**实验二 Wireshark、数据帧和IP包分析**

# 一、实验目的

1. 学习捕获和分析网络数据包
2. 掌握以太网MAC帧、802.11数据帧和IPv4数据包的构成，了解各字段的含义
3. 掌握ICMP协议，ping和tracert指令的工作原理
4. 掌握ARP协议的请求/响应机理

# 二、实验内容

任务1：捕获和分析有线以太网数据包

准备步骤：学习Wireshark基本操作

1.1 分析MAC帧

1.2 分析IP数据报首部

1.3 观察IP分片

1.4 ICMP协议分析（以ping指令为例）

1.5 tracert工作原理分析

1.6 ARP协议分析

任务2：捕获和分析802.11数据帧

构建无线环境，捕获无线数据包、分析802.11数据帧

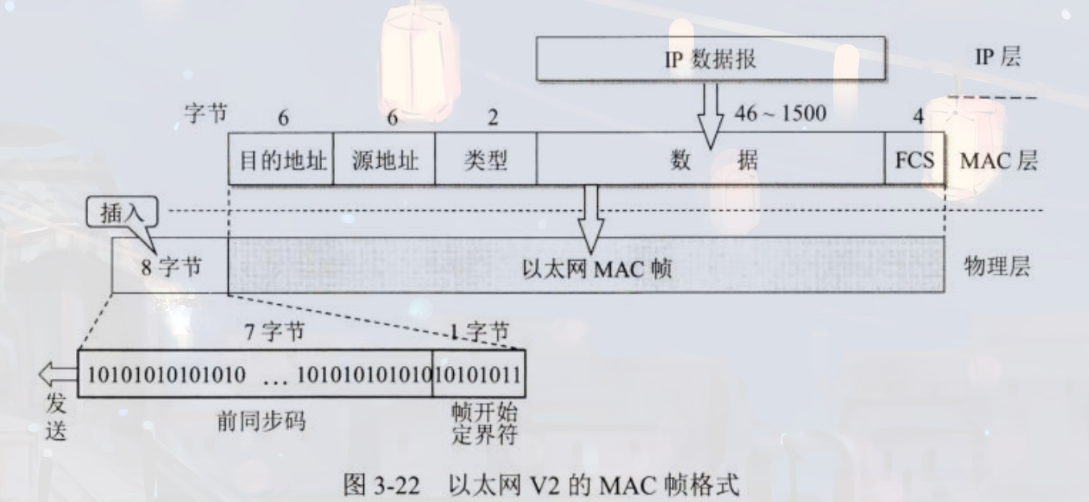
任务3：探索Wireshark更多功能和其它抓包工具(选做)

# 三、实验结果和分析

### 任务 1 捕获和分析有线以太网数据包

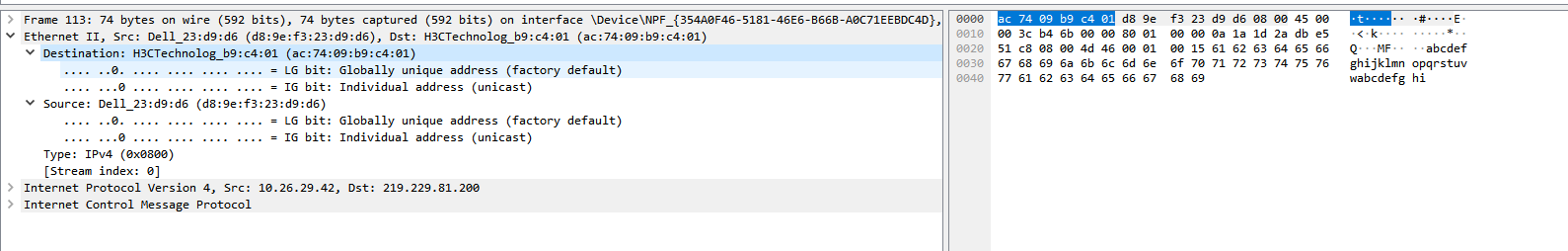
#### **观察MAC帧格式**

下图为mac帧的格式



（1）目的地址(最开始6个字节)：ac 74 09 b9 c4 01

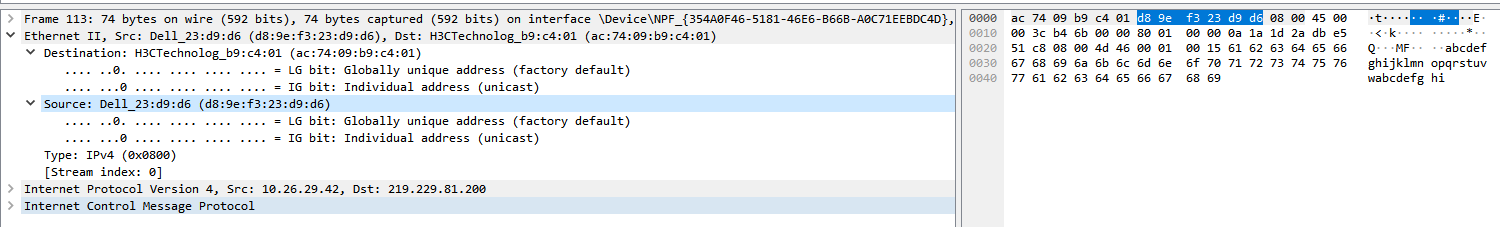
Wireshark提供的文字版分析：ac:74:09:b9:c4:01



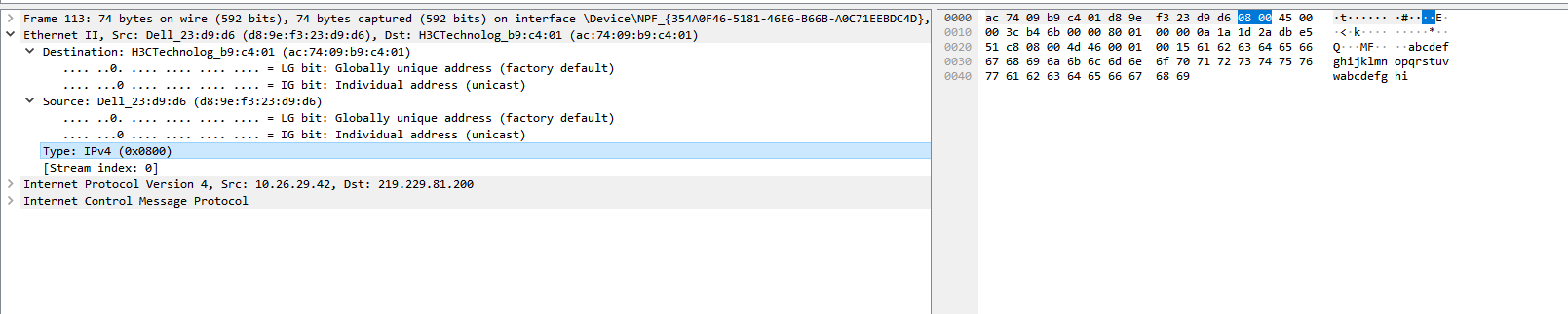
(2) 源地址（接着6个字节）：d8 9e f3 23 d9 d6

左图显示：d8:9e:f3:23:d9:d6

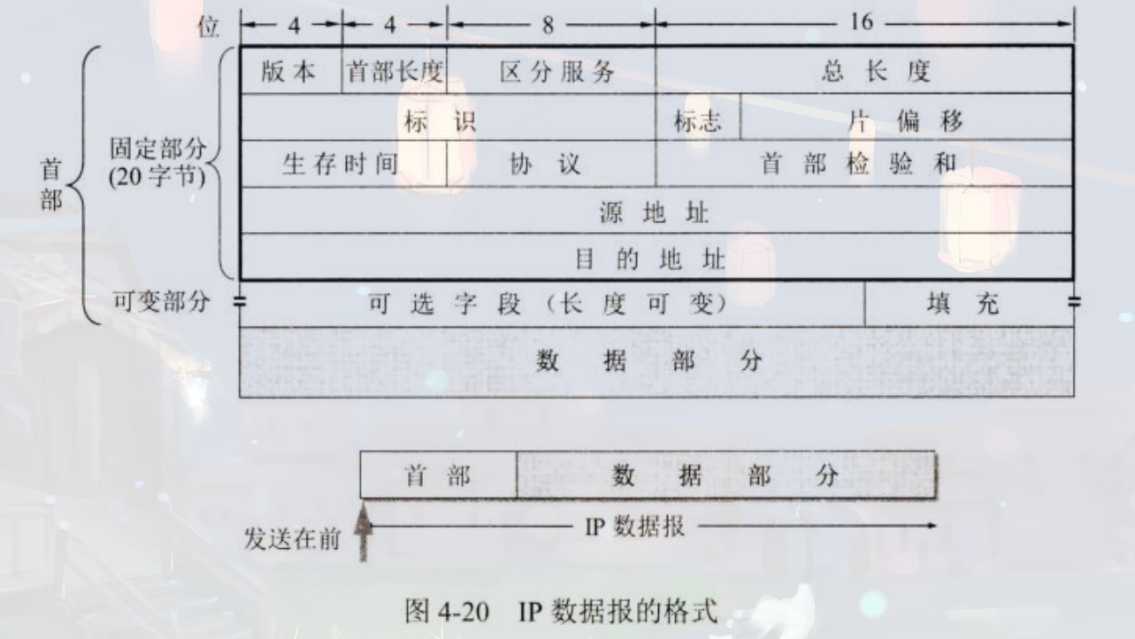
显示网卡产商：Dell\_23:d9:d6



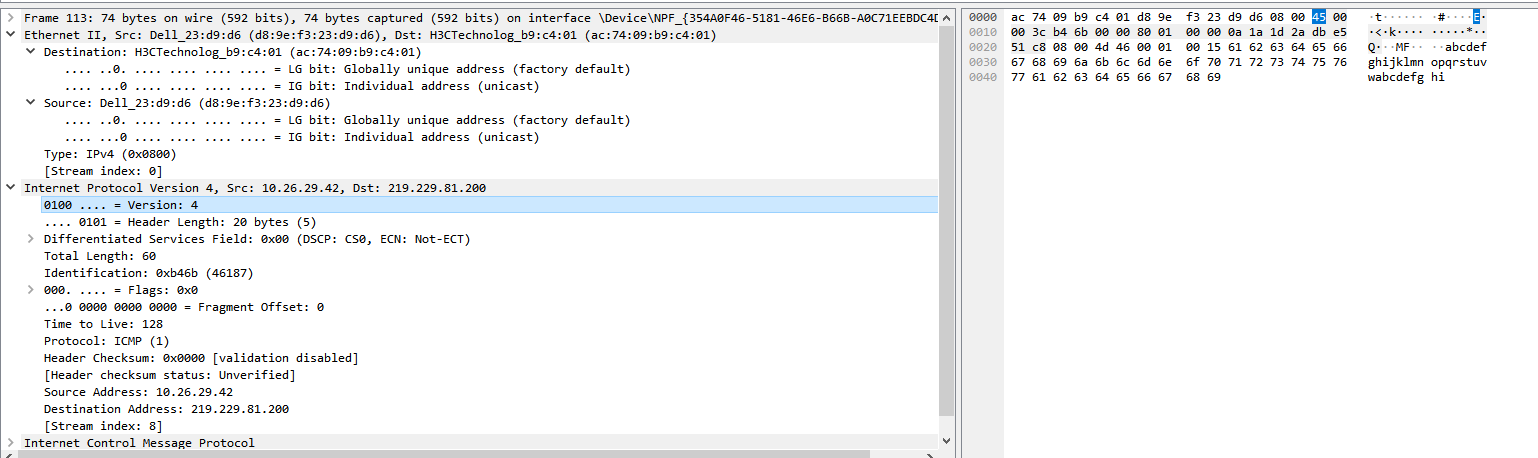
(3) 类型字段（接下来两个字节）：0x0800 ，代表上一层为IP数据报



#### **1.2.1 观察IP数据报的首部结构——IPv4**

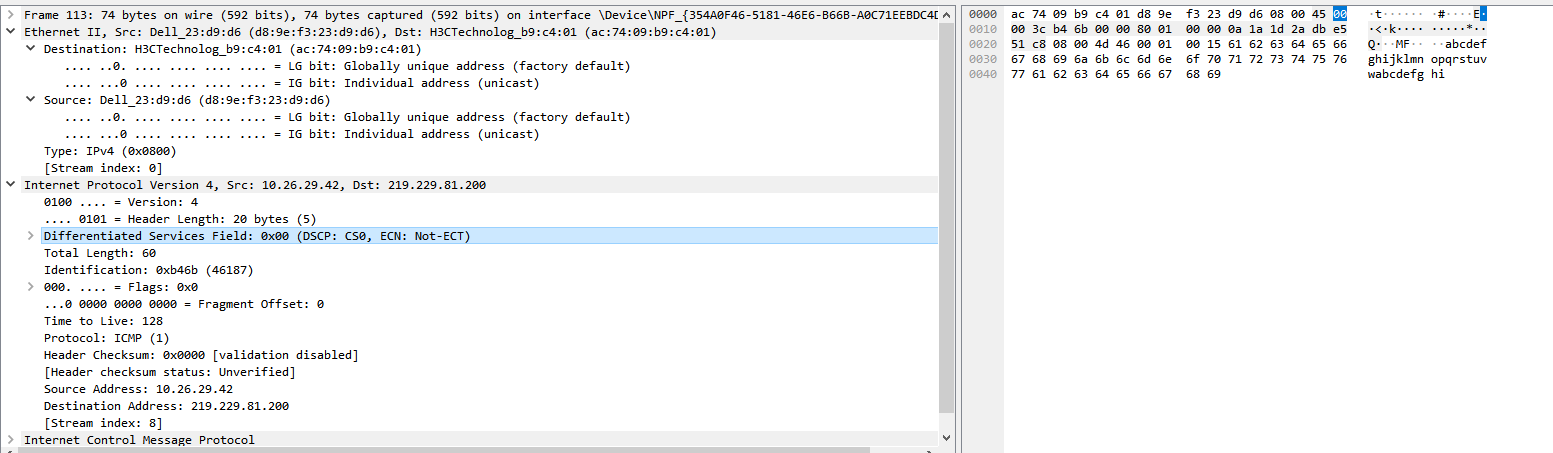


（1）版本（占4位）： 4，代表协议版本号为IPv4



（2）首部长度（占4位）：5，5\*4=20，首部长度位20字节

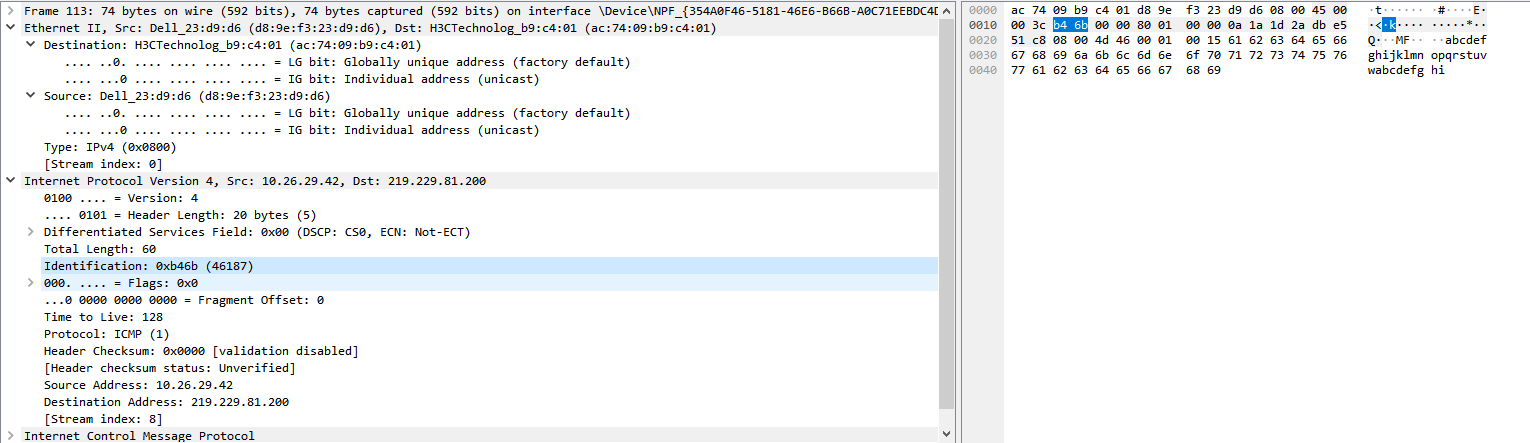
（3）区分服务（占8位）：00，这个字段一般不使用



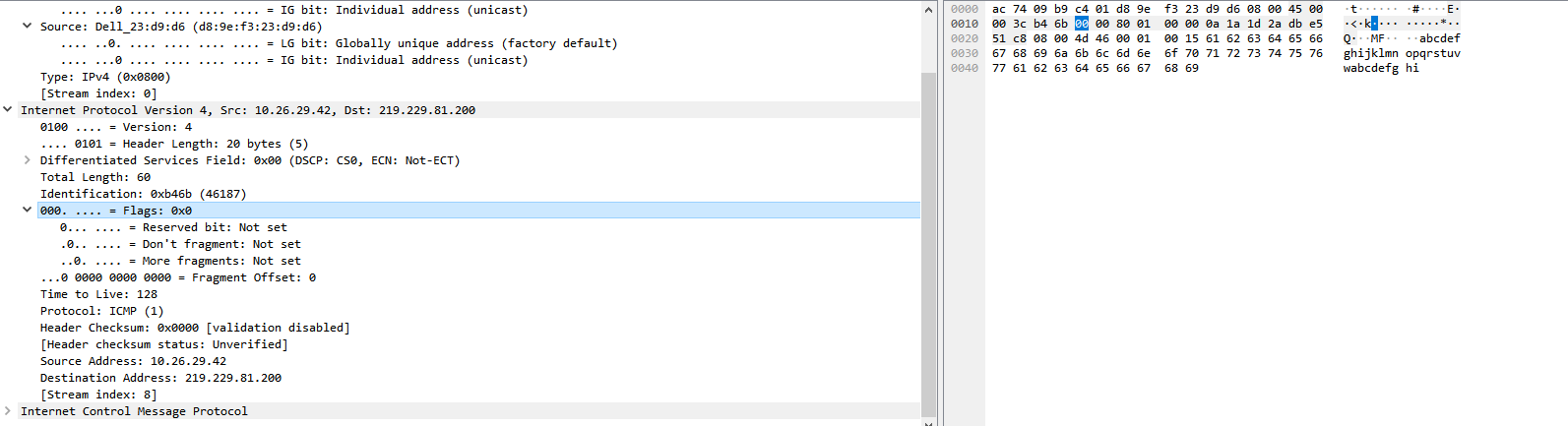
（4）总长度（占16位）：00 3c，0x3c=60(十进制)，IP数据报的总长度位60，因为首部位20，所以数据部分为40



