计算机网络实验： 使用Wireshark抓包工具进行网络层和链路层网络协议分析（IP部分）

原创

乔卿 已于 2022-01-20 10:58:42 修改 3002 收藏 7
分类专栏： 计算机网络 文章标签： 网络协议 wireshark tcp/ip
于 2022-01-19 22:12:43 首次发布

目录

实验名称：

网络层和链路层网络协议分析

实验介绍：

本实验通过执行traceroute程序，探究IP协议，实现对IP数据报发送和接收流程的追踪，研究分析 IP 首部各字段中的内容，了解 IP 分片的细节；分析了 Ethernet 协议以及 ARP协议。

实验目的：

（1） 利用 Wireshark 等工具对网络层和链路层网络协议进行分析。
（2） 研究 Ethernet 协议及 ARP 协议具体实现。

背景知识和准备：

熟悉 IP 数据报首部各字段的含义，掌握 Ethernet 协议以及 ARP 的工作原理。

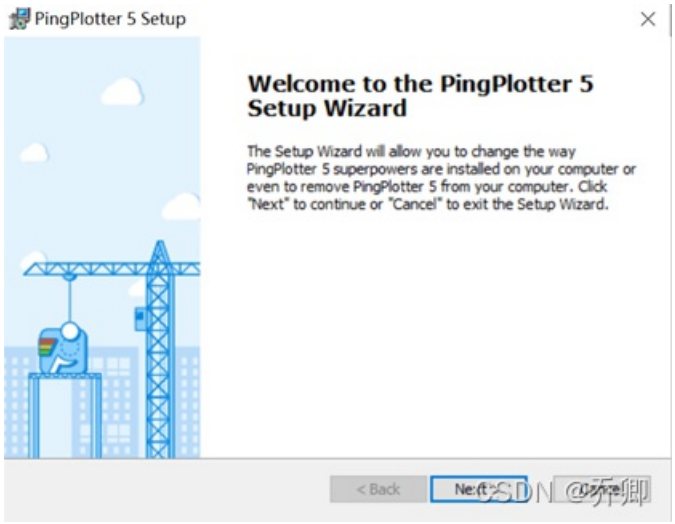
实验过程：

一、 IP协议分析

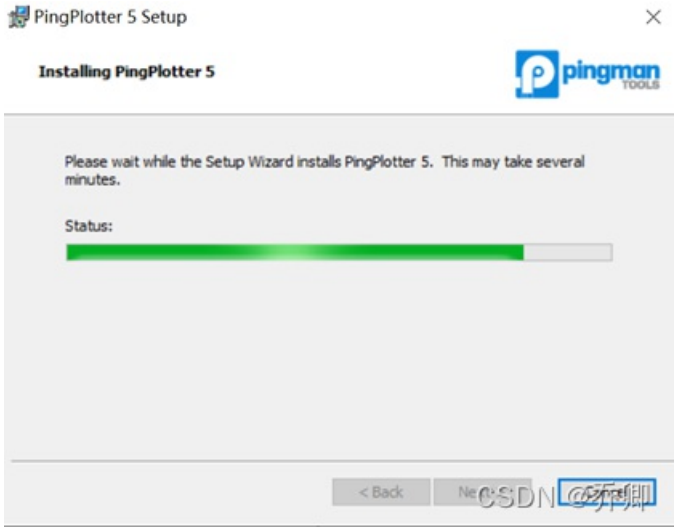
利用 Wireshark 工具抓取网络数据，分析 IP 数据报中各个字段的内容，分析 IP 数据报的分片。

