《网络与通信》课程实验报告

实验三：数据包结构分析

姓名	严昕宇	院系	计算机学院	学号	20121802
任课教师	曹晨红		指导教师	曹晨红	
实验地点	计 708		实验时间	2022 年 10 月 12 日	
实验课表现	出勤、表现得分(10)		实验报告得分(40)		实验总分	
	操作结果得分(50)					

实验目的：

1. 了解 Sniffer 的工作原理，掌握 Sniffer 抓包、记录和分析数据包的方法；
2. 在这个实验中，你将使用抓包软件捕获数据包，并通过数据包分析每一层协议。

实验内容：

使用抓包软件捕获数据包，并通过数据包分析每一层协议。

实验要求：（学生对预习要求的回答）（10 分）　　得分：

- 常用的抓包工具

  Wireshark、Charles、Fiddler、Microsoft Network Monitor、NetXray、Sniffer Pro

实验过程中遇到的问题如何解决的？（10 分）　　得分：

问题 1：怎样选择 Sniffer 软件？

答：Sniffer 软件有很多，要选择的话应该考虑易于安装部署，易于学习使用，同时也易于交流，也就是使用人群较多的软件。而 Wireshark(Ethereal)是全世界最广泛的网络封包分析软件之一，它拥有强大的过滤器引擎，用户可以使用过滤器筛选出有用的数据包，排除无关信息的干扰。Wireshark 现在也在持续更新，且有官方中文，支持各种系统。

问题 2：出现［TCP Previous segment not captured］

答：在 TCP 传输过程中，同一台主机发出的数据段应该是连续的，即后一个包的 Seq 号等于前一个包的 Seq Len(三次握手和四次挥手是例外)。如果 Wireshark 发现后一个包的 Seq 号大于前一个包的 Seq Len，就知道中间缺失了一段数据。中间缺失的数据有的时候是由于乱序导致的，在后面的包中还可以找到。

问题 3：出现 [TCP Dup ACK]

答：当乱序或者丢包发生时，接收方会收到一些 Seq 号比期望值大的包。它每收到一个这种包就会 Ack 一次期望的 Seq 值，以此方式来提醒发送方，于是就产生了一些重复的 Ack。同时，这些重复的 ACK 中，也会更新 SLE 和 SRE 的值，因为尽管 ACK 的值不变，即期望的 seq 没有收到，但可能又会新收到后面的乱序的包。当收到一个出问题的分片，Tcp 立即产生一个应答。这个相同的 ack 不会延迟。这个相同应答的意图是让对端知道一个分片被收到的时候出现问题，且告诉它希望得到的序列号。

本次实验的体会（结论）（10 分）	得分：

这次实验让我清晰了解数据包在网络中传输的格式和方式，学会了怎么样捕获数据包和分析每一层协议，比如 TCP 采用全双工模式，在连接建立后和连接中止前进行数据传输，数据传输是单向的，从发送端传输给接受端。TCP 通过序列号能够保证数据被接受端接受。TCP 建立连接是通过三次握手的方式来建立连接的。

通过抓包实验，我对各种数据包的格式更加熟悉，对各个层次的作用和相应的协议有一定的了解，如传输控制协议，为数据提供可靠的端到端传输，处理数据的顺序和错误恢复，保证数据能够到达其应到达的地方。

思考题：（10 分）

思考题 1：（4 分）	得分：

写出捕获的数据包格式。

Frame 389: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface \Device\NPF_{8BD94318-04BB-416C-9EEB-03DC30B22EEB}, id 0

- Section number: 1
- Interface id: 0 (\Device\NPF_{8BD94318-04BB-416C-9EEB-03DC30B22EEB})
- Encapsulation type: Ethernet (1)
- Arrival Time: Oct 12, 2022 08:01:40.899362000 中国标准时间
- [Time shift for this packet: 0.000000000 seconds]
- Epoch Time: 1665532900.899362000 seconds
- [Time delta from previous captured frame: 0.011101000 seconds]
- [Time delta from previous displayed frame: 0.000000000 seconds]
- [Time since reference or first frame: 10.807652000 seconds]
- Frame Number: 389
- Frame Length: 74 bytes (592 bits)
- Capture Length: 74 bytes (592 bits)
- [Frame is marked: False]
- [Frame is ignored: False]
- [Protocols in frame: eth:ethertype:ip:icmp:data]
- [Coloring Rule Name: ICMP]
- [Coloring Rule String: icmp || icmpv6]

Ethernet II, Src: IntelCor_18:20:27 (8c:c6:81:18:20:27), Dst: RuijieNe_7d:49:25 (14:14:4b:7d:49:25)

- Destination: RuijieNe_7d:49:25 (14:14:4b:7d:49:25)
- Source: IntelCor_18:20:27 (8c:c6:81:18:20:27)
- Type: IPv4 (0x0800)