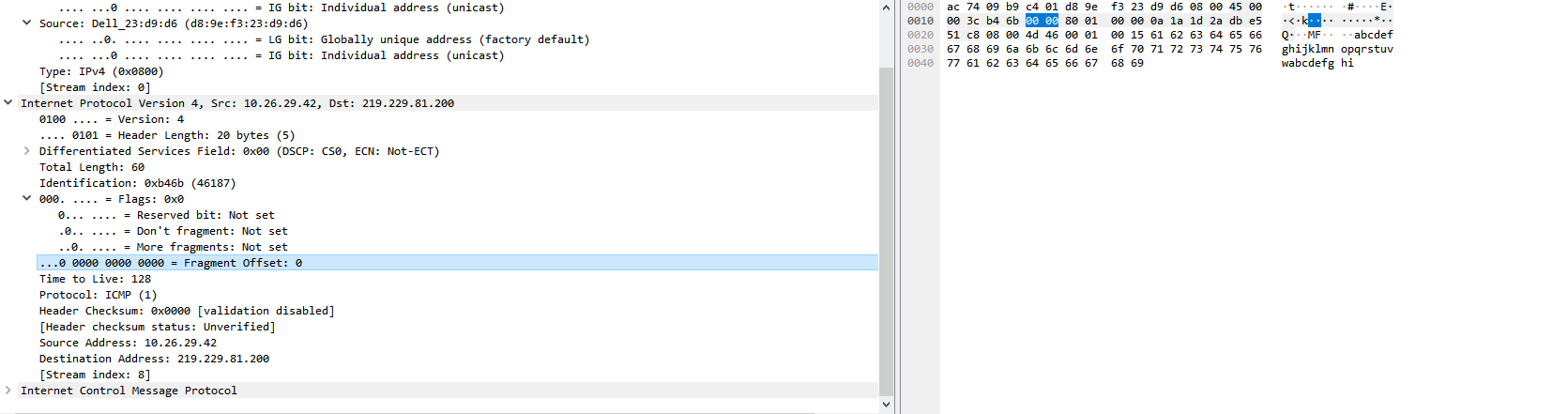
（5）标识（占16位）：b4 6b,是ip数据报内部的一个计数器，因为这里没有分片，所以这个字段没有什么意义



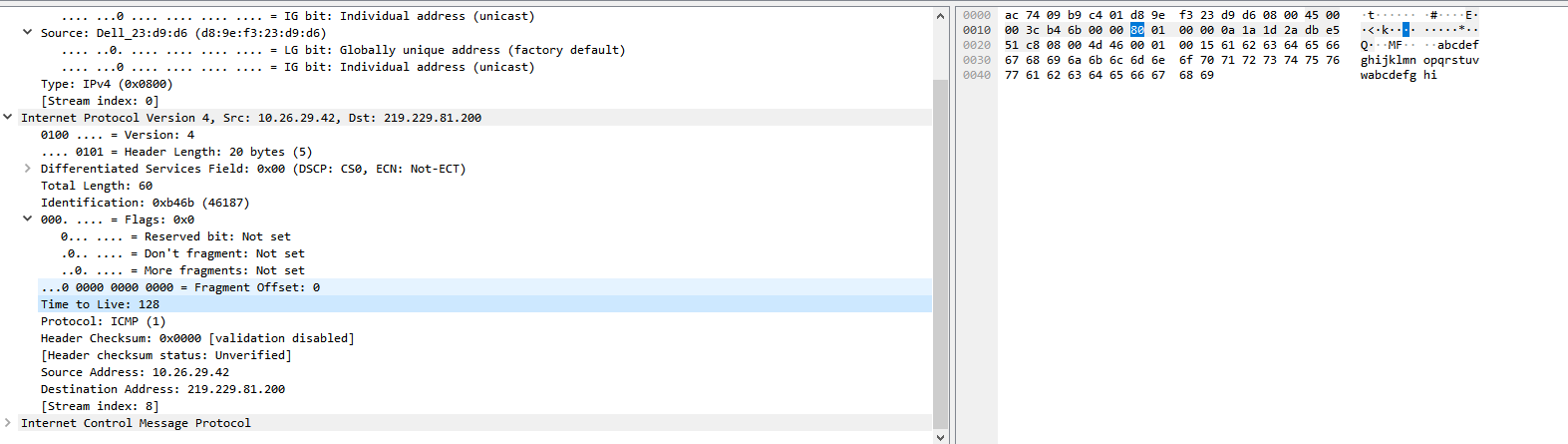
（6）标志（占3位）：00，MF=0表示后面没有分片，DF=0表示允许分片



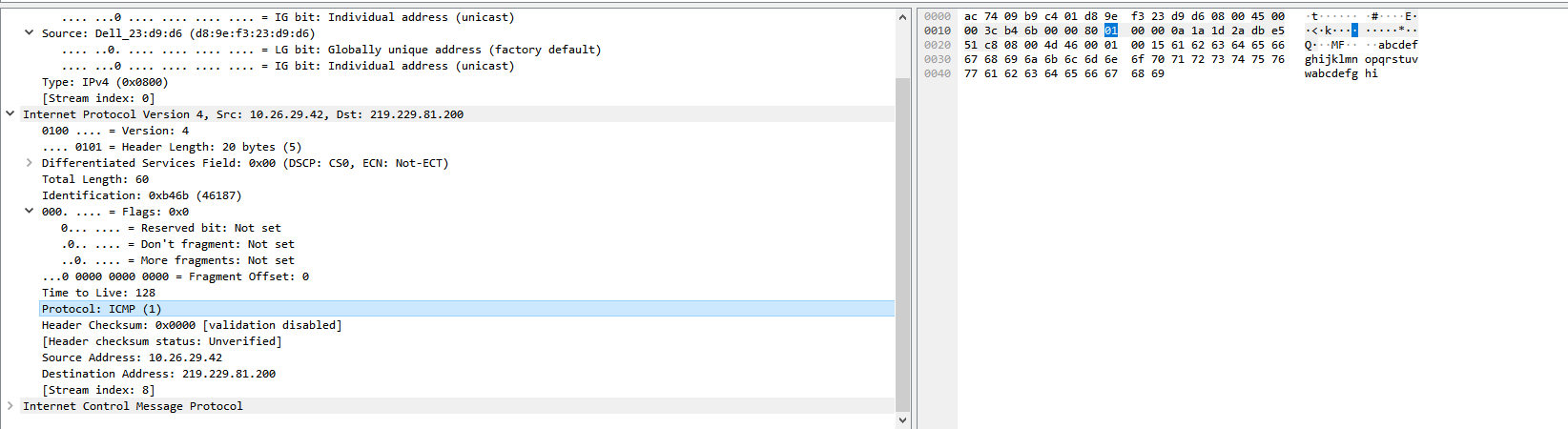
（7） 片偏移（占13位）：000，表示无偏移最低位叫做**MF**,**MF=1**表示后面还有若干个数据报,MF=0表示这已经是最后一个数据报了。中间位叫做DF,DF表示不能进行分片**,DF=0才可以进行分片操作。**



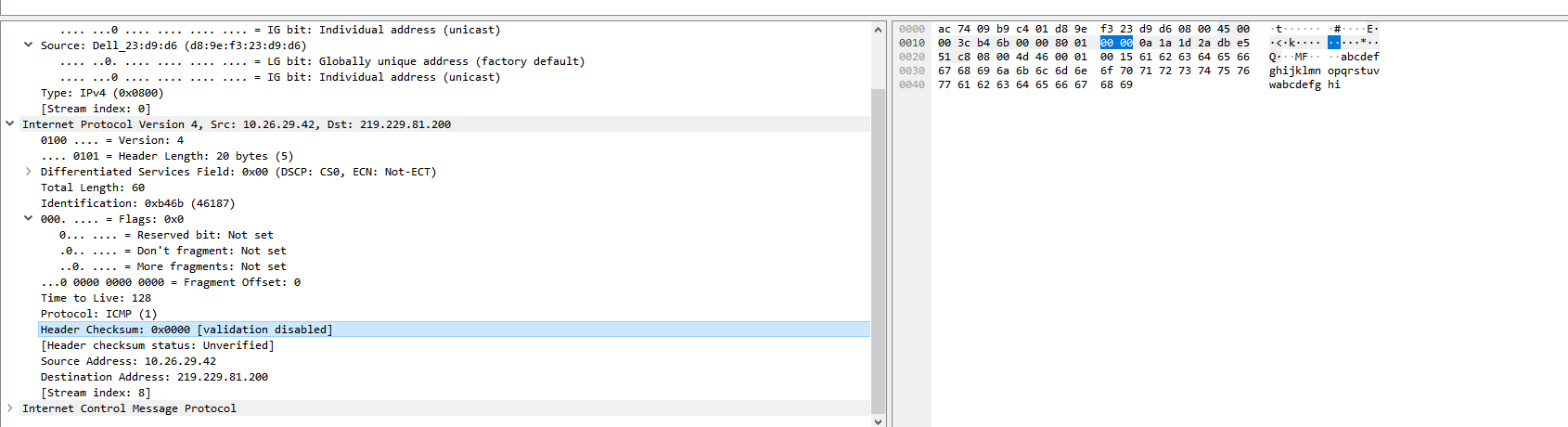
（8）生存时间TTL（占8位）：00,表示丢弃这个数据报，不再转发



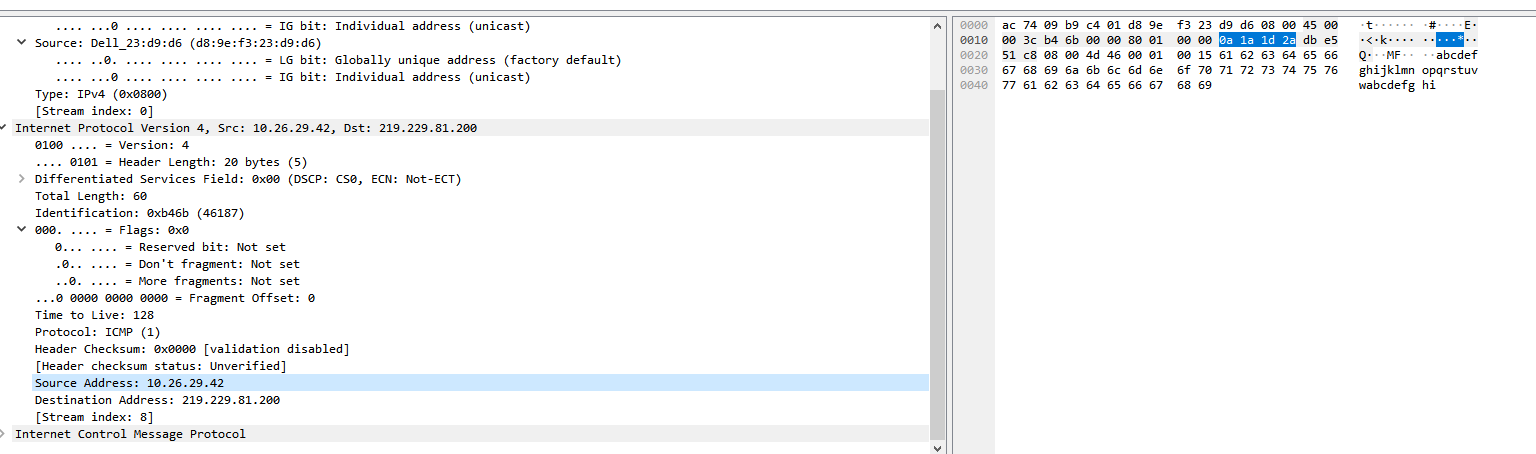
（9）协议（占8位）：01，表示ICMP协议



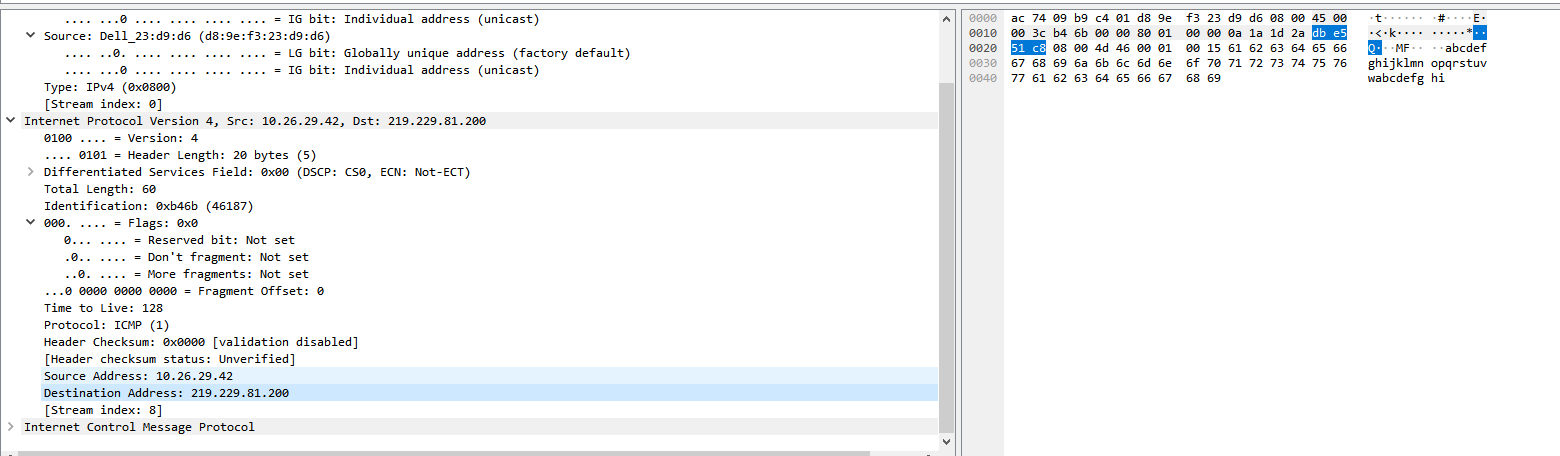
（10）首部检验和（占16位）：0000，用于检验数据报的首部是否正确



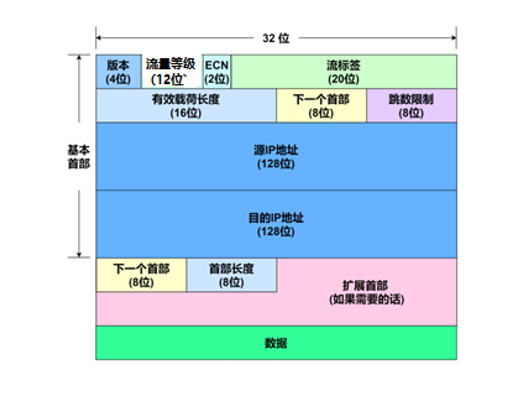
（11） 源地址（占32位）：0a 1a 1d 2a;左侧显示出10.26.29.42(对应点分十进制)