1. 抓取数据包

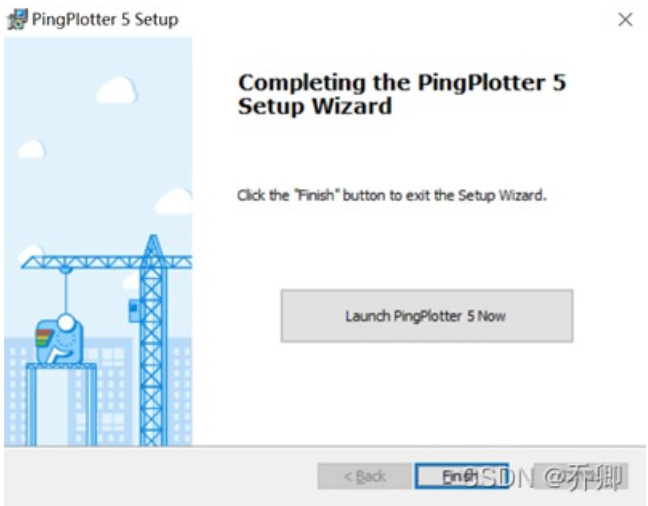
   a) 提前下载安装Ping Plotter。



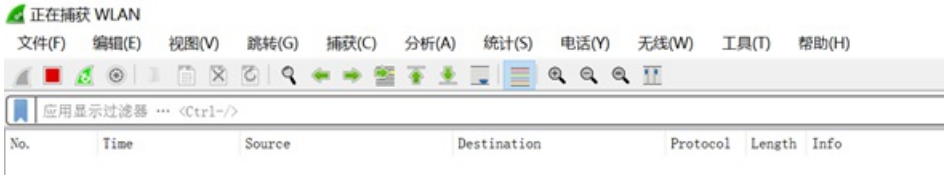
   点击Next，选择安装目录，进入安装过程。



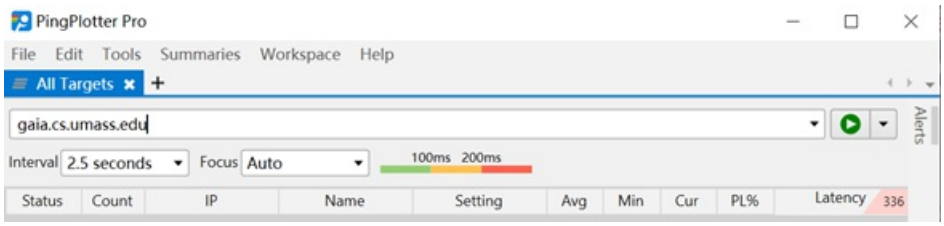
   点击Finish，完成安装。



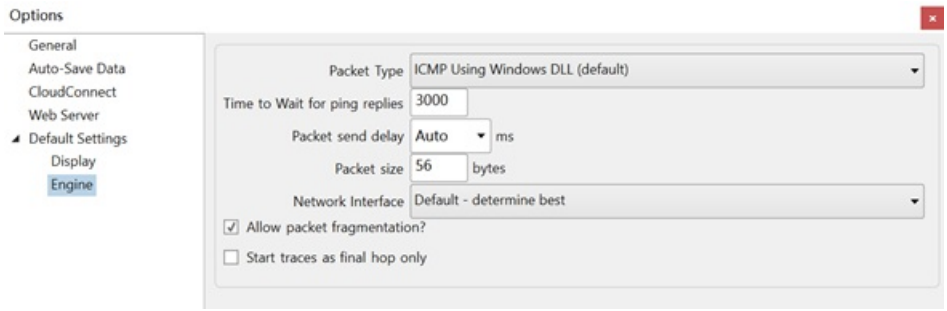
   b) 启动Wireshark开始抓包。具体操作步骤在实验二中已经介绍。在开始抓包前，退出其他软件，以免对结果的干扰。



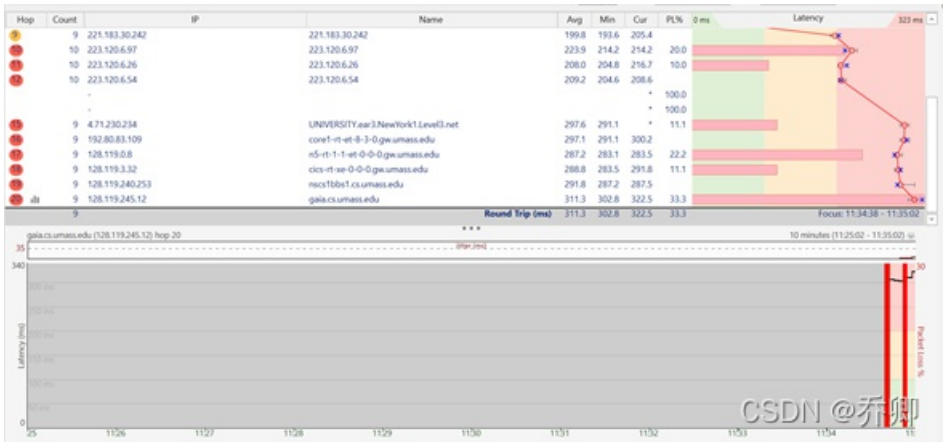
c) 启动 Ping Plotter，在“Address to Trace Window”框中输入目的地址“gaia.cs.umass.edu”。