Internet Protocol Version 4, Src: 10.88.138.143, Dst: 110.242.68.66

- 0100 .... = Version: 4
- .... 0101 = Header Length: 20 bytes (5)
- Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
- Total Length: 60
- Identification: 0xd9a9 (55721)
- 000. .... = Flags: 0x0

...0 0000 0000 0000 = Fragment Offset: 0
Time to Live: 64
Protocol: ICMP (1)
Header Checksum: 0x0000 [validation disabled]
[Header checksum status: Unverified]
Source Address: 10.88.138.143
Destination Address: 110.242.68.66

Internet Control Message Protocol
Type: 8 (Echo (ping) request)
Code: 0
Checksum: 0x4d56 [correct]
[Checksum Status: Good]
Identifier (BE): 1 (0x0001)
Identifier (LE): 256 (0x0100)
Sequence Number (BE): 5 (0x0005)
Sequence Number (LE): 1280 (0x0500)
[Response frame: 390]
Data (32 bytes)

思考题2：（6分）	得分：

写出实验过程并分析实验结果。

Wireshark 是非常流行的网络封包分析软件，可以截取各种网络数据包，并显示数据包详细信息。常用于开发测试过程各种问题定位。此次实验中主要包括：

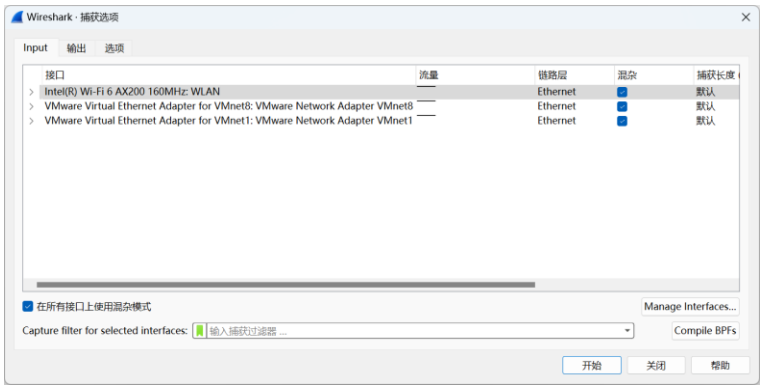
- WireShark 简单抓包。包括抓包以及简单查看并分析数据包内容；
- Wireshark 过滤器使用。通过过滤器筛选出想要分析的内容，包括按照协议过滤、端口和主机名过滤、数据包内容过滤。

- **Wireshark 抓包示例**

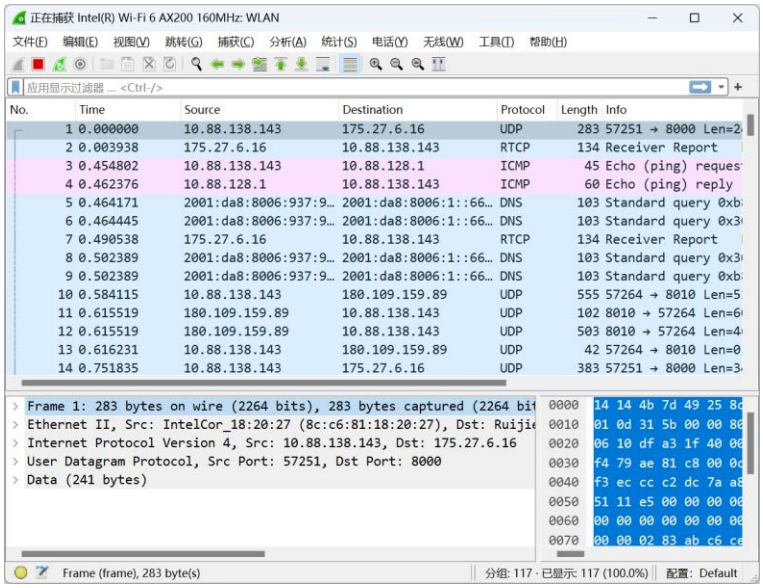
先介绍一个使用 Wireshark 工具抓取 ping 命令操作的示例，让读者可以先上手操作感受一下抓包的具体过程。

① 打开 Wireshark 网络分析器 4.0.0，主界面如下：

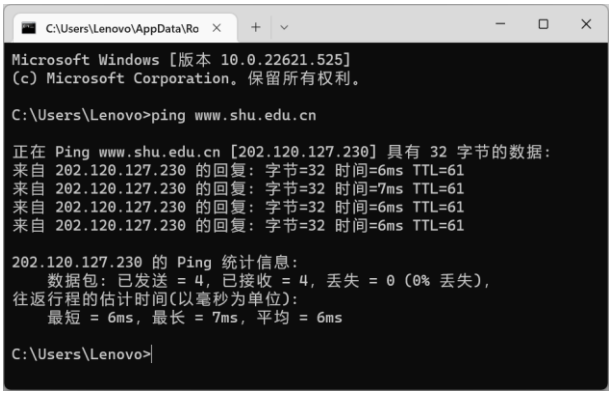
② 选择菜单栏上捕获(Capture)→选项(Option)，勾选 WLAN 网卡，并使用混杂模式。点击 Start，启动抓包。