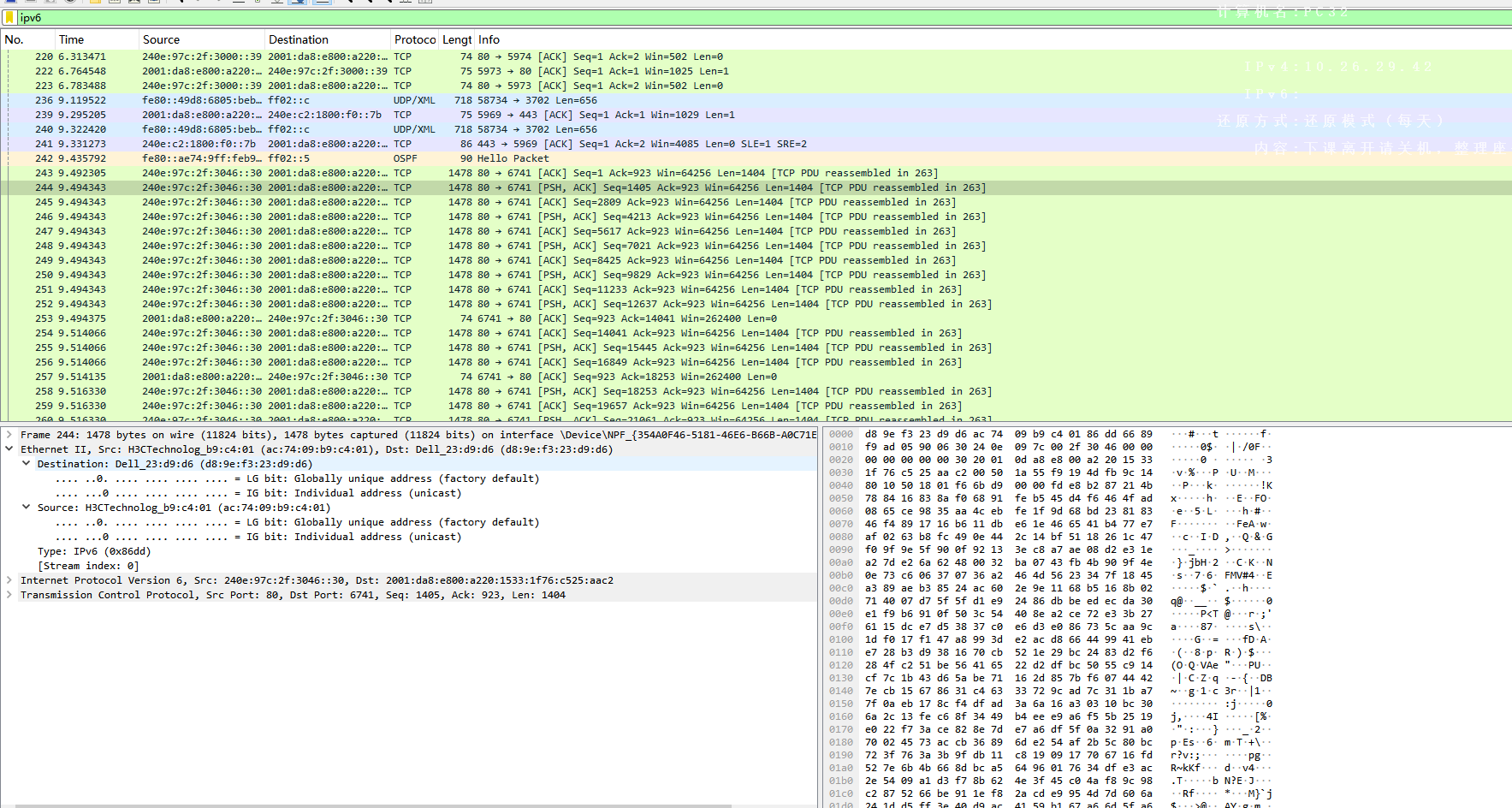
（12） 目的地址（占32位）：db e5 51 c8;左侧显示219.229.81.200（对应点分十进制）



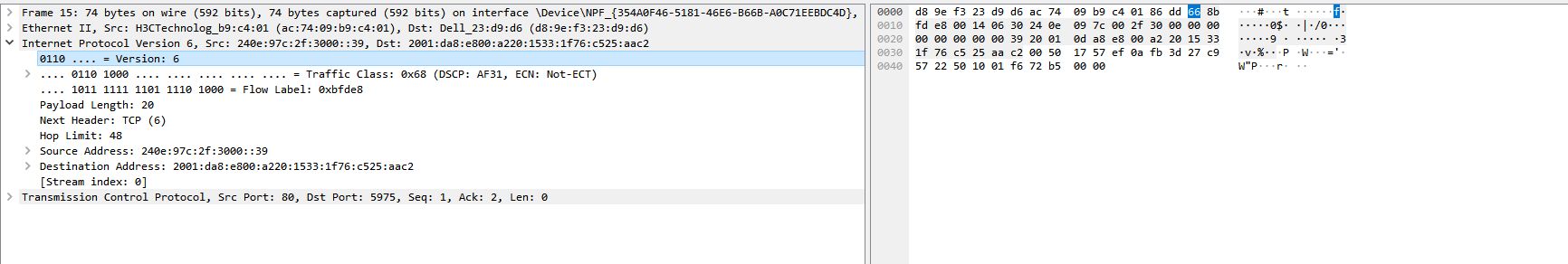
#### 1.2.2IPv6数据报

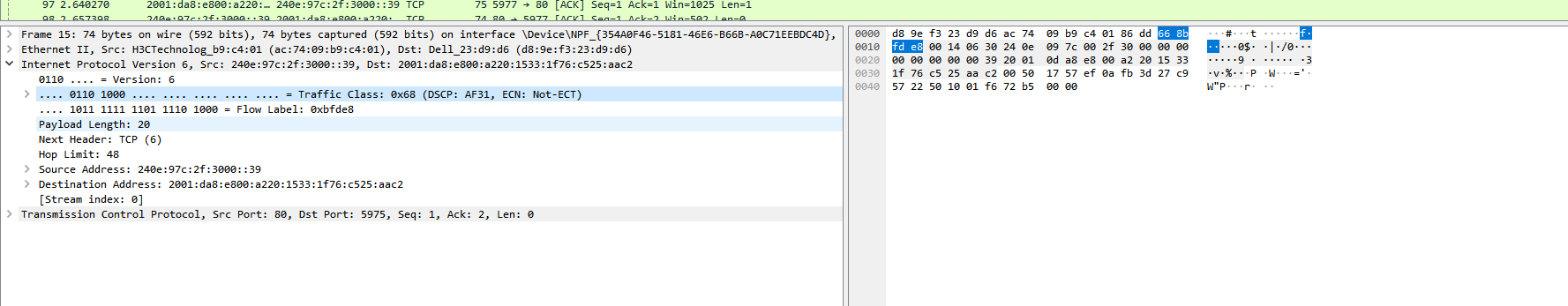


（1）在Ethrnet II中看见这个数据报是IPv6



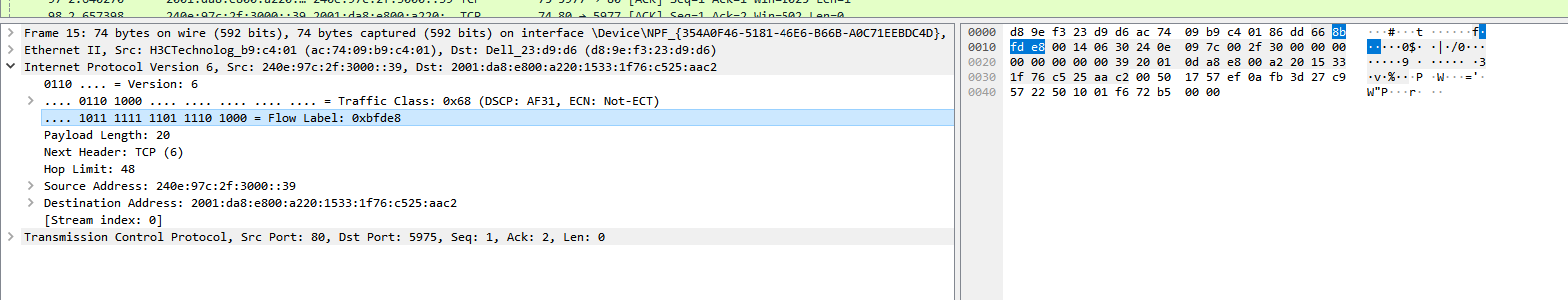
（2）版本（4位）：0110，十六进制表示为6，代表版本为6

（3）流量等级（12位）：0110 1000,即0x68，标识数据报的优先级

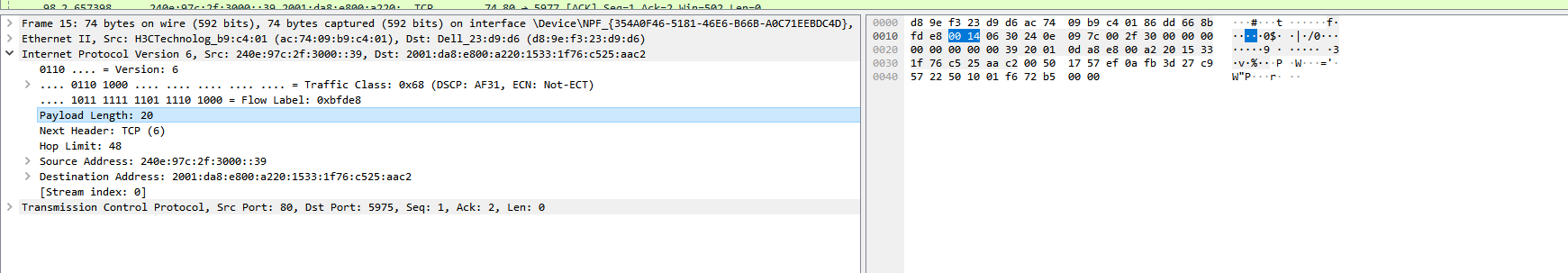


（4）ECN(2位)：10,Not-ECT

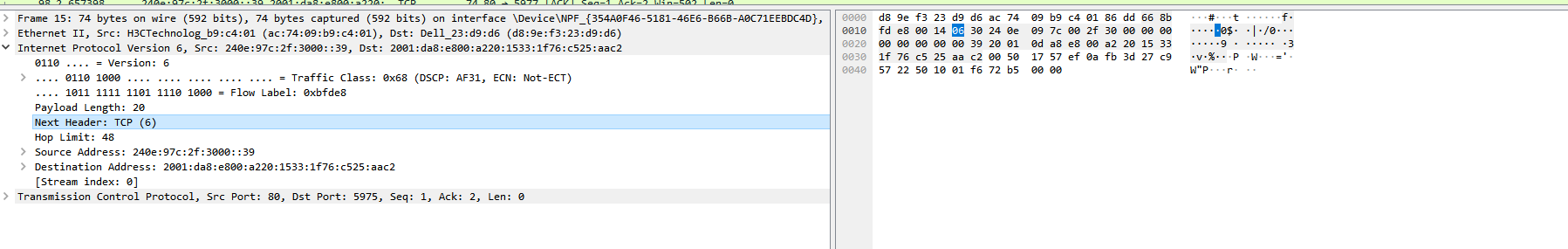
（5）流标签（20位）：0xbfde8, 标识同一“流”中的数据报。流就是从特定的主机出发到特定的目的的一系列的数据报。这一系列数据报的流标签是相同的



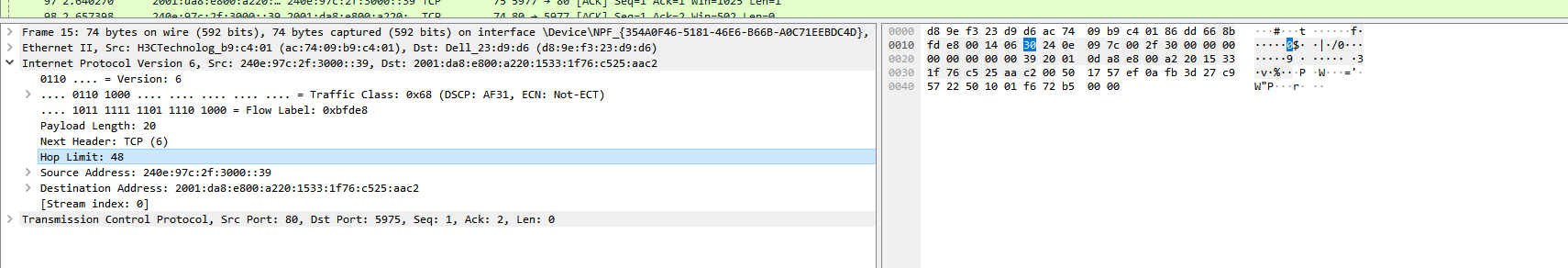
（6）有效载荷长度（16位）：00 14，十进制大小为20，即有效载荷长度为20



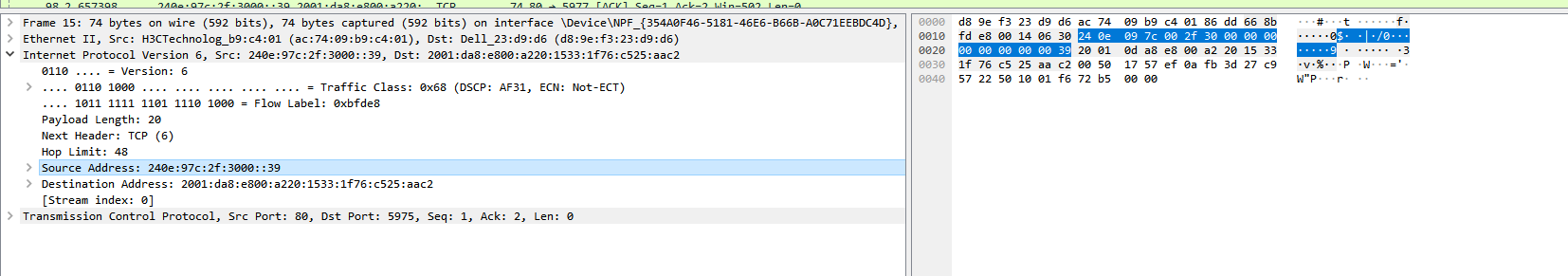
（7）下一个首部（8位）：06，下一个首部为TCP（6）



（8）跳数限制（8位）：30，即跳数限制为48



（9）源IP地址（128位）：24 0e 09 7c 00 2f 30 00 00 00 00 00 00 00 00 39



（10）目的IP地址（128位）：20 01 0d a8 e8 00 a2 20 15 33 1f 76 c5 25 aa c2



#### **IP数据报分片**

Ping命令

选项:

-t Ping 指定的主机，直到停止。

若要查看统计信息并继续操作，请键入 Ctrl+Break；

若要停止，请键入 Ctrl+C。

-a 将地址解析为主机名。

**-n count 要发送的回显请求数。**

**-l size 发送缓冲区大小。**

**-f 在数据包中设置“不分段”标记(仅适用于 IPv4)。**

-i TTL 生存时间。

-v TOS 服务类型(仅适用于 IPv4。该设置已被弃用，

对 IP 标头中的服务类型字段没有任何

影响)。

-r count 记录计数跃点的路由(仅适用于 IPv4)。

-s count 计数跃点的时间戳(仅适用于 IPv4)。

-j host-list 与主机列表一起使用的松散源路由(仅适用于 IPv4)。

-k host-list 与主机列表一起使用的严格源路由(仅适用于 IPv4)。

-w timeout 等待每次回复的超时时间(毫秒)。

-R 同样使用路由标头测试反向路由(仅适用于 IPv6)。

根据 RFC 5095，已弃用此路由标头。

如果使用此标头，某些系统可能丢弃

回显请求。

-S srcaddr 要使用的源地址。

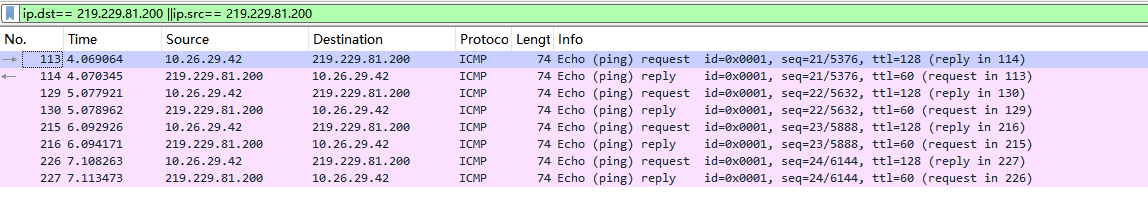
-c compartment 路由隔离舱标识符。

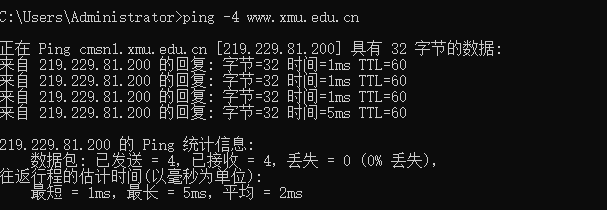
-p Ping Hyper-V 网络虚拟化提供程序地址。

**-4 强制使用 IPv4。**

**-6 强制使用 IPv6。**

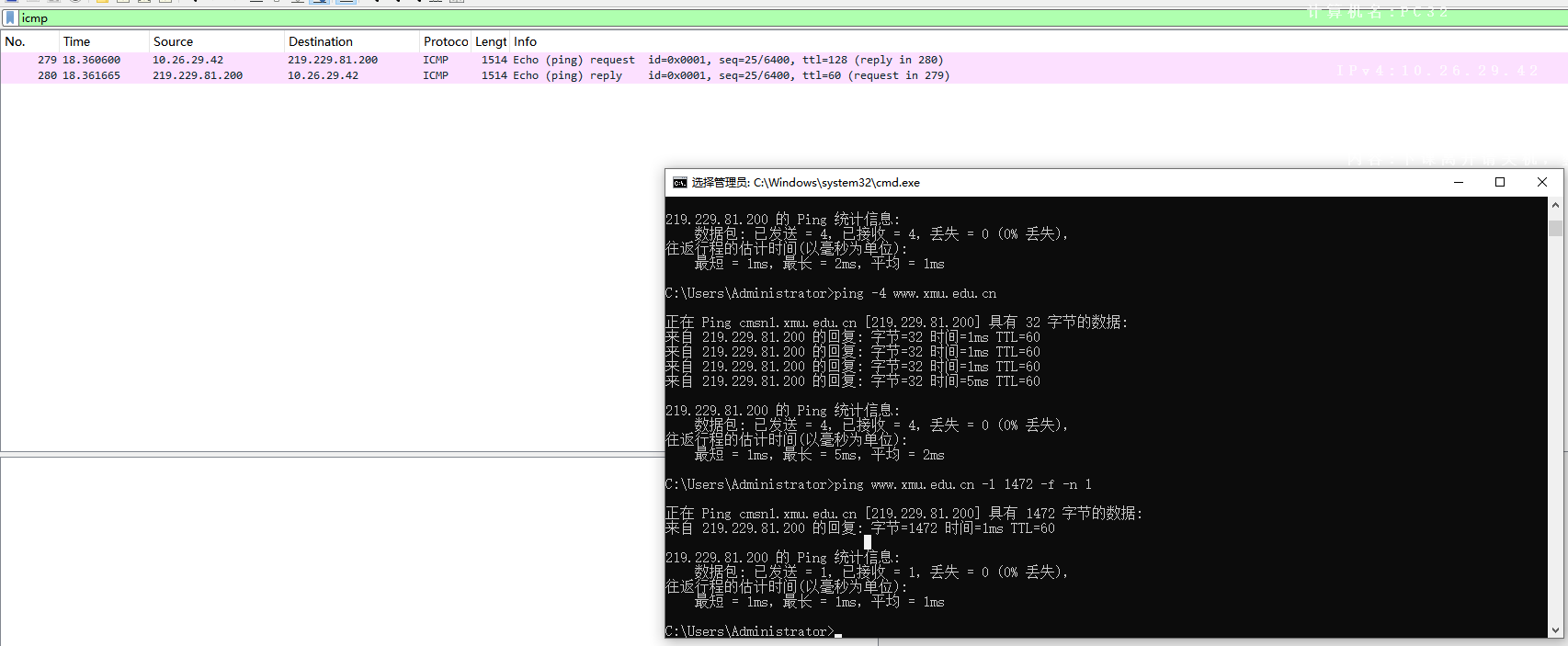
##### （a）**ping -4 www.xmu.edu.cn**

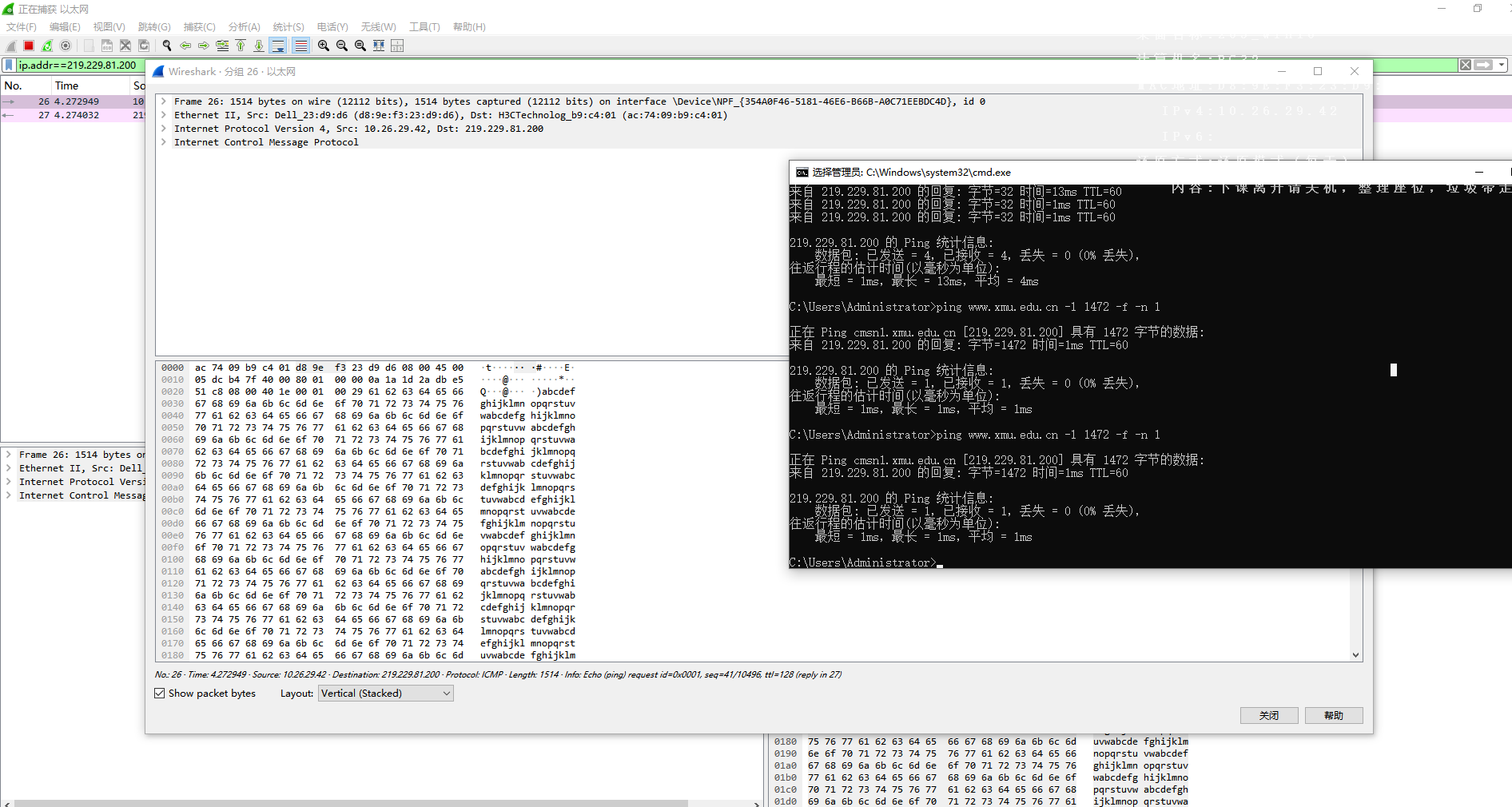




可以发现，这个 命令发出了 4 个 ICMP 包，根据抓包结果，每个包发送了 52 字节的数据，去掉 20 字节的首部长度，数据部分发送了 **32** 字节，与终端中所示相符。

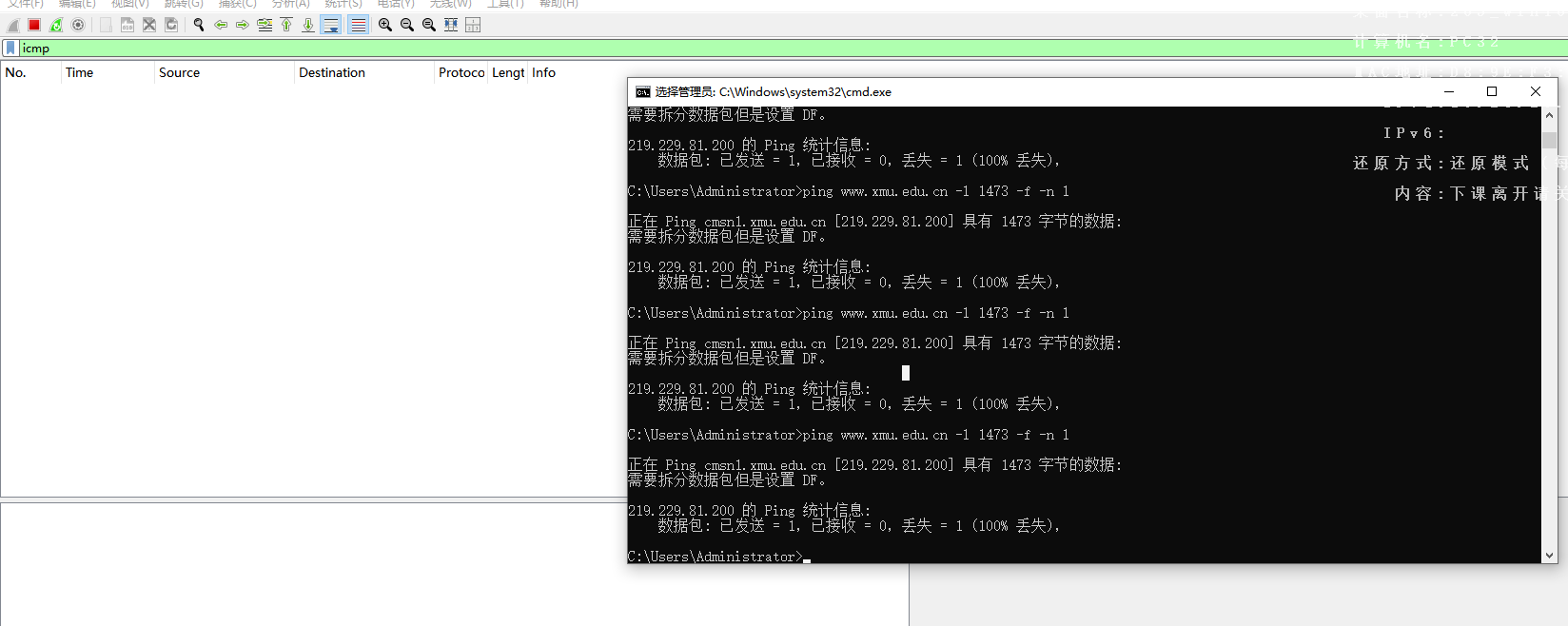
##### （b）**ping www.xmu.edu.cn -l 1472 -f -n 1**





可以发现，这个命令发送了一个 ICMP 包，每个包的总长度刚好为 1500，数据部分长度为 1472。这个长度刚好到达了以太网的 MTU 限制。

##### （c）**ping www.xmu.edu.cn -l 1473 -f -n 1**



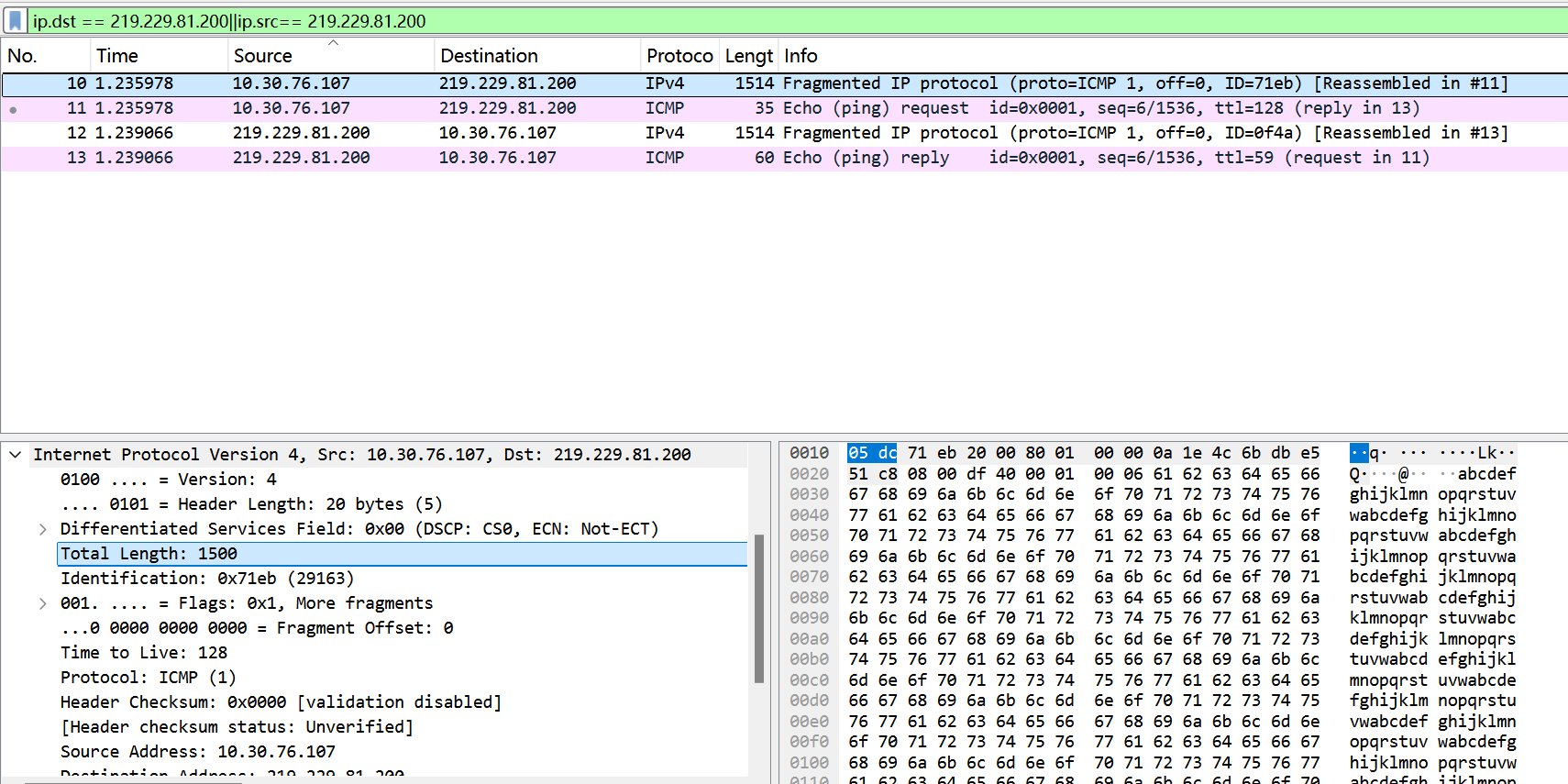
可以发现，在禁止分片的情况下，发送 1473 字节的数据报根本无法发送。

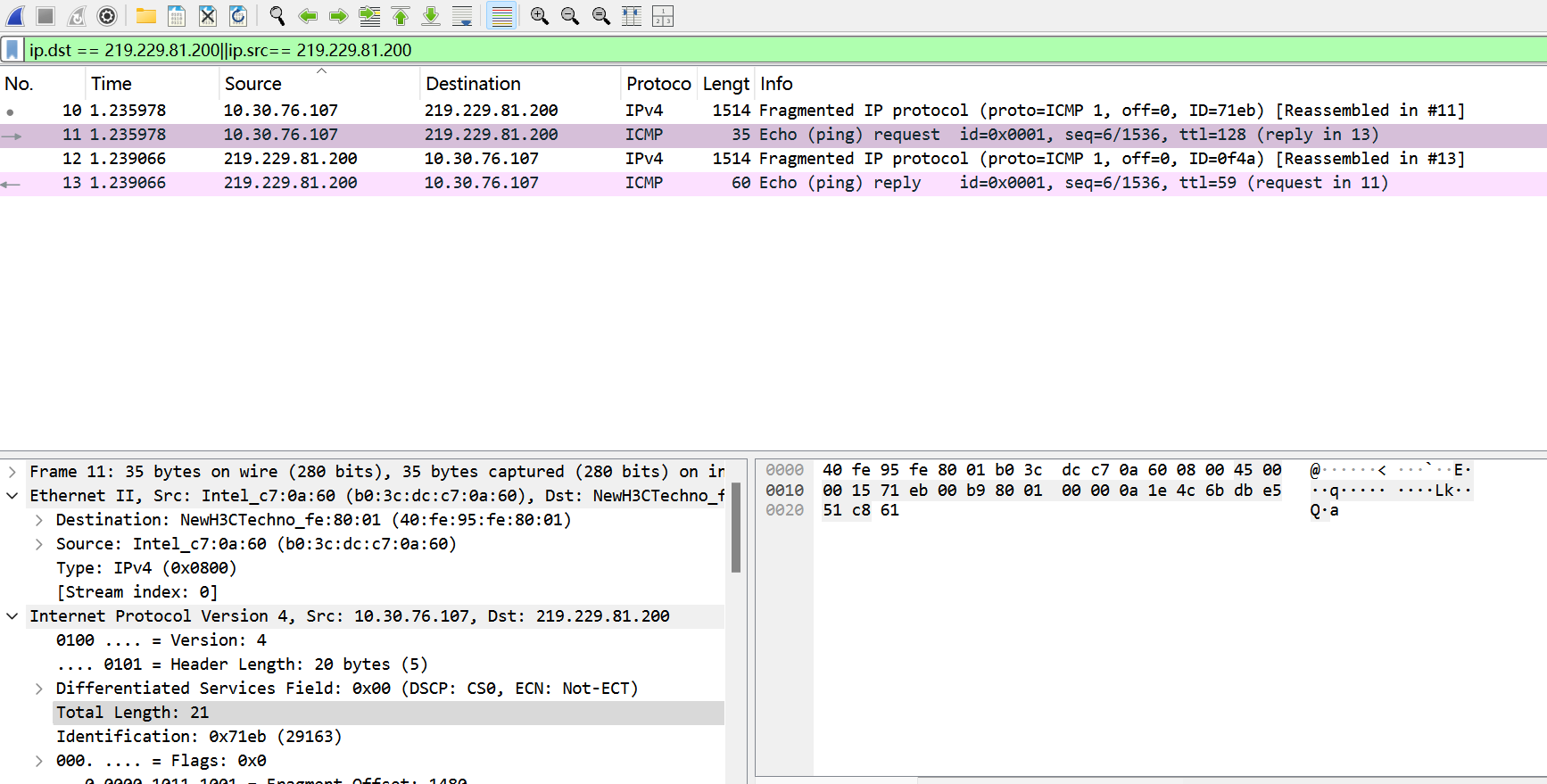
（1473 字节数据 + ICMP 头部 8 字节 + IP 头部 20 字节，总共 1501 字节）

