实验 2　Web 服务器配置，HTTP 报文捕获

一、实验要求

1.搭建 Web 服务器（自由选择系统），并制作简单 Web 页面，包含简单文本信息（至少包含专业、学号、姓名）。

2.通过浏览器获取自己编写的 Web 页面，使用 Wireshark 捕获与 Web 服务器的交互过程，并进行简单分析说明。

二、功能实现

1.Web 服务器搭建

基于 Windows 7 系统，使用 Internet 信息服务管理器（IIS）。

在 IIS 中选择添加网站，输入网站名称 hw2，选择应用程序池为 DefaultAppPool，选择网站内容目录的路径，设置 IP 地址为主机的 IPv4 地址：10.136.84.186，端口号设置为 8001。启用目录浏览功能，并将默认文档设置为 index.html，接下来选择浏览网站，测试网站运行是否成功。

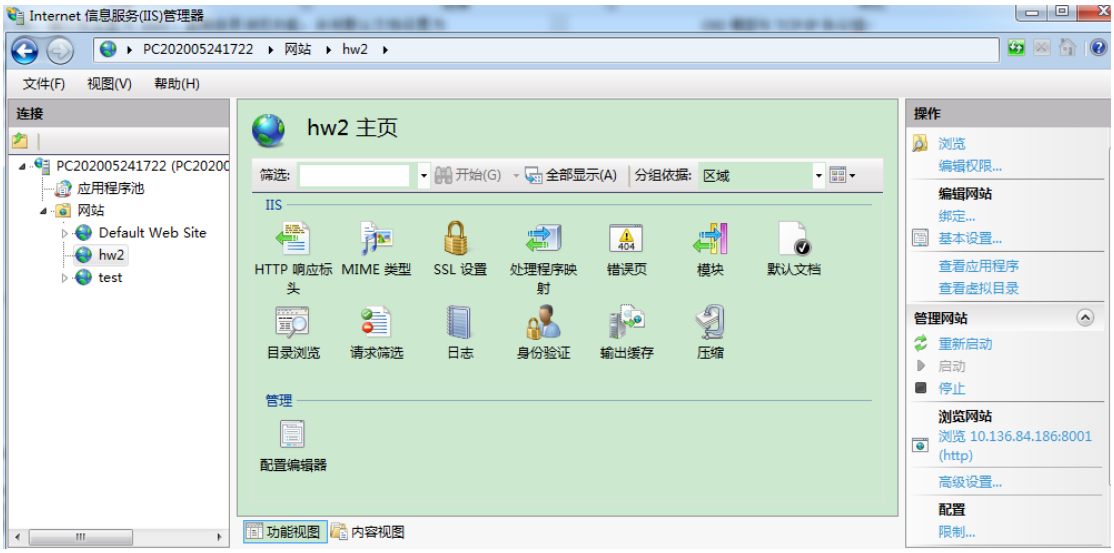
之后便能在浏览器输入 http://10.136.84.186:8001/来获取自己的网站。

在 hw2 文件夹下的 index.html 文件内容（包含简单的个人信息和两个链接）。

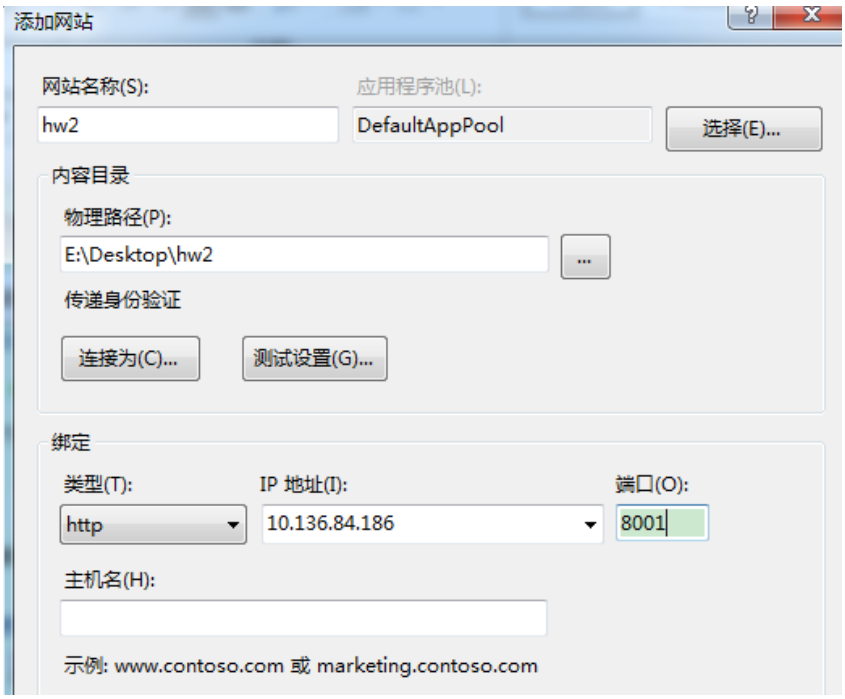
```
<body>
    <div class="main">
        <h1><b>计算机网络课程实验二</b></h1>
        <h2>专业：计算机科学与技术</h2>
        <h2>学号：1813800</h2>
        <h2>姓名：沈哲</h2>

        <table align="center">
            <tr>
            <td><a href="http://www.nankai.edu.cn/" target="_blank"><h3>南开大学</h3></a></td>
            <td><a href="http://www.lib.nankai.edu.cn/" target="_blank"><h3>南开大学图书馆</h3></a></td>
            </tr>
        </table>
    </div>
</body>
```