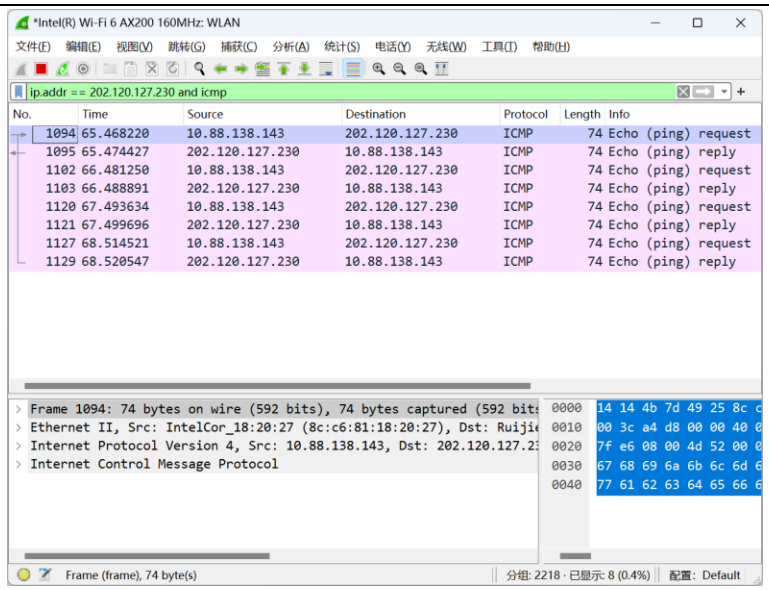
③ Wireshark 启动后，其处于抓包状态中。



④ 执行需要抓包的操作，此处以 ping www.shu.edu.cn 为例。



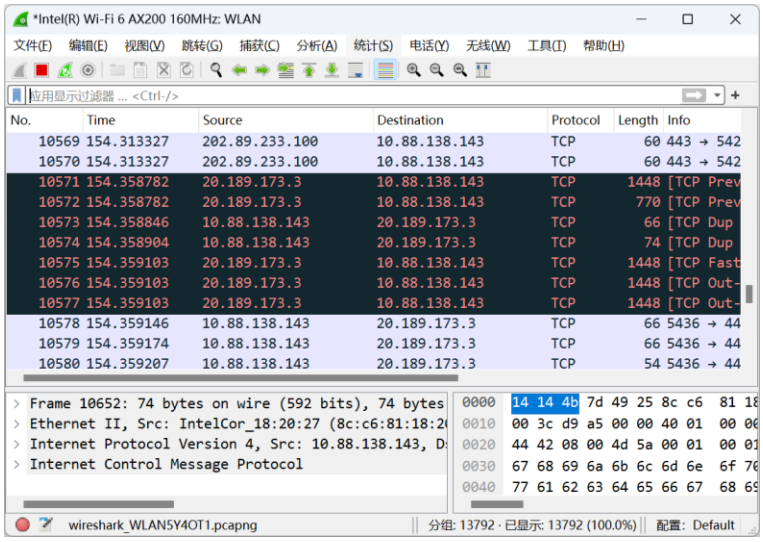
⑤ 操作完成后，相关数据包即可被抓获。为避免其他无用的数据包影响分析，可通过在过滤栏设置过滤条件进行数据包列表过滤，结果如下：



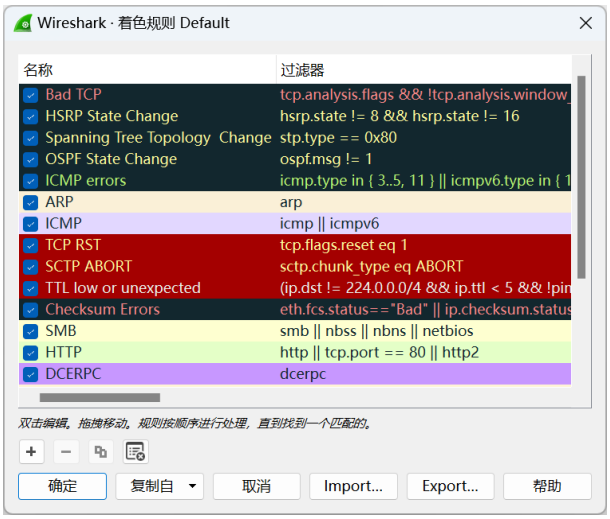
说明：ip.addr == 202.120.127.230 and icmp，表示只显示 ICPM 协议且源主机 IP 或者目的主机 IP 为 202.120.127.230 的数据包。