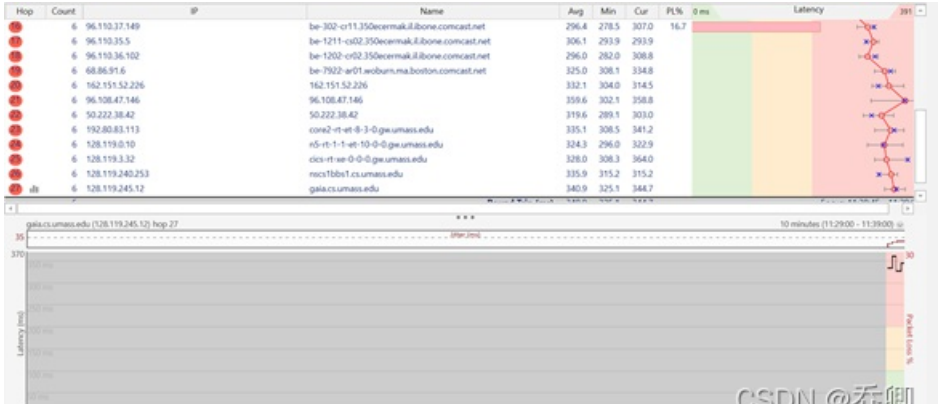
报文字段大小和等待时间可在菜单“Edit -> Options -> Default Settings -> Engine”中改变，初始值分别为56字节、3s。