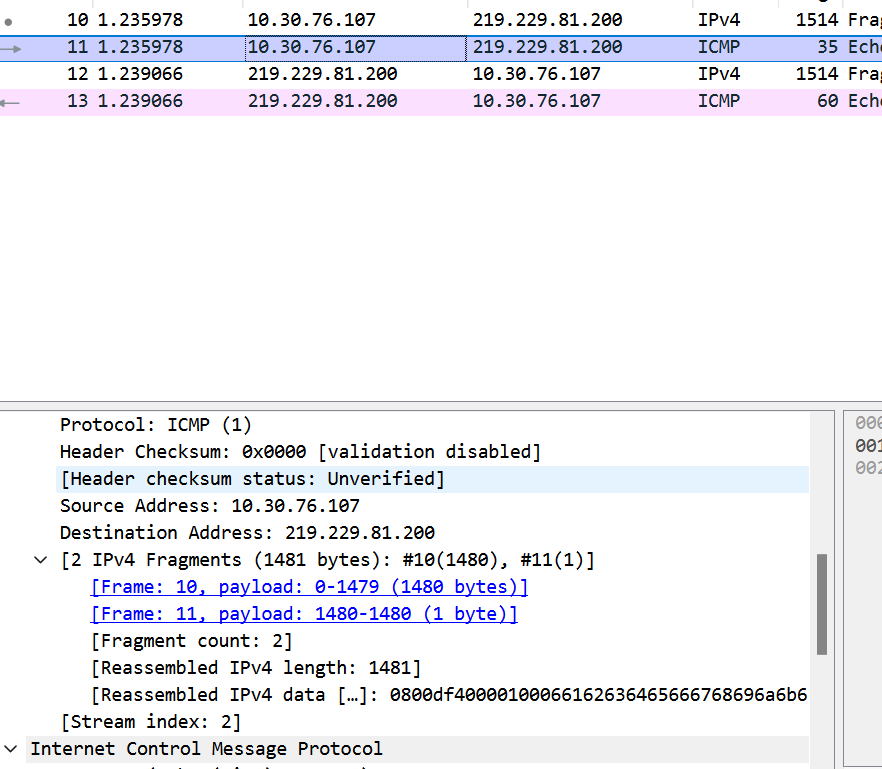
而以太网MTU为1500

##### （d)**ping www.xmu.edu.cn -l 1473 -n 1**

##### （要使发出icmp而不是icmpv6命令可以**为ping -4 www.xmu.edu.cn -l 1473 -n 1**





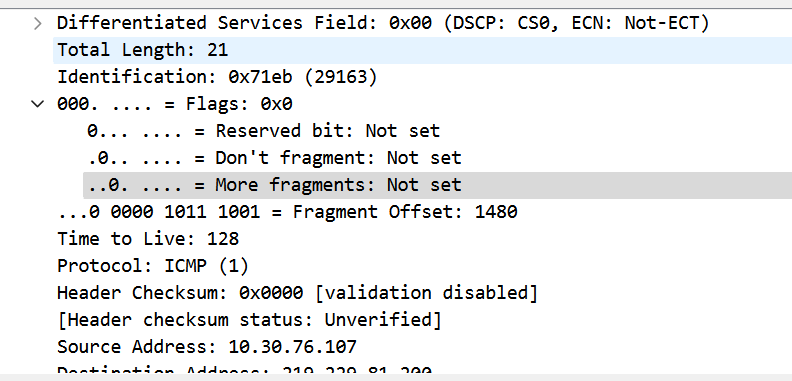
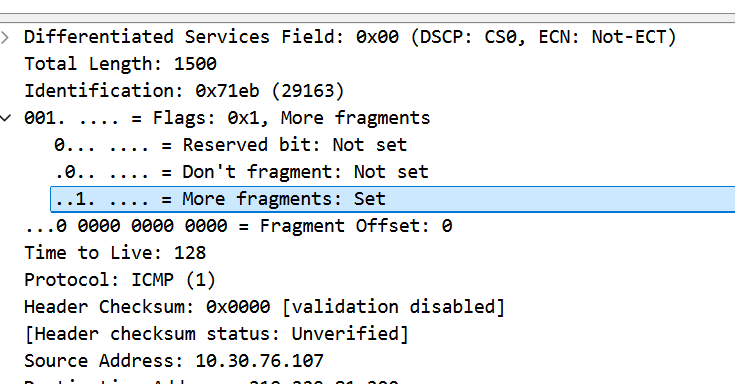


在左下角ipv4的字段中发现[2 IPv4 Fragments (1481 bytes): #12(1480), #13(1)]

这条信息是一个网络分段的日志记录，表示一个IPv4数据包被分成了两个分片（fragments）。这通常发生在数据包大小超过网络的最大传输单元（MTU）时，以便于数据包可以在低于MTU的链路上传输。

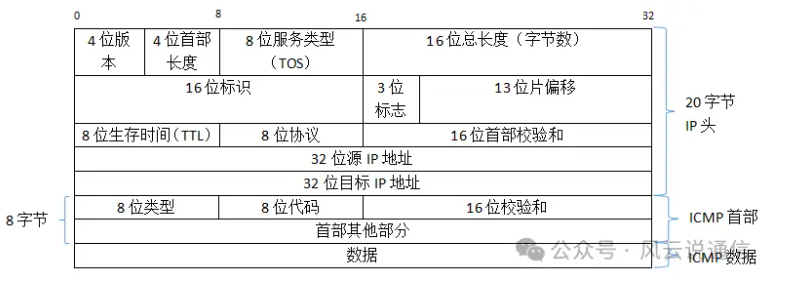
[2 IPv4 Fragments (1481 bytes): #12(1480), #13(1)]：表示这是两个IPv4分片，总共包含1481字节，第一个分片（#12）包含1480字节，第二个分片（#13）包含1字节。

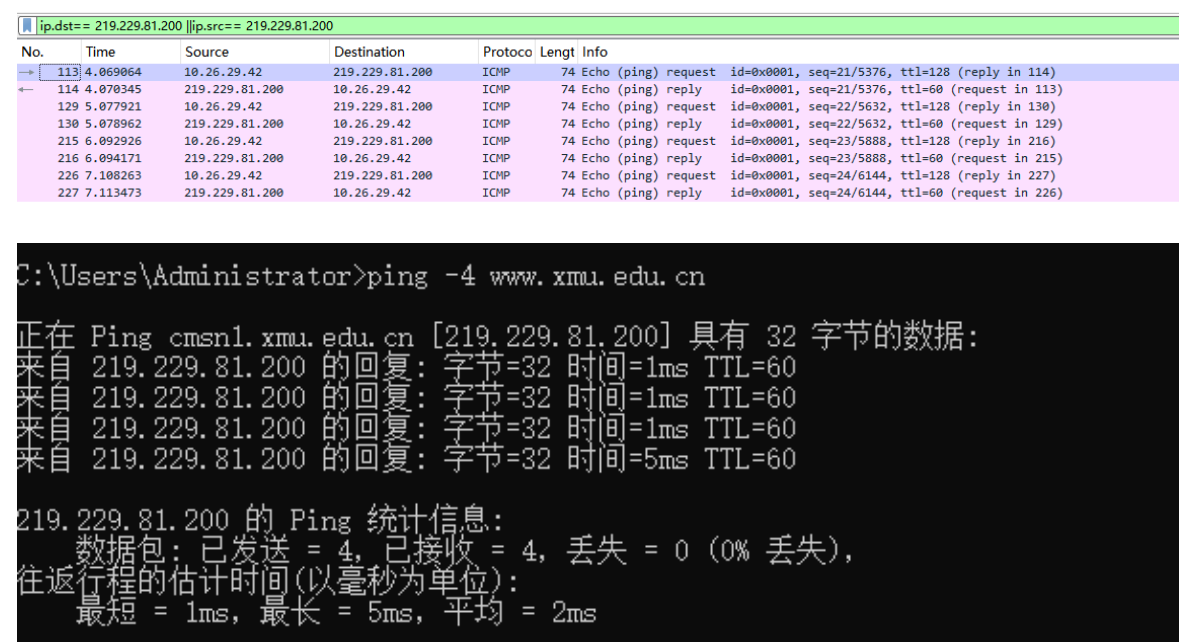
结合捕获到的ip.dst == 219.229.81.200的数据报共有两条，一条为长度为1500的ipv4数据报和一条长度为21的icmp数据报，即一条数据长度为1500-20-8=1472，其标志字段为 0x1，代表后面还有分片。其片偏移为0，表示这是第一个分片。一条数据长度为21-20=1，其标志字段为 0x0，表示后面没有分片了。其片偏移为1480，代表这个分片从第1481个字节开始。



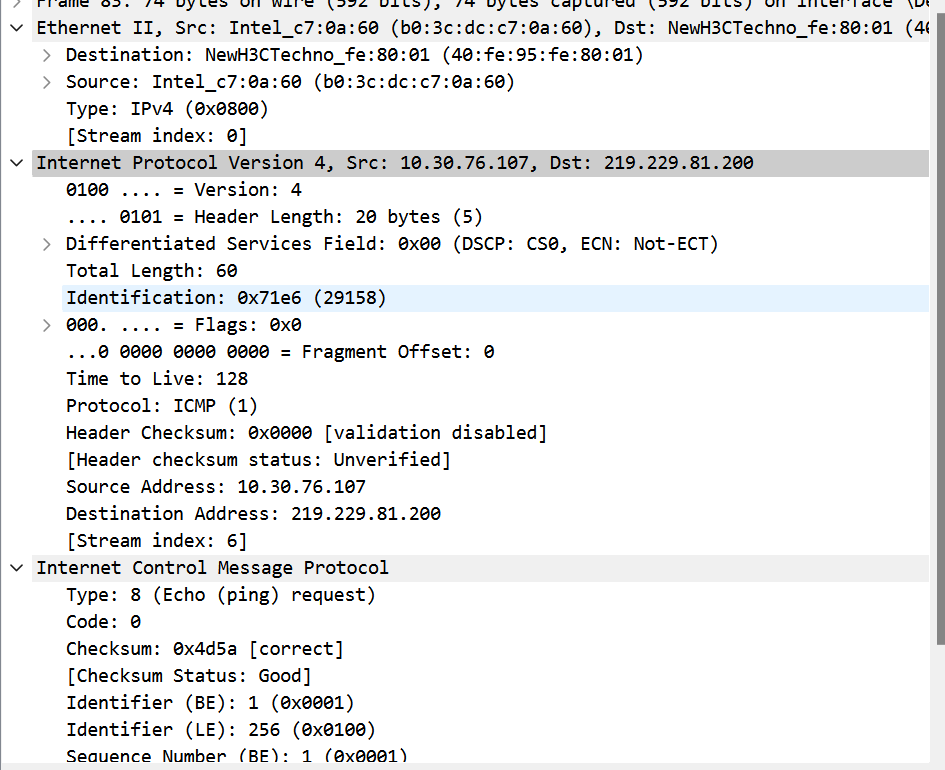
#### 1.4 ICMP报文

图文结合 ，说明执行 一 次 ping 命令 ，共有 几个ICMP 请求帧和回应帧，比较请求帧和回应帧的差别及其对应IP头部的变化。

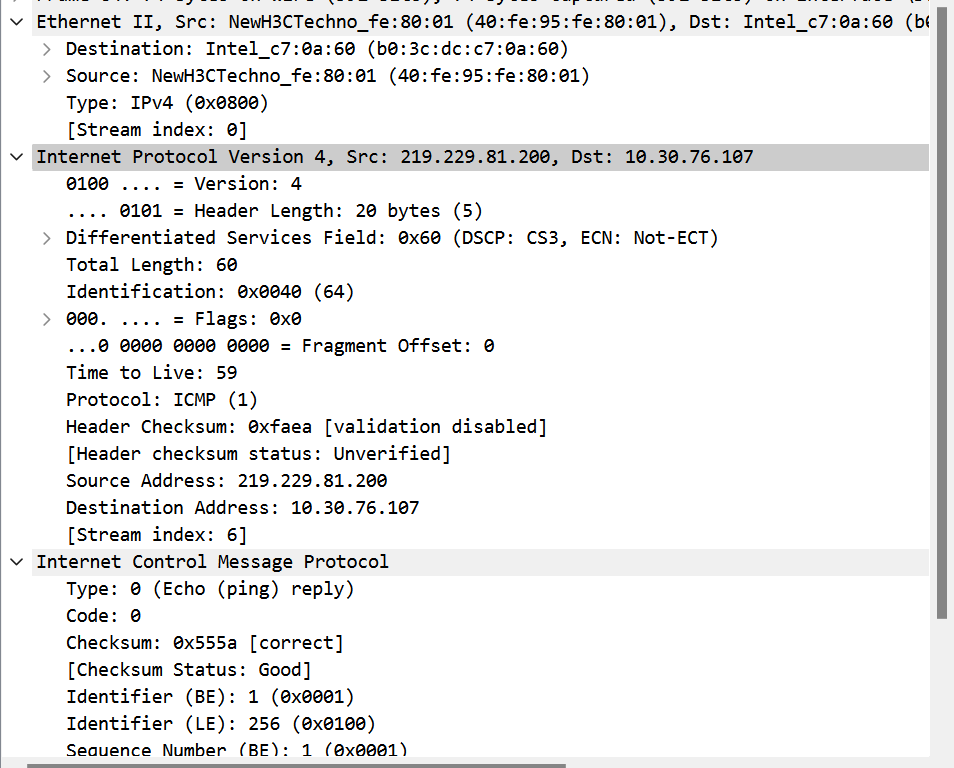




**ping -4 www.xmu.edu.cn** 后有4个ICMP请求帧4个ICMP回应帧



**请求帧**



**回应帧**

通过对比可以发现，ICMP 报文的内容（也就是对应 IP 数据报的数据部分），只有两个地方有所差别：**Type** 和 **Checksum**。**Checksum** 应该是 ICMP 报文的校验和，因此本质上只有 **Type** 有区别，也就是 **ping** 对应的 ICMP 请求帧和回应帧，除了校验和，在数据部分只有类型不一样，请求帧type为request,回应帧type为reply,其余全部相同。

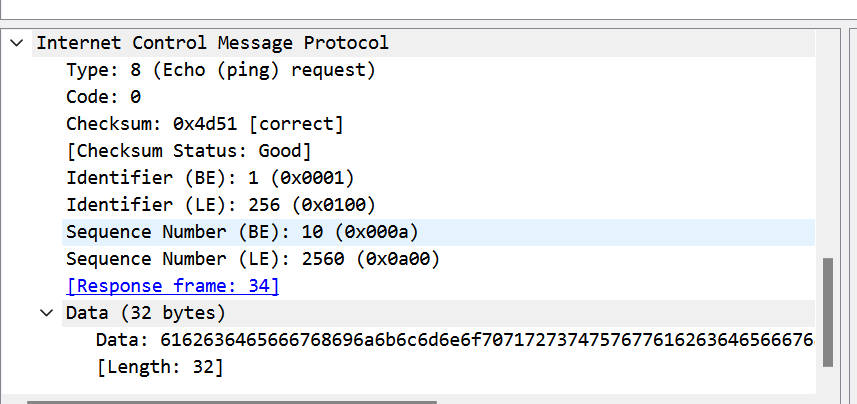
接下来看 IP 数据报的首部，通过对比可以发现只有标识字段、校验和、TTL 和发送地址、接收地址不同。发送和接收的 IP 地址发生了交换，这是很容易理解的。**TTL 不同是因为这里监听到的请求帧是刚发出的，而收到的回应帧是经过了好几个路由器转发过来的。**标识字段不同也很正常，这是两个不同主机上不同的计数器产生的值。校验和字段不同是因为别的首部字段不同。值得注意的是，IP 数据报的总长度字段也是相同的，这说明发送和接收的 ICMP 请求/回应帧的数据格式其实是相同的。

最后是 MAC 帧的首部。MAC 帧的首部只有 MAC 地址的源地址和目的地址发生了交换，而代表上层协议的字段相同，都是 IP 数据报。

##### 对icmp部分的具体分析：

1. 类别（8位）：0为reply,8为request

（2）代码（8位）：此处为0，主要是指，不同的ICMP报文种类具体有哪些错误



（3）校验和（16位）：0xdf40,根据具体首部不同而不同

（4）首部其他部分（32位）：图中有Identifier（标志符）和Sequence Number（序列号）

Identifier（BE）指的是标示符（大端顺序）：1（0x0100）；

Identifier（LE）指的是标示符（小端顺序）：256（0x0100）；

Sequence number(BE)指的是序列号（大端顺序）：10（0x000a）

Sequence number(LE)指的是序列号（小端顺序）：2560(0x0a00)