⑥ 使用 Wireshark 抓包完成。而在本次实验中，关于 Wireshark 的过滤条件和如何查看数据包中的详细内容，将在本实验报告的后续部分展开分析。

- **Wireshark 抓包界面**



说明：数据包列表区中不同的协议使用了不同的颜色区分。协议颜色标识定位在菜单栏视图(View)→着色规则(Coloring Rules)。如下所示：



- **WireShark 界面功能划分**

① Display Filter(显示过滤器)。用于设置过滤条件进行数据包列表过滤。菜单路径：分析(Analyze)→显示过滤器(Display Filters)。

② Packet List Pane(数据包列表)。显示捕获到的数据包，每个数据包包含编号，时间戳，源地址，目标地址，协议，长度，以及数据包信息。不同协议的数据包使用了不同的颜色区分显示。

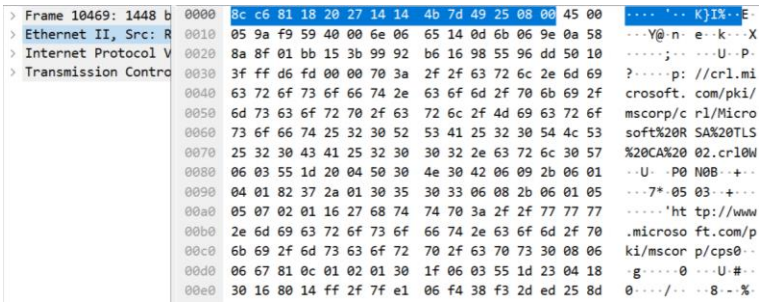
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	222.94.109.108	10.88.138.143	OICQ	129	OICQ Prot
2	0.234190	222.94.109.108	10.88.138.143	OICQ	129	OICQ Prot
3	1.775229	222.94.109.108	10.88.138.143	OICQ	129	OICQ Prot
4	2.515556	222.94.109.108	10.88.138.143	OICQ	129	OICQ Prot
5	5.122573	fe80::1614:4bff:fe7…	ff02::1	ICMPv6	118	Router Ad
6	5.856704	2001:da8:8006:941:8…	2603:1047:1:168::83	TCP	75	5292 → 44
7	5.872686	10.88.138.143	202.89.233.100	TCP	55	5293 → 44
8	5.904122	202.89.233.100	10.88.138.143	TCP	66	443 → 529
9	5.966537	2603:1047:1:168::83	2001:da8:8006:941:8…	TCP	86	443 → 529
10	6.054717	fe80::1614:4bff:fe7…	ff02::1	ICMPv6	118	Router Ad
11	6.328649	2001:da8:8006:941:8…	2620:1ec:c11::239	TCP	75	5294 → 44

③ Packet Details Pane(数据包详细信息)，在数据包列表中选择指定数据包，在数据包详细信息中会显示数据包的所有详细信息内容。数据包详细信息面板是最重要的，用来查看协议中的每一个字段。各行信息分别为：

- Frame：物理层的数据帧概况
- Ethernet II：数据链路层以太网帧头部信息
- Internet Protocol Version 4：互联网层 IP 包头部信息
- Transmission Control Protocol：传输层 T 的数据段头部信息，此处是 TCP
- Hypertext Transfer Protocol：应用层的信息，此处是 HTTP 协议

```
> Frame 10469: 1448 bytes on wire (11584 bits), 1448 bytes captured (11584 bits) on interfac
> Ethernet II, Src: RuijieNe_7d:49:25 (14:14:4b:7d:49:25), Dst: IntelCor_18:20:27 (8c:c6:81:
> Internet Protocol Version 4, Src: 13.107.6.158, Dst: 10.88.138.143
v Transmission Control Protocol, Src Port: 443, Dst Port: 5435, Seq: 2789, Ack: 292, Len: 13
    Source Port: 443
    Destination Port: 5435
    [Stream index: 142]
    [Conversation completeness: Complete, WITH_DATA (47)]
    [TCP Segment Len: 1394]
    Sequence Number: 2789    (relative sequence number)
    Sequence Number (raw): 2576528918
    [Next Sequence Number: 4183    (relative sequence number)]
    Acknowledgment Number: 292    (relative ack number)
    Acknowledgment number (raw): 2555746013
    0101 .... = Header Length: 20 bytes (5)
  > Flags: 0x010 (ACK)
```