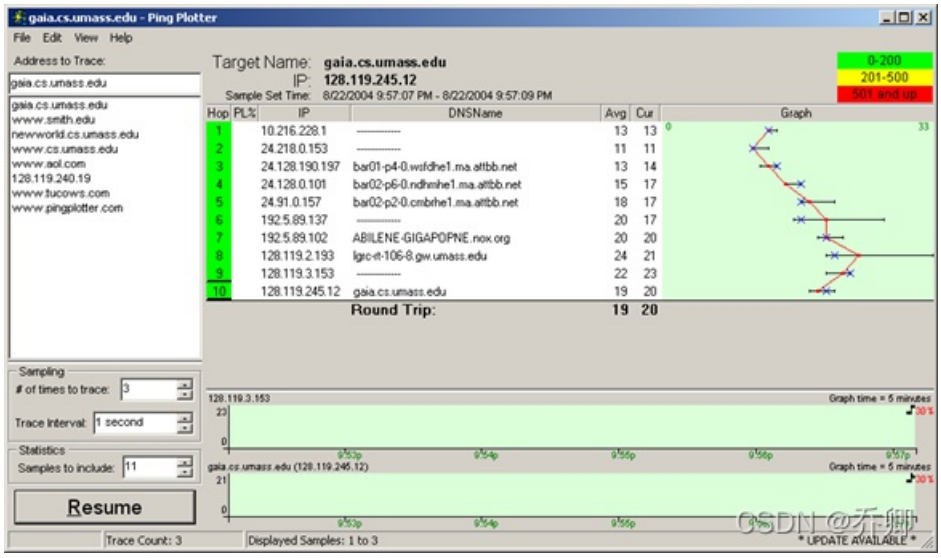
然后点击 Trace 按钮，弹出如下图所示的 Ping Plotter 窗口：



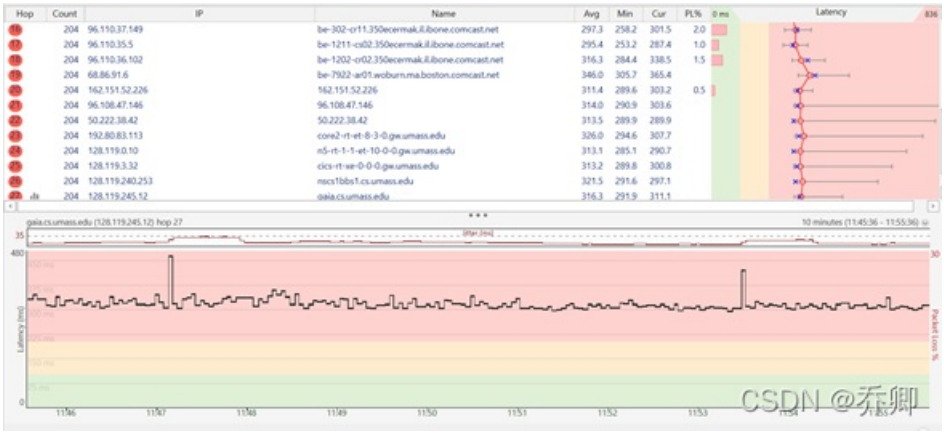
此时不应该出现这么多的丢包现象，担心会影响对IP报文的分析，因此尝试改用手机热点：



与实验指导书中给出的结果比对，二者一致。

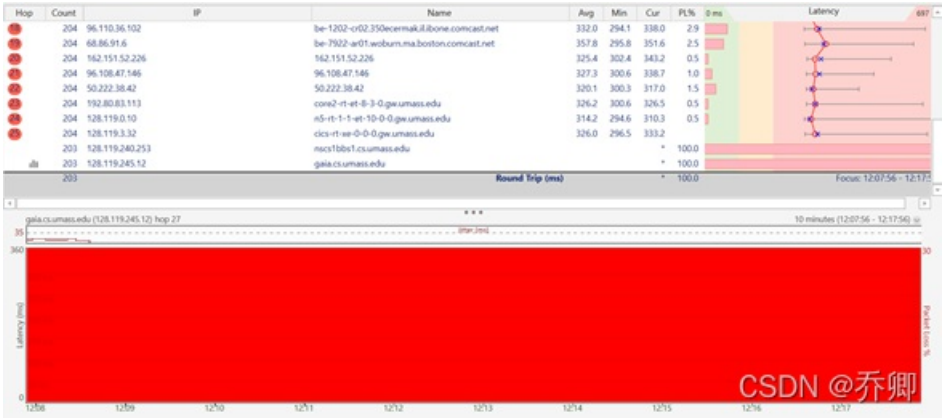


d) 等待一段时间。



此时，依次点击“Edit -> Options -> Default Settings -> Engine”，在Packet Size 框中输入 2000，再依次点击 OK -> Resume按钮。

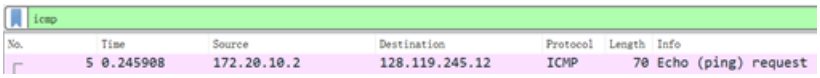
e) 以同样的方式将数据包大小设为3500字节。



f) 停止抓包。

2. 分析抓取的数据包并回答相关问题。
   在回答问题之前，首先回顾IP数据报的格式：

1. 选择你的电脑所发送的第一个ICMP请求消息，在包详细信息窗口扩展包的Internet协议部分。你的电脑的IP地址是多少？
   可以得知gaia.cs.umass.edu的IP地址为128.119.245.12，观察捕获到的ICMP请求消息：



   得知本机的IP地址为172.20.10.2。由于使用的是手机热点，所以和先前192.168开头的IP地址不同。在Wi-Fi属性中验证：

   二者结果一致，结论正确。

2. 在IP包头部，上层协议区域的值是多少？

□ IP包头长度（Header Length）：这个字段的作用是为了描述IP包头的长度，因为在IP包头中有变长的可选部分。IP包头最小长度为20字节，由于变长的可选部分最大长度可能会变成24字节。

□ 标记（Flags）：该字段第一位不使用。第二位是度DF位，DF位设为1时表明路由器不能对该上层数据包分段。如果一个上回层数据包无法在不分段的情况下进行转发，则路由器会丢弃该上层数据包并返回一个错误信息。第三位是MF位，当路由器对一个上层数据包分段，则路由器会在除了最答后一个分段的IP包的包头中将MF位设为1。