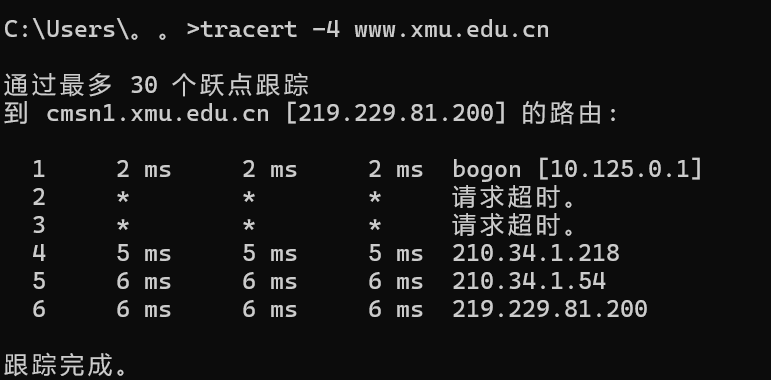
（5）数据：此处数据有32位

#### 1.5 tracert命令

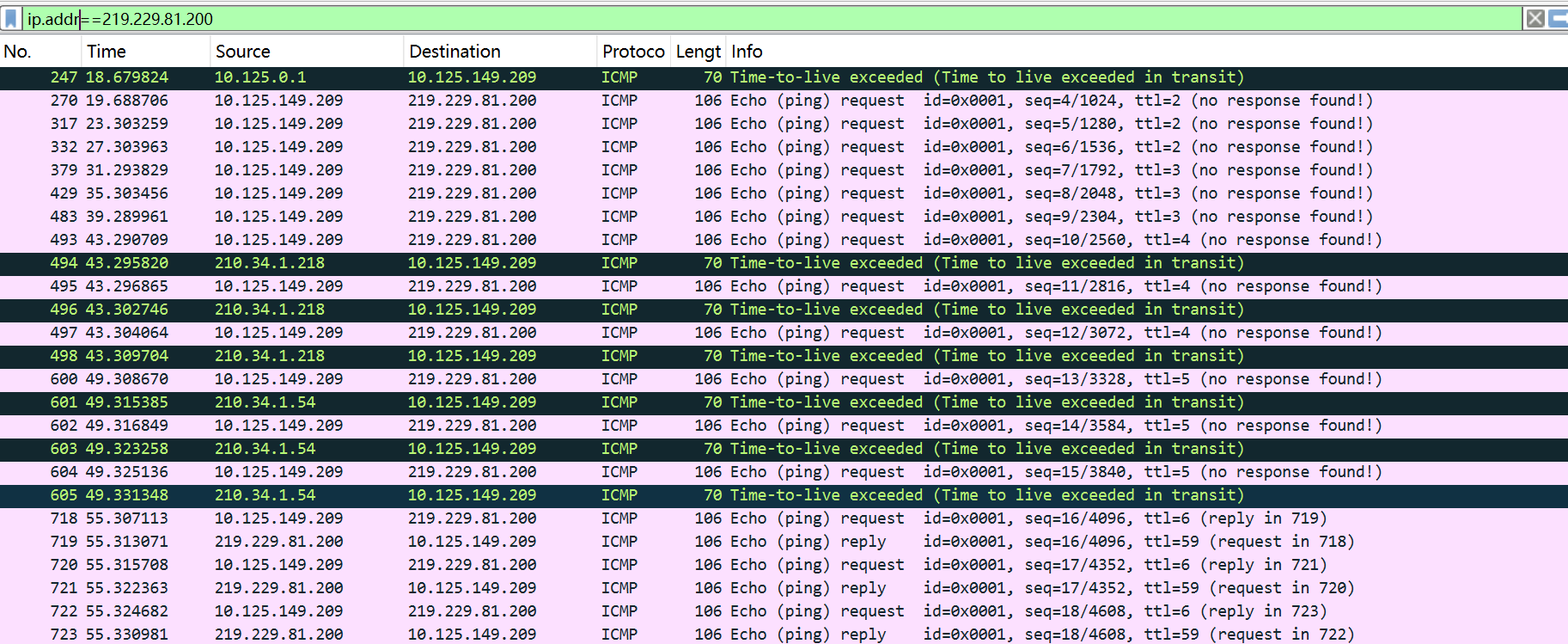
图文结合，描述 tracert 工作原理，画出示意图。

命令（-4是为了指定ipv4和icmp）：**tracert -4 [www.xmu.edu.cn](http://www.xmu.edu.cn)**

（1）抓包，在终端发起网络命令：ping 36.152.44.96，停止抓包，保存



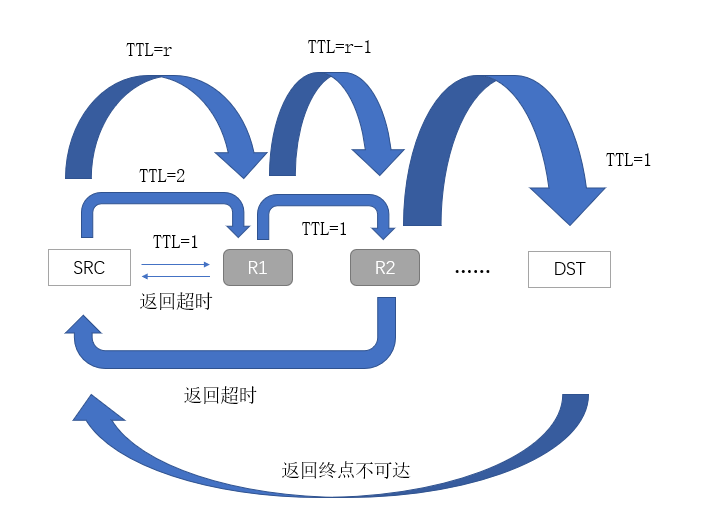
（2）设置过滤条件icmp，显示所捕获的ICMP数据包



（3）描述tracert工作原理，画出示意图

tracert从源主机向目的主机发送一连串的IP数据报，数据报中封装的是无法交付的UDP用户数据报。每个路由器在收到数据包后会将TTL值减一，当TTL为0时，路由丢弃数据包，并向源主机发送一个ICMP时间超过的差错报文。当最后一个数据报刚刚到达目的主机时，目的主机向源主机发送ICMP终点不可达差错报告报文。tracert先发送TTL为1的Echo数据包，然后在后来的发送过程中不断让TTL的值加1,直到目标相应或TTL值达到最大，这样就能获得路径上所有路由的IP地址。

示意图如下所示：



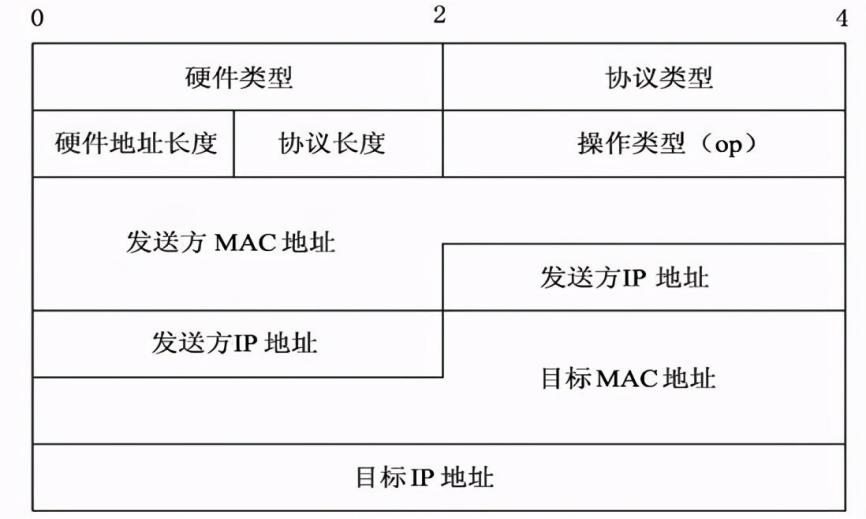
注：对应于每个TTL值，源主机要发送三次同样的IP数据报，但是为了示意图的简洁性，对应于每个TTL值，只画出了一次。

#### 1.6.1 ARP协议分析

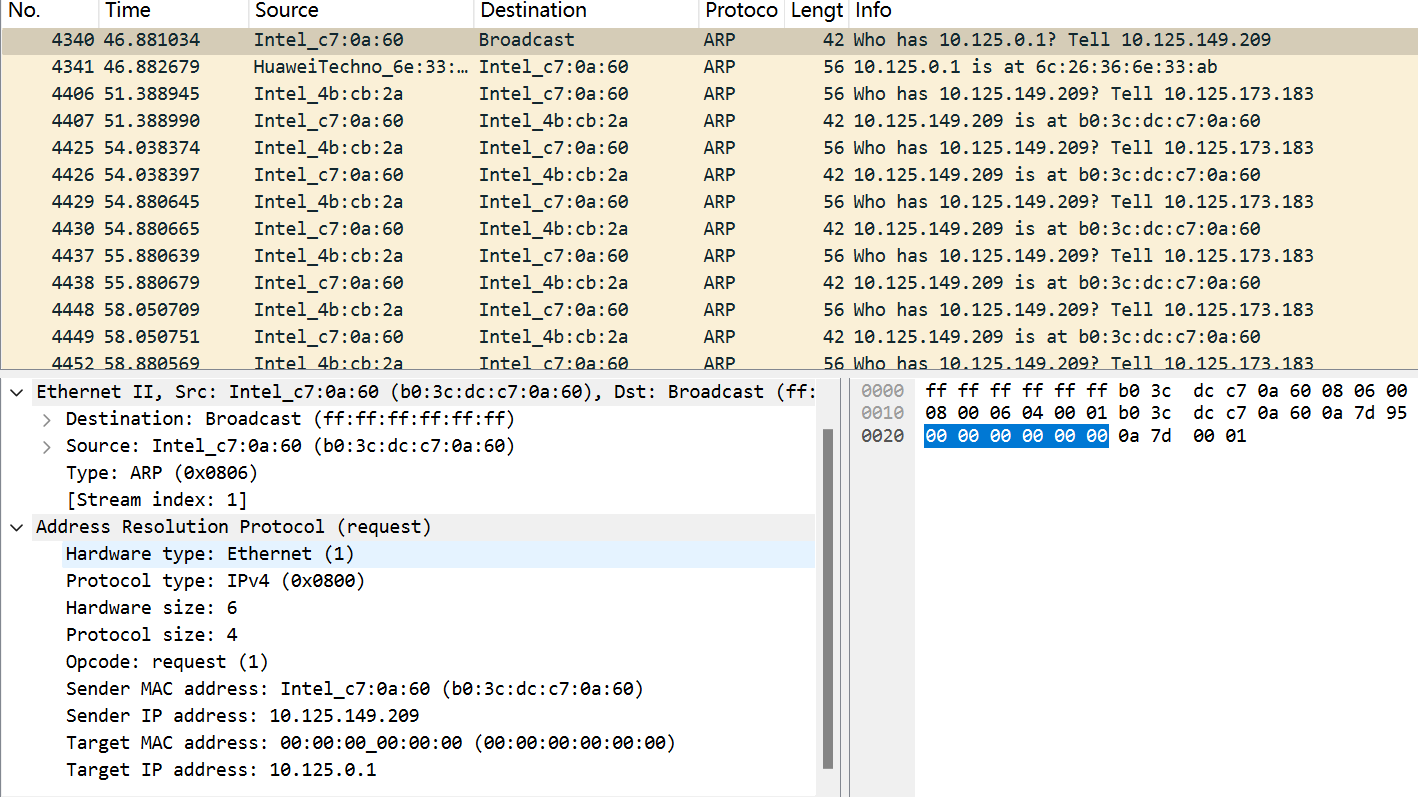
(1）以管理员身份运行终端窗口，运行arp –d命令，清空本机已有的ARP缓存；抓包，先ping旁边同学的ip，再ping www.baidu.com,停止抓包，保存。



（2）ARP报文



（3）Ping www.baidu.com的ARP数据报

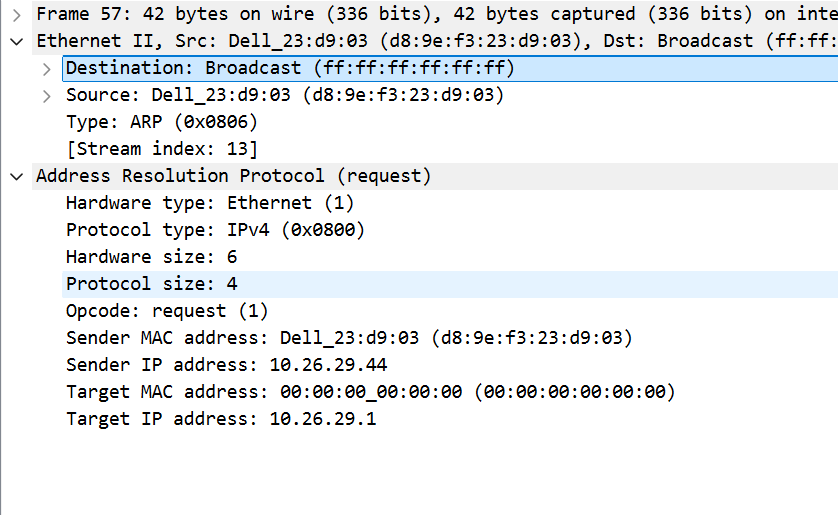


在发出对[www.baidu.com的ping](http://www.baidu.com的ping)命令之前，先询问“who has 10.125.0.1? tell 10.125.149.209”，先找到网关的mac地址，即距离最近的路由器的mac地址：ac:74:09:b9:c4:01；然后看到ping了四条ICMP,可见成功通过路由器将本网域的数据报转发到了另一个网域进行通信。ICMP的mac目的地址为路由器 10.125.0.1发出。

|  |  |  |
| --- | --- | --- |
|  | 内容 | 备注 |
| 以太网目的地址 | ff ff ff ff ff ff | 目的地址的硬件地址（Broadcast） |
| 以太网源地址 | b0:3c:dc:c7:0a:60 | 发送者的硬件地址 |
| 帧类型 | 08 06 | 类型是arp包 |
| 硬件类型 | 00 01 |  |
| 协议类型 | 08 00 | IPv4 |
| 硬件地址长度 | 06 |  |
| 协议地址长度 | 04 |  |
| 操作码 | 00 01 | 00 01表示请求包，00 02表示应答包 |
| 发送者MAC地址 | b0:3c:dc:c7:0a:60 |  |
| 发送者IP地址 | 10.125.149.209 |  |
| 目标MAC地址 | 00：00:00——00:00:00 | 此时发送端不知道目标的MAC地址，要从响应包中获取目标的MAC地址 |
| 目标IP地址 | 10.125.0.1 |  |

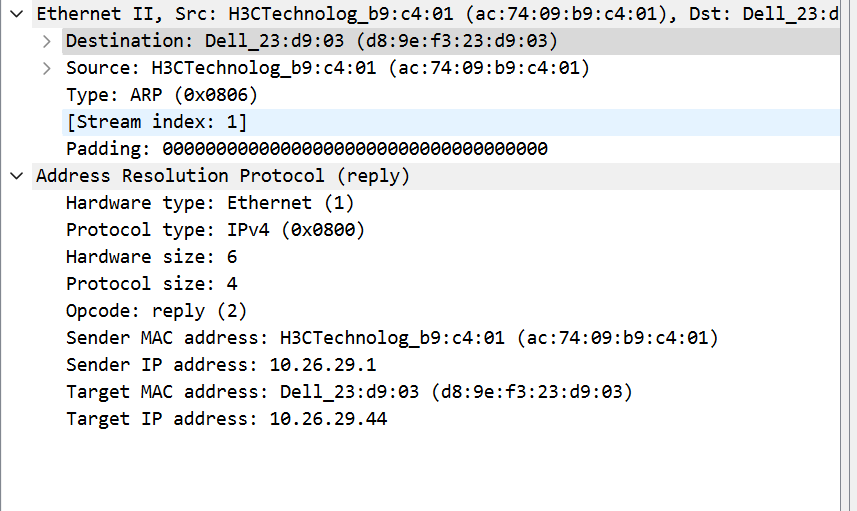
（4）ping旁边同学的ip的结果，分别对request数据报和reply数据报进行分析

A.request数据报



|  |  |  |
| --- | --- | --- |
|  | 内容 | 备注 |
| 以太网目的地址 | ff ff ff ff ff ff | 目的地址的硬件地址（Broadcast） |
| 以太网源地址 | D8 9e f3 23 d9 03 | 发送者的硬件地址 |
| 帧类型 | 08 06 | 类型是arp包 |
| 硬件类型 | 00 01 |  |
| 协议类型 | 08 00 | Ipv4 |
| 硬件地址长度 | 06 |  |
| 协议地址长度 | 04 |  |
| 操作码 | 00 01 | 00 01表示请求包，00 02表示应答包 |
| 发送者MAC地址 | d8 9e f3 23 d9 03 |  |
| 发送者IP地址 | 0a 1a 1d 2c |  |
| 目标MAC地址 | 00 00 00 00 00 00 | 此时发送端不知道目标的MAC地址，要从响应包中获取目标的MAC地址 |
| 目标IP地址 | 0a 1a 1d 2c |  |