从下图可以看到 Wireshark 捕获到的 TCP 包中的每个字段。

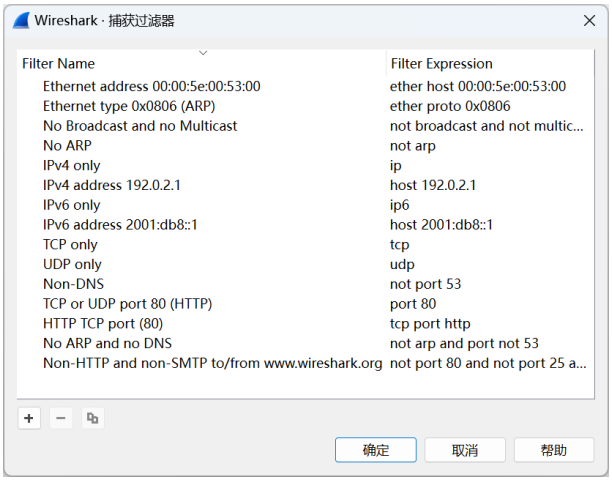


- **Wireshark 过滤器设置**

使用 Wireshark 时，将会得到大量的冗余数据包列表，以至于很难找到期望抓取的数据包部分。Wireshark 工具中自带了两种类型的过滤器，使用这两种过滤器会帮助我们在大量的数据中迅速找到我们需要的信息。

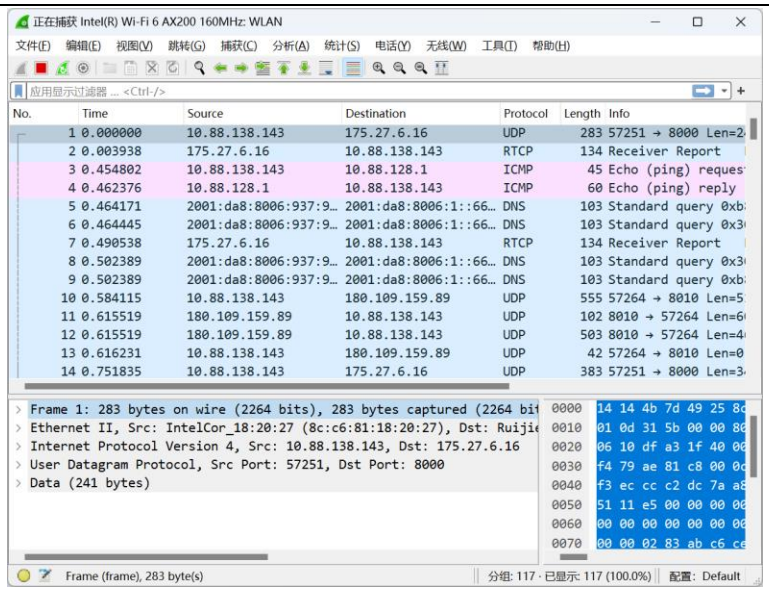
① 抓包过滤器

捕获过滤器的菜单栏路径为捕获(Capture)→捕获过滤器(Capture Filters)。用于在抓取数据包前设置。

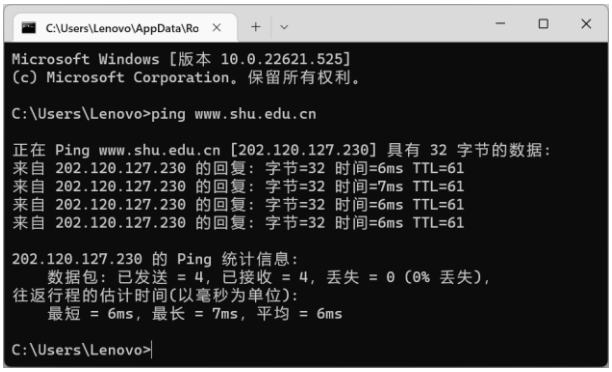


② 显示过滤器

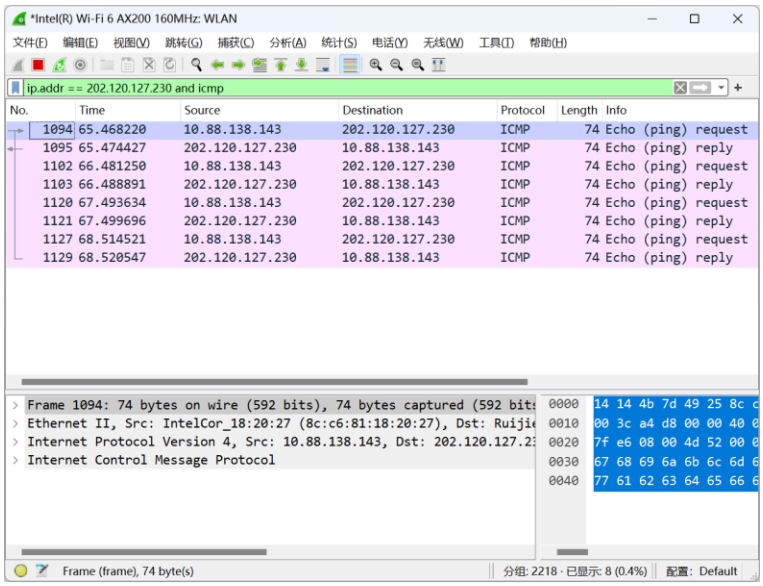
显示过滤器是用于在抓取数据包后设置过滤条件进行过滤数据包。通常是在抓取数据包时设置条件相对宽泛，抓取的数据包内容较多时使用显示过滤器设置条件顾虑以方便分析。同样上述场景，在捕获时未设置捕获规则直接通过网卡进行抓取所有数据包，如下所示：



