据传输过程中，服务器向客户端发送一个 FIN+ACK 报文，其中序列号为 c ，确认号为 $b+1$ ，FIN 和 ACK 标志位均为 1，表示服务器请求关闭连接并请求客户端确认。

客户端收到 FIN+ACK 报文后向服务器发送一个 ACK 报文，其中序列号为 $b+1$ ，确认号为 $c+1$ ，ACK 标志位为 1，表示客户端确认关闭连接。

第三章 实验结论和心得

本次实验主要通过实际捕获和分析网络数据，加深对 IP、TCP、UDP、ICMP、ARP 和 DHCP 等协议的理解。通过分析各个协议的数据包格式，理解了各个字段的含义和作用，对协议的工作原理有了更加直观的认识。

在实验过程中，理解数据包头格式和字段信息存在一定难度。通过多次检查教材和资料，分析模拟，逐渐理清了各个字段的含义。在分析 DHCP、ARP 消息序列时，需要多次观察捕获的消息，结合资料理解协议的工作过程。此外，分析 TCP 连接建立、释放和数据传输过程时，对于序号、窗口管理等概念的理解不够深刻，通过多次调试和修改，才能总结出较为合理的消息序列图。

总的来说，本次实验给我带来很多收获。通过实际操作，一是巩固和扩展了对协议工作原理的理解，二是学习了实用的网络分析工具。在后续的网络学习和工作中，Wireshark 软件将作为研究网络问题和故障排除的必要工具。