B.reply数据报

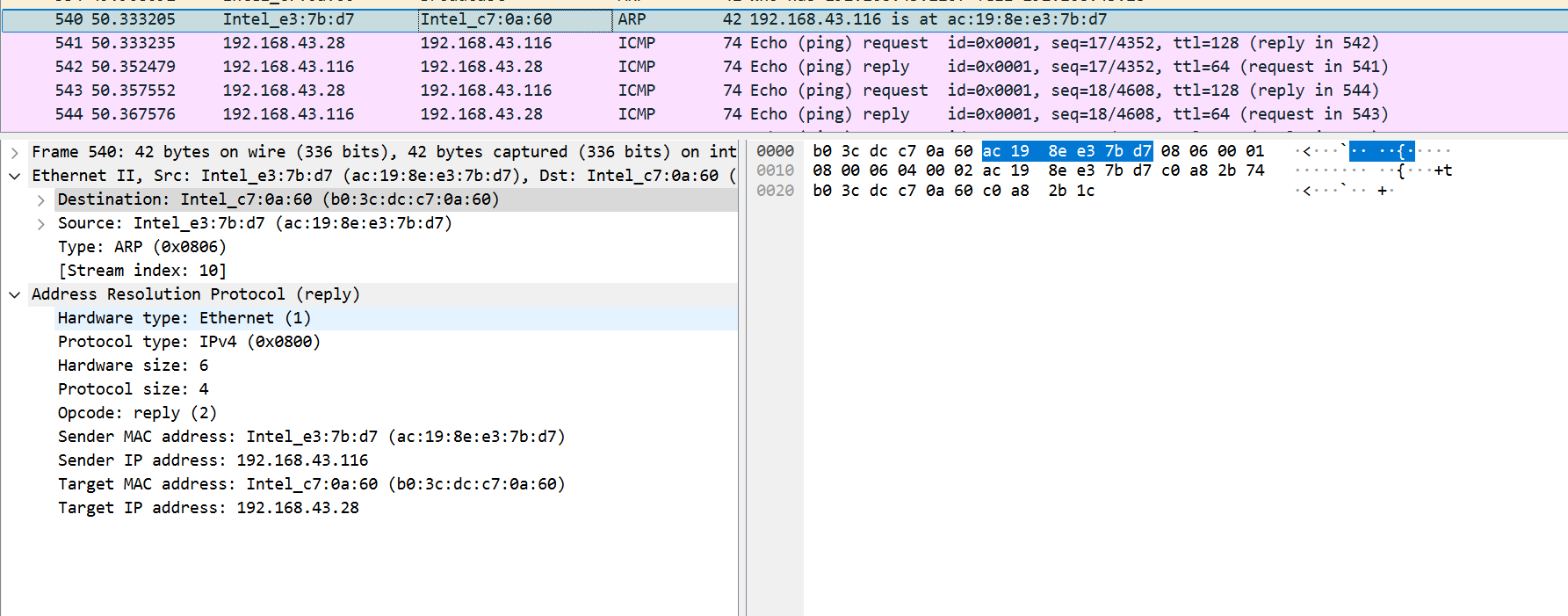


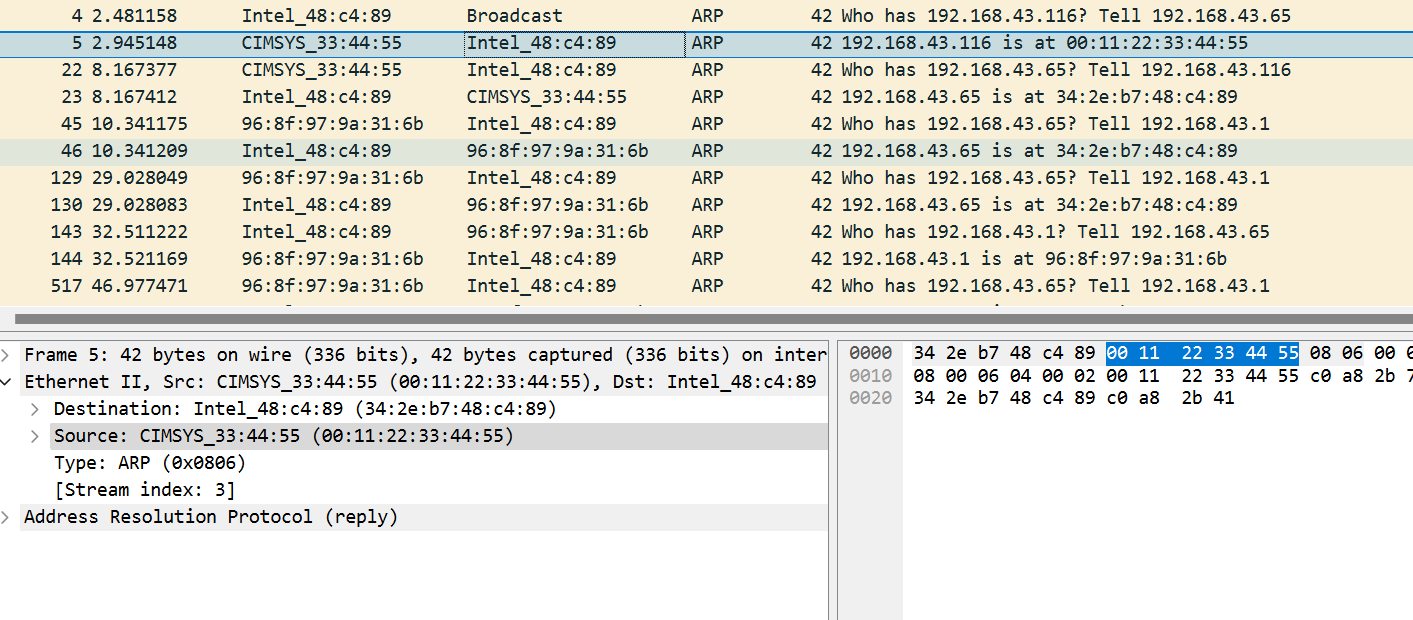
|  |  |  |
| --- | --- | --- |
|  | 内容 | 备注 |
| 以太网目的地址 | d8 9e f3 23 d9 03 | 目的地址的硬件地址（Broadcast） |
| 以太网源地址 | Ac 74 09 b9 c4 01 | 发送者的硬件地址 |
| 帧类型 | 08 06 | 类型是arp包 |
| 硬件类型 | 00 01 |  |
| 协议类型 | 08 00 | Ipv4 |
| 硬件地址长度 | 06 |  |
| 协议地址长度 | 04 |  |
| 操作码 | 00 02 | 00 01表示请求包，00 02表示应答包 |
| 发送者MAC地址 | ac 74 09 b9 c4 01 |  |
| 发送者IP地址 | 0a 1a 1d 01 |  |
| 目标MAC地址 | D8 9e f3 23 d9 03 | 此时发送端不知道目标的MAC地址，要从响应包中获取目标的MAC地址 |
| 目标IP地址 | 0a 1a 1d 2c |  |

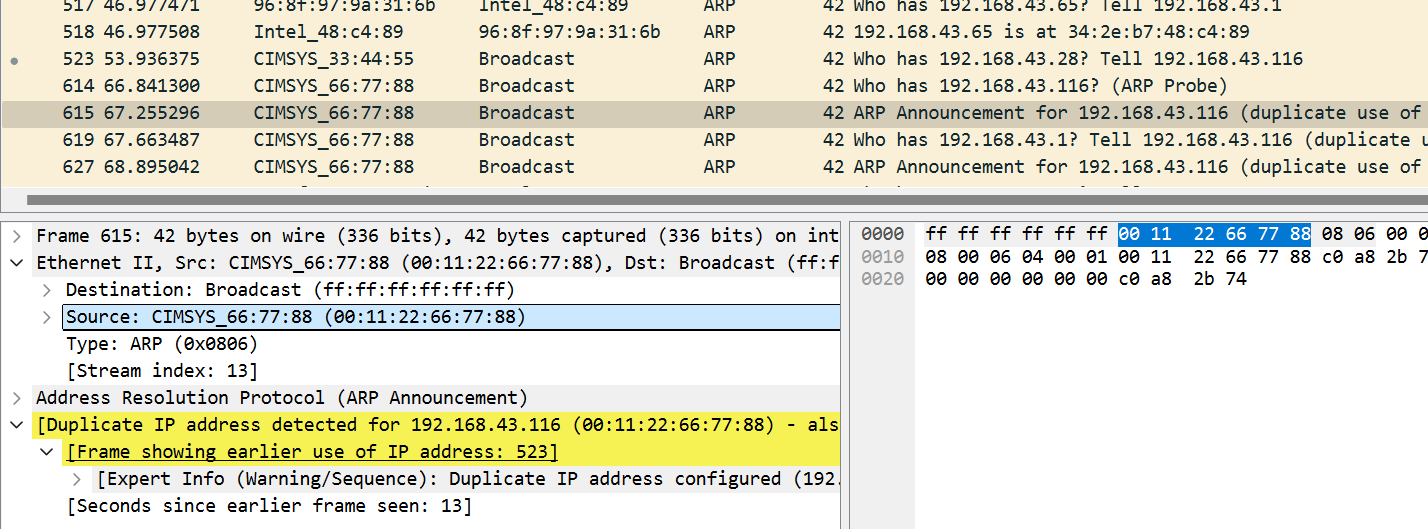
（5）ping同一局域网内的计算机和局域网外的计算机，产生的ARP请求的不同：

不论是ping一个局域网内还是局域网外的计算机，最后主机收到的arp都是由局域网内的主机发送过来的。ping一个局域网内的主机，会直接得到该主机的回复，从而获得目的主机的MAC地址。ping一个局域网外的主机，得到的是与源主机在同一个局域网上的路由器的MAC地址。因此ARP只能用于解决同一个局域网上的主机或路由器的IP地址和MAC地址的映射问题。

#### 1.6.2修改电脑的mac地址观察mac帧







像ip地址为192.168.43.116前后发送两次arp，mac地址从00:11:22:33:44:55变成00:11:22:66:77:88,并且arp警告ip地址被重复使用了。

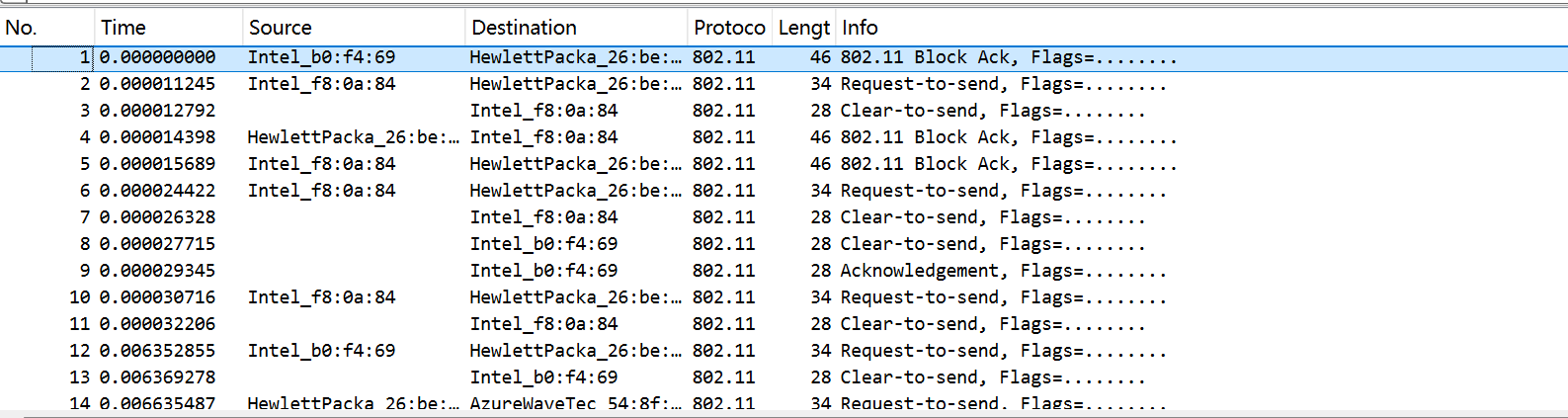
### 任务 2 捕获和分析802.11数据

#### 说明：

**802.11 无线 WiFi 有 数据帧，管理帧、[控制帧](https://zhida.zhihu.com/search?content_id=248515510&content_type=Article&match_order=1&q=%E6%8E%A7%E5%88%B6%E5%B8%A7&zhida_source=entity" \t "https://zhuanlan.zhihu.com/p/_blank) 三种类型帧。**

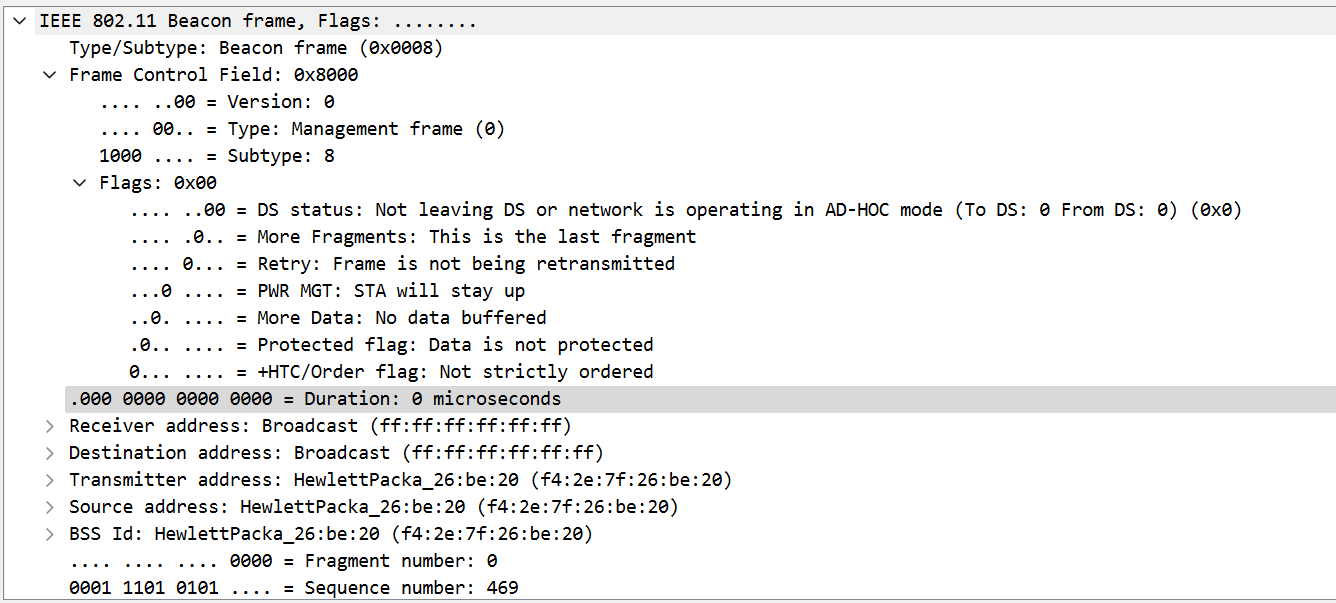
**其中控制帧分为RTS\CTS\ACK**





总体得到的数据报

#### 管理帧：



帧类型：0x0008

持续时间：0000

接收地址：ff ff ff ff ff ff

目的地址:ff ff ff ff ff ff

发送地址：f4 2e 7f 26 be 20

源地址：f4 2e 7f 26 be 20

BSS Id :f4 2e 7f 26 be 20

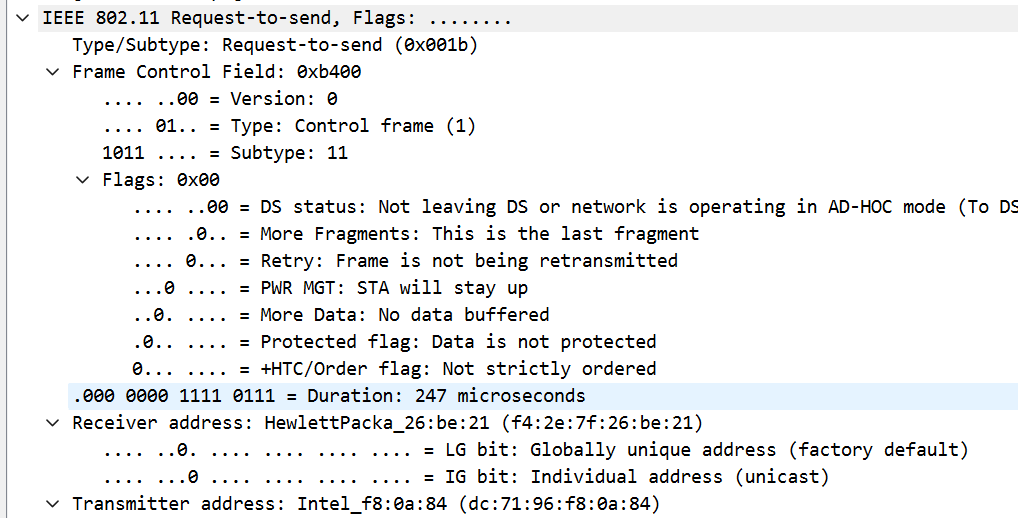
分片号:50 1d

#### （3）控制帧：RTS\CTS\ACK

##### ①RTS

RTS帧格式如下图所示：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 帧控制（2字节） | 持续期（2字节） | 接收地址（6字节） | 发送地址（6字节） | FCS（4字节，不显示） |