执行 ping www.shu.edu.cn 获取的数据包列表如下：



观察上述获取的数据包列表，其中含有大量的无效数据。此时可以通过设置显示器过滤条件 ip.addr == 202.120.127.230 and icmp 并进行提取分析信息。



上述介绍了抓包过滤器和显示过滤器的基本使用方法。在组网不复杂或者流量不大情况下，使用显示器过滤器进行抓包后处理就可以满足日常使用。下面介绍一下两者间的语法以及它们的区别。

- **Wireshark 过滤器表达式的规则**

① 抓包过滤器语法和实例

抓包过滤器类型 Type(host、net、port)、方向 Dir(src、dst)、协议 Proto(ether、ip、tcp、udp、http、icmp、ftp 等)、逻辑运算符(&&、||、！)。

(1) 协议过滤

其操作比较简单，直接在抓包过滤框中直接输入协议名即可。

- TCP，只显示 TCP 协议的数据包列表
- HTTP，只查看 HTTP 协议的数据包列表
- ICMP，只显示 ICMP 协议的数据包列表

(2) IP 过滤

- host 192.168.1.104
- src host 192.168.1.104
- dst host 192.168.1.104

(3) 端口过滤

- port 80
- src port 80
- dst port 80

(4) 逻辑运算符&& 与、|| 或、！非

- src host 192.168.1.104 && dst port 80，表示抓取主机地址为 192.168.1.80、目的端口为 80 的数据包
- host 192.168.1.104 || host 192.168.1.102，表示主机为 192.168.1.104 或者 192.168.1.102 的数据包
- ！broadcast 不抓取广播数据包

② 显示过滤器语法和实例

(1) 比较操作符

- 比较操作符有==、！=、>、<、>=、<=

(2) 协议过滤

比较简单，直接在 Filter 框中直接输入协议名即可。注意：协议名称需要输入小写。

- tcp，只显示 TCP 协议的数据包列表
- http，只查看 HTTP 协议的数据包列表
- icmp，只显示 ICMP 协议的数据包列表

(3) ip 过滤

- ip.src ==192.168.1.114，表示显示源地址为 192.168.1.114 的数据包列表
- ip.dst==192.168.1.114，表示显示目标地址为 192.168.1.114 的数据包列表
- ip.addr == 192.168.1.114，表示显示源 IP 地址或目标 IP 地址为 192.168.1.114 的数据包列表

(4) 端口过滤

- tcp.port ==80，显示源主机或者目的主机端口为 80 的数据包列表
- tcp.srcport == 80，只显示 TCP 协议的源主机端口为 80 的数据包列
- tcp.dstport == 80，只显示 TCP 协议的目的主机端口为 80 的数据包列表

(5) Http 模式过滤

- http.request.method=="GET"，只显示 HTTP GET 方法的

(6) 逻辑运算符为 and / or / not

- 过滤多个条件组合时，使用 and/or。比如获取 IP 地址为 192.168.1.104 的 ICMP 数据包表达式为 ip.addr == 192.168.1.104 and icmp

(7) 按照数据包内容过滤。由于实验报告篇幅内容，此处省略。

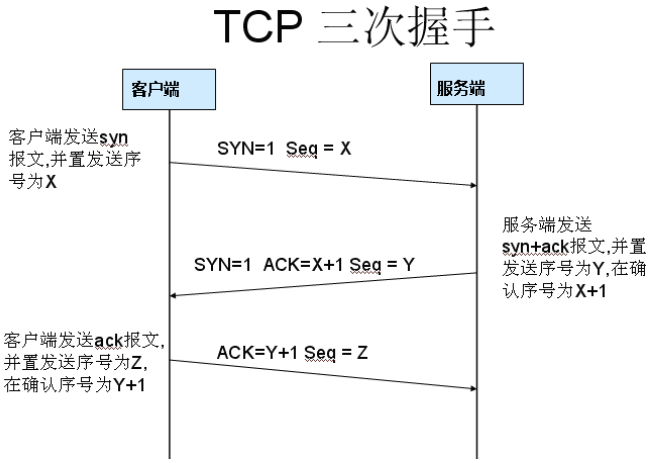
- **Wireshark 抓包分析 TCP 三次握手**

(1) TCP 三次握手连接建立过程

Step1：客户端发送一个 SYN=1，ACK=0 标志的数据包给服务端，请求进行连接，这是第一次握手；

Step2：服务端收到请求并且允许连接的话，就会发送一个 SYN=1，ACK=1 标志的数据包给发送端，告诉它，可以通讯了，并且让客户端发送一个确认数据包，这是第二次握手；