□ 协议（Protocol）：标识了上层所使用的协议，这是一个可变长的字段。该字段由源设备根据需要改写。

在IP数据包的头部上层协议字段的值，就是在IP头部表示的上层所使用的协议。使用本机所发送的第一个ICMP请求消息，或选择一个捕获的IP数据报文，该报文必须是本机(172.20.10.2)与目标地址(128.119.245.12)之间的IP报文，如下：

```
Internet Protocol Version 4, Src: 172.20.10.2, Dst: 128.119.245.12
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x20 (DSCP: CS1, ECN: Not-ECT)
    Total Length: 56
    Identification: 0xb8aa (47274)
  > Flags: 0x0000
    Fragment offset: 0
  > Time to live: 1
    Protocol: ICMP (1)
    Header checksum: 0xd560 [validation disabled]
    [Header checksum status: Unverified]
    Source: 172.20.10.2
    Destination: 128.119.245.12
Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x4089 [unverified] [in ICMP error packet]
    [Checksum Status: Unverified]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
    Sequence number (BE): 20054 (0x4e56)
    Sequence number (LE): 22094 (0x564e)
```


因此上层协议为ICMP(1)。

3. IP头部有多少字节？IP数据包的有效载荷是多少字节？解释你是怎样确定有效载荷的数量的？

   IP头部字节数为20，即Header Length。

   IP数据包的有效载荷=Total Length-Header Length。

```
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
```

   对于上图所示的报文，该值为36。

4. 这个IP数据包被分割了吗？解释你是怎样确定这个数据包是否被分割？

   这个IP数据包没有被分割。理由为标志、偏移均为0。

```
  Flags: 0x0000
      0... .... .... .... = Reserved bit: Not set
      .0.. .... .... .... = Don't fragment: Not set
      ..0. .... .... .... = More fragments: Not set
    Fragment offset: 0
```

5. 接下来单击列名按IP源地址排序数据包，选择你的电脑发送的第一个ICMP请求消息，扩展显示IP协议的数据。

```
      5 0.245908      172.20.10.2          128.119.245.12       ICMP
Internet Protocol Version 4, Src: 172.20.10.2, Dst: 128.119.245.12
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 56
    Identification: 0xb838 (47160)
  > Flags: 0x0000
      0... .... .... .... = Reserved bit: Not set
      .0.. .... .... .... = Don't fragment: Not set
      ..0. .... .... .... = More fragments: Not set
```

```
    Fragment offset: 0
    Time to live: 255
    Protocol: ICMP (1)
    Header checksum: 0xd7f1 [validation disabled]
    [Header checksum status: Unverified]
    Source: 172.20.10.2
    Destination: 128.119.245.12
v Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0xe858 [correct]
    [Checksum Status: Good]
    Identifier (BE): 1 (0x0001)
    Identifier (LE): 256 (0x0100)
```


6. 在包捕获列表窗口，你能看到在第一个ICMP下的所有并发的ICMP消息吗？
   可以看到在第一个ICMP下的所有并发的ICMP消息：

```
  5 0.245908    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19940/58445, ttl=255
  8 0.292917    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19941/58701, ttl=1 (
 11 0.339270    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19942/58957, ttl=2 (
 12 0.387221    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19943/59213, ttl=3 (
 15 0.433632    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19944/59469, ttl=4 (
 17 0.480414    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19945/59725, ttl=5 (
 20 0.526617    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19946/59981, ttl=6 (
 25 0.574681    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19947/60237, ttl=7 (
 27 0.620941    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19948/60493, ttl=8 (
 30 0.668413    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19949/60749, ttl=9 (
 33 0.714498    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19950/61005, ttl=10
 34 0.762141    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19951/61261, ttl=11
 35 0.808880    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19952/61517, ttl=12
 41 0.855636    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19953/61773, ttl=13
 43 0.902142    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19954/62029, ttl=14
 45 0.948836    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19955/62285, ttl=15
 47 0.995155    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19956/62541, ttl=16
 51 1.042959    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19957/62797, ttl=17
 54 1.089802    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19958/63053, ttl=18
 60 1.136773    172.20.10.2     128.119.245.12    ICMP     70 Echo (ping) request  id=0x0001, seq=19959/63309, ttl=19
```