类型为0x001b,代表RTS帧，持续期为f700,接收地址为f4 2e 7f 26 be 21

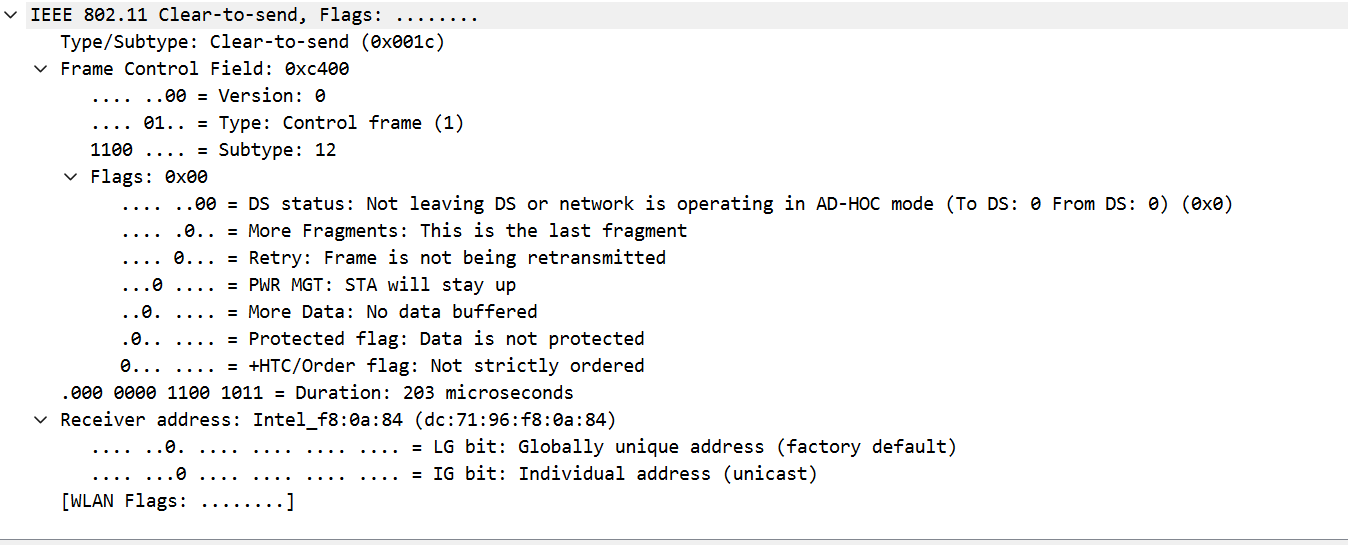
发送地址为dc 71 96 f8 0a 84

##### ②CTS



CTS帧格式如下图所示：

|  |  |  |  |
| --- | --- | --- | --- |
| 帧控制（2字节） | 持续期（2字节） | 接收地址（6字节） | FCS（4字节，不显示） |



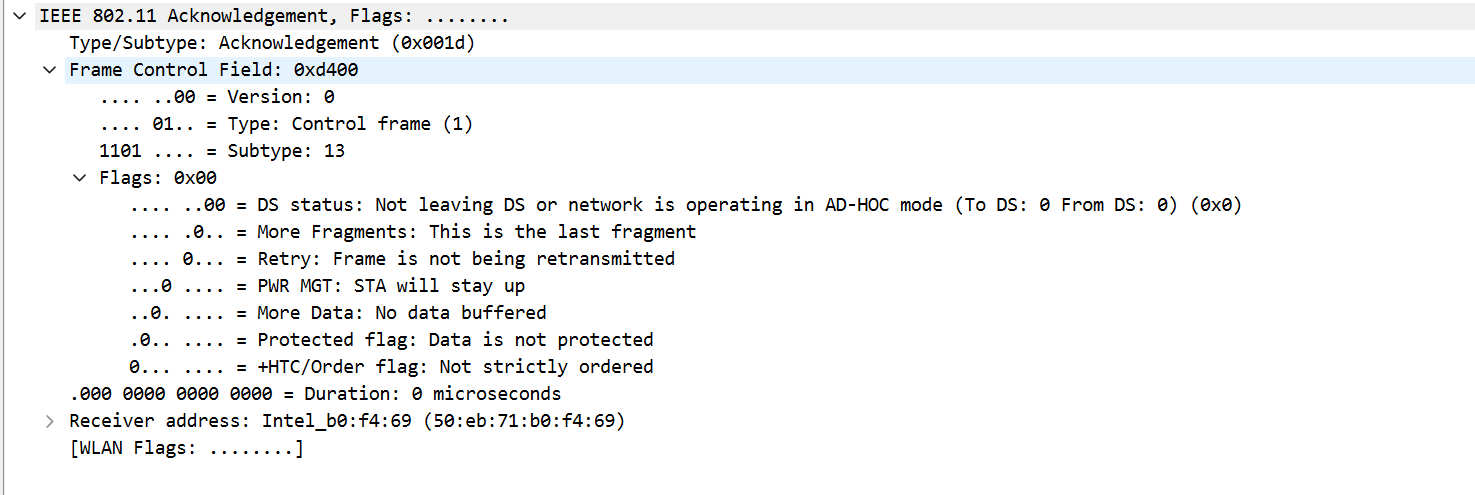
帧控制中的第一个字节是c4，即1100 0100，表示是CTS。持续期是00 00，接收地址是dc 71 96 f8 0a 84。

##### ③ACK



ACK帧和CTS帧格式相同。

帧控制中的第一个字节是d4，即1101 0100，表示是ACK。持续期是00 00，接收地址是50 eb 71 b0 f4 69。



#### （4）数据帧



类型：0x002c

帧控制的第一个字节为c8，即1100 1000，子类型1100，表示数据帧。

持续期：3c00

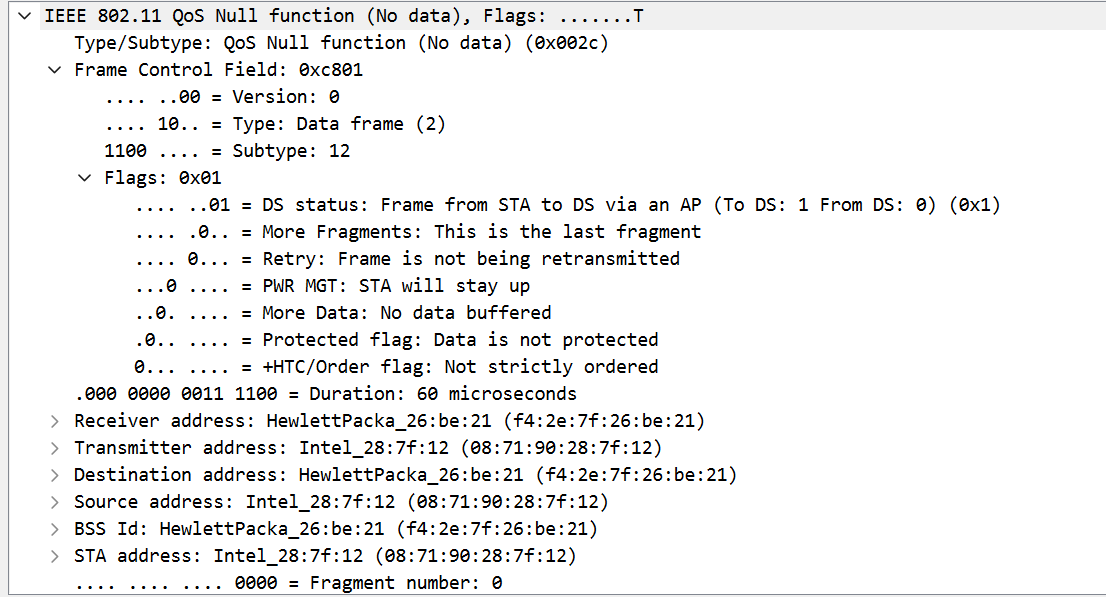
接收地址：f4 2e 7f 26 be 21

发送地址：08 71 90 28 7f 12

目的地址：f4 2e 7f 26 be 21

源地址：08 71 90 28 7f 12

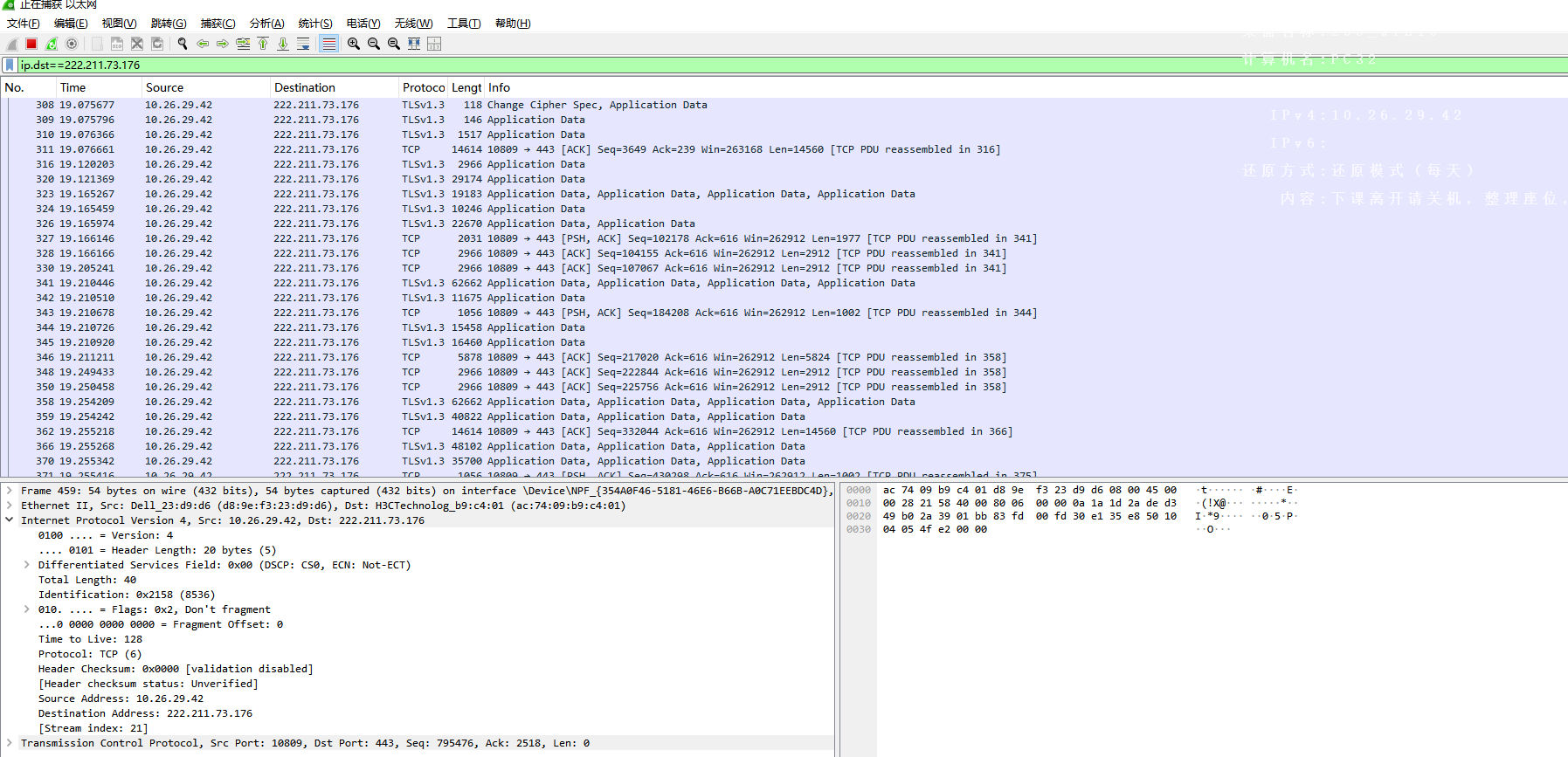
BSS Id:f4 2e 7f 26 be 21

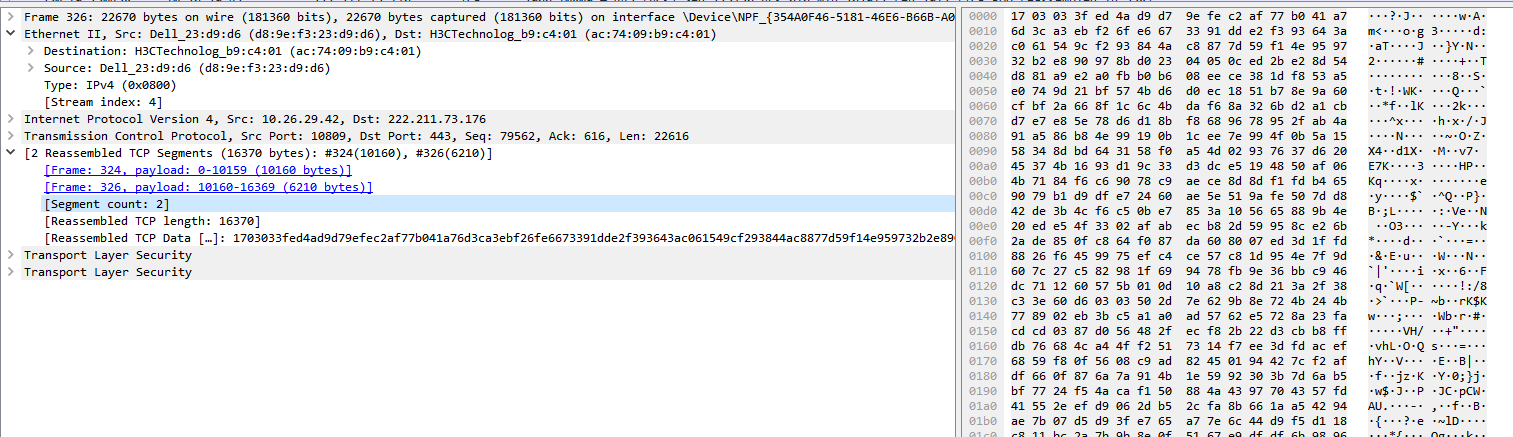


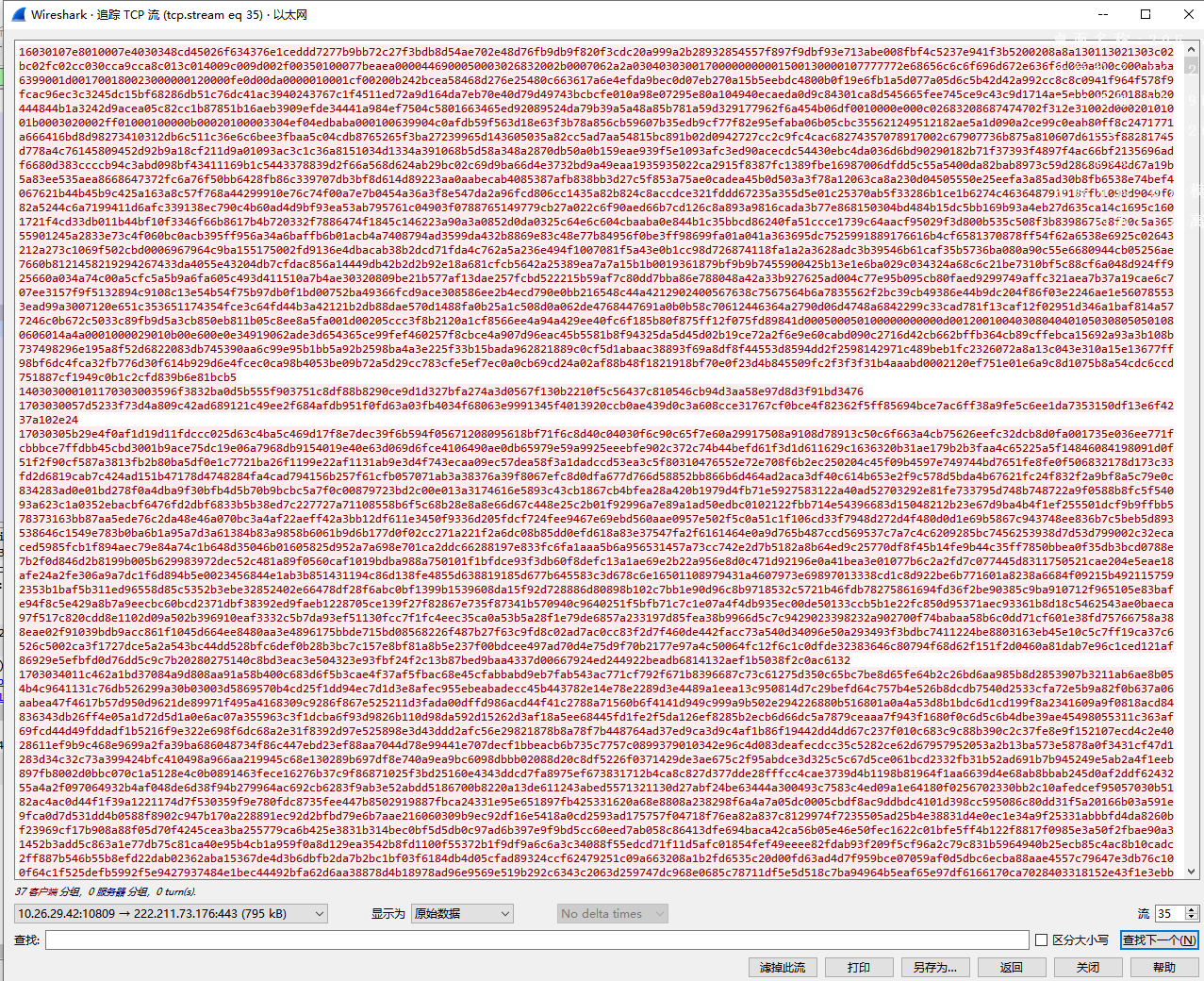
### 任务3.1：探索Wireshark更丰富的功能（选做）——数据追踪流

（1）结果

找到ip为222.211.73.176的服务器，捕获到发送流量如下







设置从10.26.29.42流向222.211.73.176且显示为原始数据

（2）分析说明

整个UDP流会在一个新的窗口中显示出来，这个窗口中的文件以两种颜色显示，其中红色部分是从源地址前往目的地址的流量，蓝色部分是从目的地址到源地址的流量。

# 四、实验小结

在本次实验中，通过使用Wireshark进行数据包抓取和分析，我深入探讨了MAC帧、IP数据报头、IP分片和ICMP协议等内容，这让我直观地认识到网络协议的重要性。Wireshark的抓包功能让我能够观察到数据帧在网络中的传输和流动，增强了我对数据传输过程的具体理解。

最初进行实验时，我对网络体系结构中的各个层次并没有太多印象，但在实验过程中，我查阅了相关书籍，详细了解了帧的格式，并将这些知识与Wireshark抓到的数据相对照。通过这种对比分析，我对数据帧的结构和格式有了更为深入的理解。