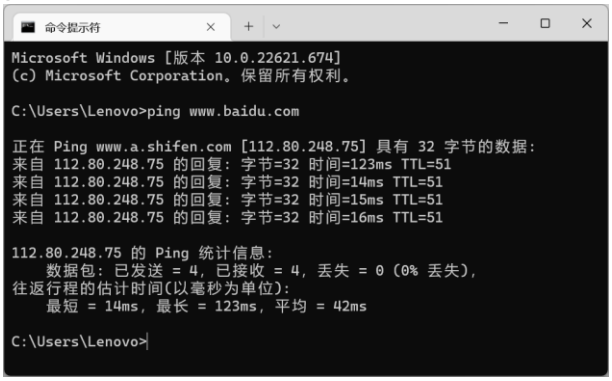
Step3：服务端发送一个 SYN=0，ACK=1 的数据包给客户端端，告诉它连接已被确认，这就是第三次握手。TCP 连接建立，开始通讯。



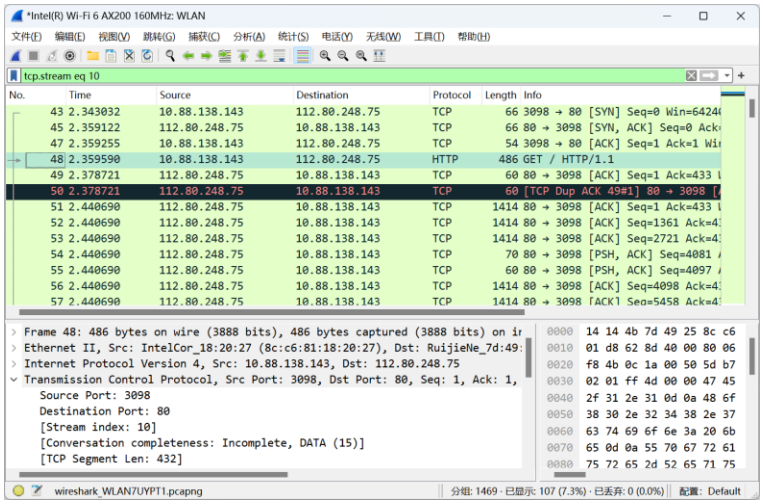
- **Wireshark 抓包获取访问指定服务端数据包**

Step1：启动 Wireshark 抓包，打开浏览器输入 www.baidu.com。

Step2：使用 ping www.baidu.com 获取 IP。



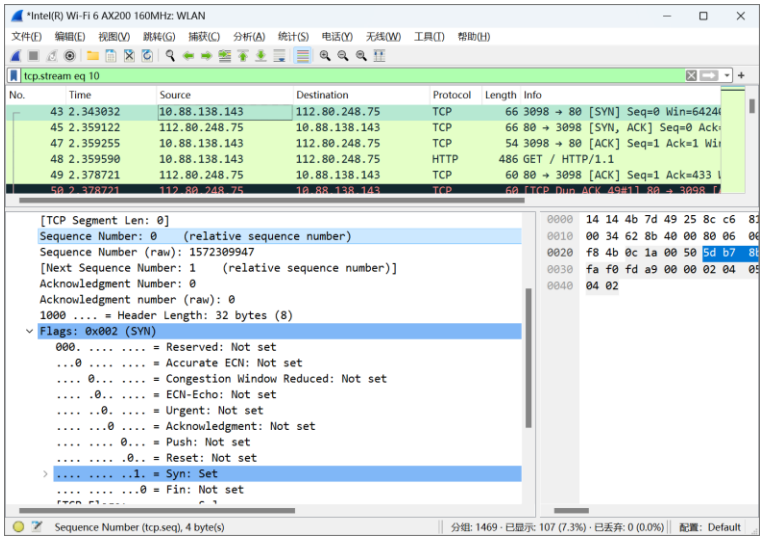
Step3：输入过滤条件获取待分析数据包列表 ip.addr == 112.80.248.75，并追踪 TCP 流。