观察到这些ICMP消息的ID相同，seq各不相同，由于它们的flag均为0，因此不是IP分片，故为并发消息。

7. 往同一IP的数据包哪些字段在改变，而且必须改变？为什么？哪些字段是保持不变的，而且必须保持不变？

```
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
> Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 540
    Identification: 0x0c43 (3139)
> Flags: 0x0172
    Fragment offset: 2960
    Time to live: 5
    Protocol: ICMP (1)
    Header checksum: 0x7a92 [validation disabled]
    [Header checksum status: Unverified]
    Source: 172.20.10.2
    Destination: 128.119.245.12
> [3 IPv4 Fragments (3480 bytes): #61549(1480), #61550(1480), #61551(520)]
```


必须改变的字段：

- identification（标识符），用于区分每个数据包
- header checksum（头部检查和），会随着首部数据的改变而改变

必须保持不变的字段：

- version（版本号）
- header length（头部长度）
- differentiated services field（区分服务）
- protocol（协议）
- source（源地址）
- destination（目的地址）

8. 描述一下在IP数据包的Identification字段的值是什么样的？
   每一个Identification都是一个唯一的标识符，如0x0c42，且相邻数据包的Identification以步长为1递增。

IP分片

首先回顾一下IP分片的知识：

a) 单击Time栏，按时间先后顺序排列数据报，找到在Ping Plotter中将分组大小改为2000字节后的第一个ICMP Echo Request报文，即下图中的报文：

```
    19719 1040.285274   128.119.245.12    172.20.10.2       ICMP      70 Echo (ping) reply
    19721 1041.616536   172.20.10.2       128.119.245.12    ICMP     534 Echo (ping) request
    19723 1041.666883   172.20.10.2       128.119.245.12    ICMP     534 Echo (ping) request
> Frame 19721: 534 bytes on wire (4272 bits), 534 bytes captured (4272 bits) on interface \Device\NPF
> Ethernet II, Src: IntelCor_4f:af:ab (88:b1:11:4f:af:ab), Dst: be:9f:ef:ca:83:64 (be:9f:ef:ca:83:64)
v Internet Protocol Version 4, Src: 172.20.10.2, Dst: 128.119.245.12
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 520
    Identification: 0xde32 (56882)
  > Flags: 0x00b9
    Fragment offset: 1480
    Time to live: 255
    Protocol: ICMP (1)
    Header checksum: 0xaf6e [validation disabled]
    [Header checksum status: Unverified]
    Source: 172.20.10.2
    Destination: 128.119.245.12
  > [2 IPv4 Fragments (1980 bytes): #19720(1480), #19721(500)]
```


b) 分析IP 数据报分片信息；

i. 首先， flags不为0；

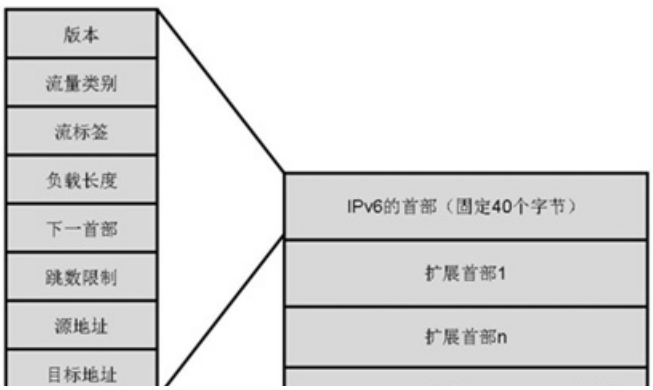
ii. 其次，观察到fragment offset不为0，而是1480.由于我们是按照time先后顺序进行排序，因此排在前面的报文可能不是分片中前面的报文。

附：几个比较好奇的问题的总结

a) 关于IPv4与IPv6

```
v Ethernet II, Src: IntelCor_4f:af:ab (88:b1:11:4f:af:ab), Dst: be:9f:ef:ca:83:64 (be:9f:ef:ca:83:64)
  > Destination: be:9f:ef:ca:83:64 (be:9f:ef:ca:83:64)
  > Source: IntelCor_4f:af:ab (88:b1:11:4f:af:ab)
    Type: IPv4 (0x0800)
```