图中可以看到 Wireshark 截获到了三次握手的三个数据包，第四个包才是 HTTP 的，这说明 HTTP 的确是使用 TCP 建立连接的。

- **第一次握手数据包**

客户端发送一个 TCP，标志位为 SYN，序列号为 0，代表客户端请求建立连接，如下图所示：

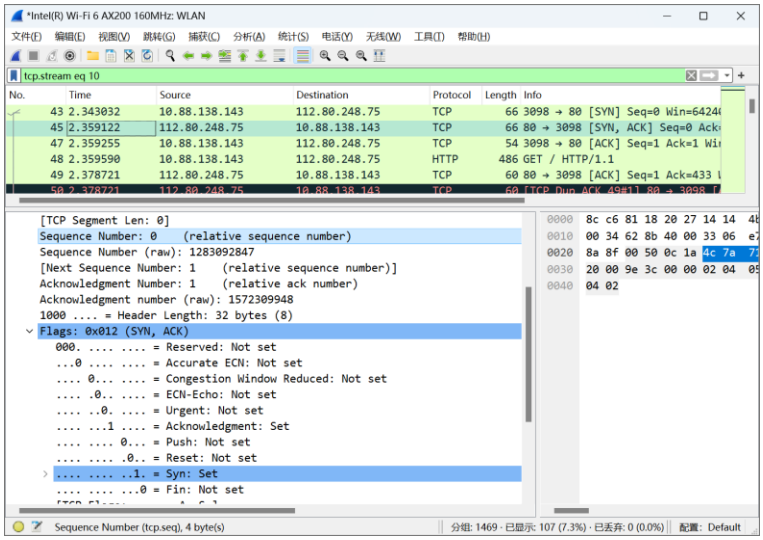


数据包的关键属性如下：

- SYN：标志位，表示请求建立连接
- Seq = 0：初始建立连接值为 0，数据包的相对序列号从 0 开始，表示当前还没有发送数据
- Ack = 0：初始建立连接值为 0，已经收到包的数量，表示当前没有接收到数据

- **第二次握手的数据包**

服务器发回确认包，标志位为 SYN，ACK。将确认序号(Acknowledgement Number)设置为客户的 ISN 加 1，如下图：

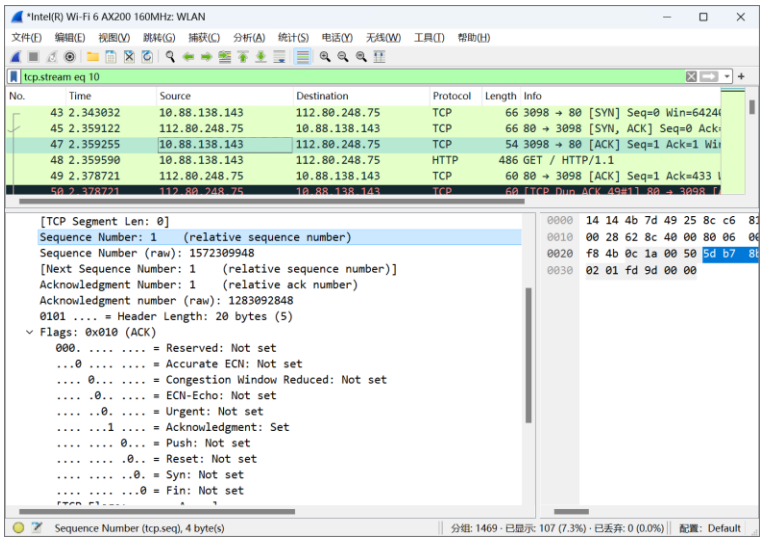


数据包的关键属性如下：

- [SYN + ACK]：标志位，同意建立连接，并回送 SYN+ACK
- Seq = 0：初始建立值为 0，表示当前还没有发送数据
- Ack = 1：表示当前端成功接收的数据位数，虽然客户端没有发送任何有效数据，确认号还是被加 1，因为包含 SYN 或 FIN 标志位。(并不会对有效数据的计数产生影响，因为含有 SYN 或 FIN 标志位的包并不携带有效数据)

- **第三次握手的数据包**

客户端再次发送确认包(ACK)SYN 标志位为 0，ACK 标志位为 1.并且把服务器发来 ACK 的序号字段+1，放在确定字段中发送给对方.并且在数据段放写 ISN 的+1，如下图：



数据包的关键属性如下：

- ACK：标志位，表示已经收到记录
- Seq = 1：表示当前已经发送 1 个数据
- Ack = 1：表示当前端成功接收的数据位数，虽然服务端没有发送任何有效数据，确认号还是被加 1，因为包含 SYN 或 FIN 标志位(并不会对有效数据的计数产生影响，因为含有 SYN 或 FIN 标志位的包并不携带有效数据)。

就这样通过了 TCP 三次握手，建立了连接。开始进行数据交互。
指导教师评语：
日期：