观察到，捕获的报文使用的协议为IPv4（网际协议版本4）。在课堂上，我们简单地了解了下IPv6协议，知道该协议长度为128位，可以容纳更多的地址。IPv6的报文结构可以表示为：



具体如下：
□ 版本：标识IP报文的版本，其作用与IPv4的版本一样。但是值为IPv66。
□ 流量类别：指示IPv6数据流通信类别或优先级。功能类似于IPv4的服务类型（TOS）字段。用于服务质量（QOS）功能，长度为8位。
□ 流标签：IPv6新增字段，长度为20位。标记需要IPv6路由器特殊处理的数据流。该字段用于某些对连接的服务质量有特殊要求的通信，诸如音频或视频等实时数据传输。事实上所谓的“流”事实上是一系列具备相同特性的数据集合，通常是实时数据，比如：这些数据具备相同的目标IP地址、源IP地址、目标端口等，那么它们将具备一个相同的标签值。规划这个字段的意义目标在于对实时数据进行低延迟交付应用。
□ 载荷长度：16位负载长度表示IPv6报文中负载的长度，而不是整个IPv6数据报文的长度。没有包括整个IPv6报文中的主首部的长度。
□ 下一个首部：长度为8位。代替IPv4中的协议字段，因为IPv6提出了扩展首部的思想，该字段就是指示下一个扩展首部的标识，也就是IPv6数据报的下一个首部。如果一个IPv6报文没有下一个首部的引入，那么该字段就和IPv4中的协议字段的作用相同。指示使用的上层协议类型。
□ 跳数限制：该字段8位长度。替代IPv4的TTL（生命期）字段，指示在路由器之间的转发次数来限定IPv6报文的生命周期。每经过路由器一次转发，该字段减1，减到0时就把这个包丢弃。
□ 源地址：与IPv4中的源IP地址作用一样，只是用128比特进行表示。
□ 目标地址：与IPv4中的目标IP地址作用一样，只是用128比特进行表示。

b) Version：4代表什么？

```
Internet Protocol Version 4, Src: 172.20.10.2, Dst: 128.119.245.12
  0100 .... = Version: 4
```

这里的Version：4指的就是IPv4。

c) ICMP是什么协议？它就是IP协议吗？

```
Protocol: ICMP (1)
```

互联网在数据传输时，必定会发生差错，有些差错是不可知的，但有些却是可以知道的。TCP/IP协议包含一个专门的用于发送差错报文的协议，这一协议即为ICMP（Internet控制报文协议）。它是TCP/IP协议族的一个IP层子协议，用于在IP主机、路由器之间传递控制消息。控制消息是指网络通不通、主机是否可达、路由是否可用等网络本身的消息。
一个标准的IP协议中，ICMP是不可少的。ICMP的主要功能包括：确认IP包是否成功送达目标地址，通知在发送过程当中IP包被废弃的具体原因，改善网络设置等。同时ICMP的这种通知消息会使用IP进行发送，收到ICMP包的主机会分解ICMP的首部和数据与以后得知具体发生的原因。
d) ttl（time to live）具体是什么？

```
Time to live: 255
```

生存时间，表明是数据报在网络中的寿命。由发出数据报的源点设置这个字段。其目的是防止无法交付的数据报无限制地在因特网中兜圈子，因而白白消耗网络资源。最初的设计是以秒作为TTL的单位。每经过一个路由器时，就把TTL减去数据报在路由器消耗掉的一段时间。若数据报在路由器消耗的时间小于1秒，就把TTL值减1。当TTL值为0时，就丢弃这个数据报。后来把TTL字段的功能改为“跳数限制”（但名称不变）。路由器在转发数据报之前就把TTL值减1。若TTL值降低到零，就丢弃这个数据报，不再转发。因此，如今TTL的单位不再是秒，而是跳数。TTL的意义是指明数据报在网络中至多可经过多少个路由器。

二、 Ethernet & ARP 协议分析

这部分我另写了一篇博客详细介绍。地址为：
https://blog.csdn.net/qq_41112170/article/details/122591096

如果这篇文章对你有帮助，请给博主点个赞鼓励一下吧！