html 文件主要内容



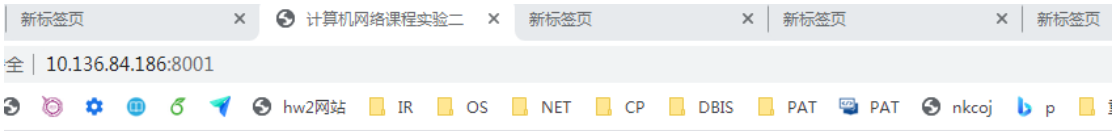
IIS 界面



在 IIS 中添加网站

2. 通过浏览器获取 Web 页面和 Wireshark 捕获交互过程

（1）在浏览器输入 http://10.136.84.186:8001/可以获取 Web 页面。



计算机网络课程实验二

专业：计算机科学与技术

学号：1813800

姓名：沈哲

南开大学　南开大学图书馆

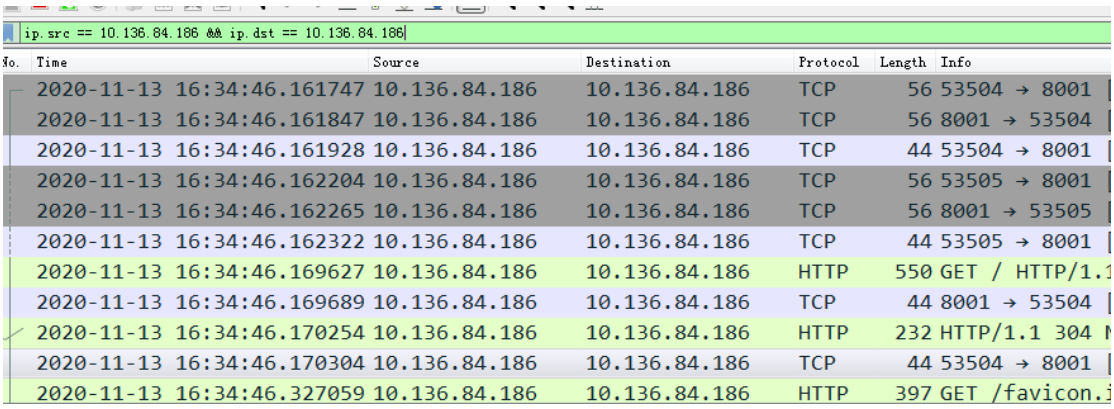
获取 Web 页面

（2）通过 Wireshark 捕获交互过程

在 Wireshark 中选择 Adapter for loopback traffic capture 可以方便地捕获本地环回的数据包。

输入筛选条件“ip.src == 10.136.84.186 && ip.dst == 10.136.84.186”可以只看关于此网站 ip 的数据包。

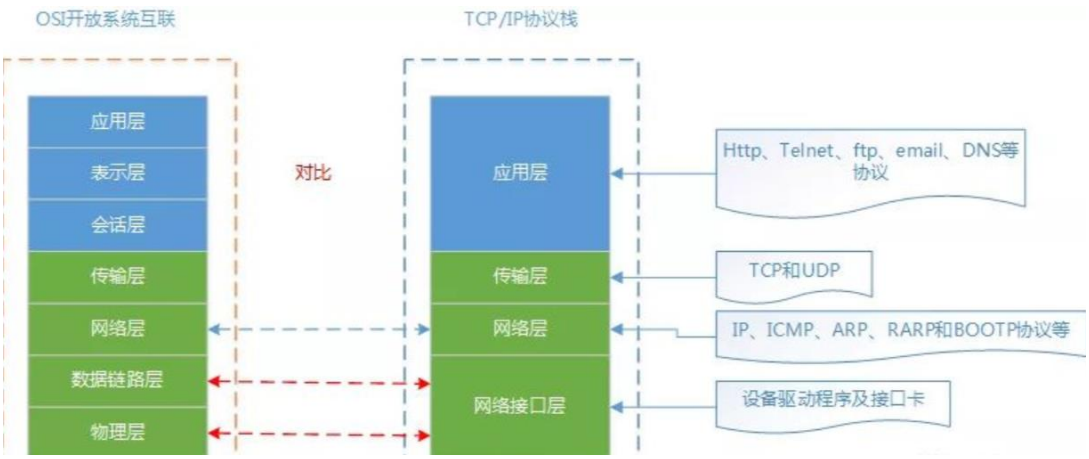
点击进入每一个分组可以查看 TCP/IP 协议栈的详细内容。



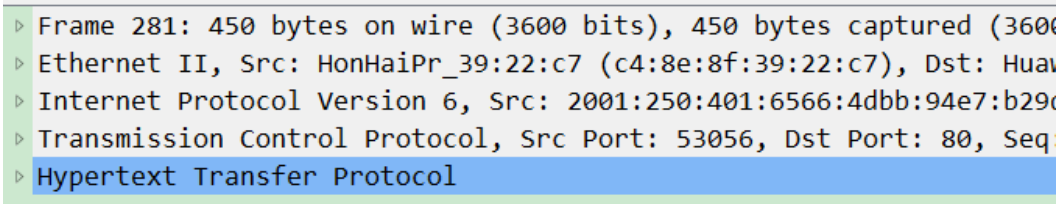
三、Wireshark 捕获结果分析

1.Wireshark 捕获结果介绍

通过 Wireshark 的抓包结果可以看出 TCP/IP 协议栈不同层次的内容，包括网络接口层（对应 OSI 模型的物理层、数据链路层）、网络层、传输层、应用层。



OSI 模型与 TCP/IP 协议栈



Wireshark 捕获信息中显示了每一层

每一层的含义如下：

Frame: 物理层的数据帧概况

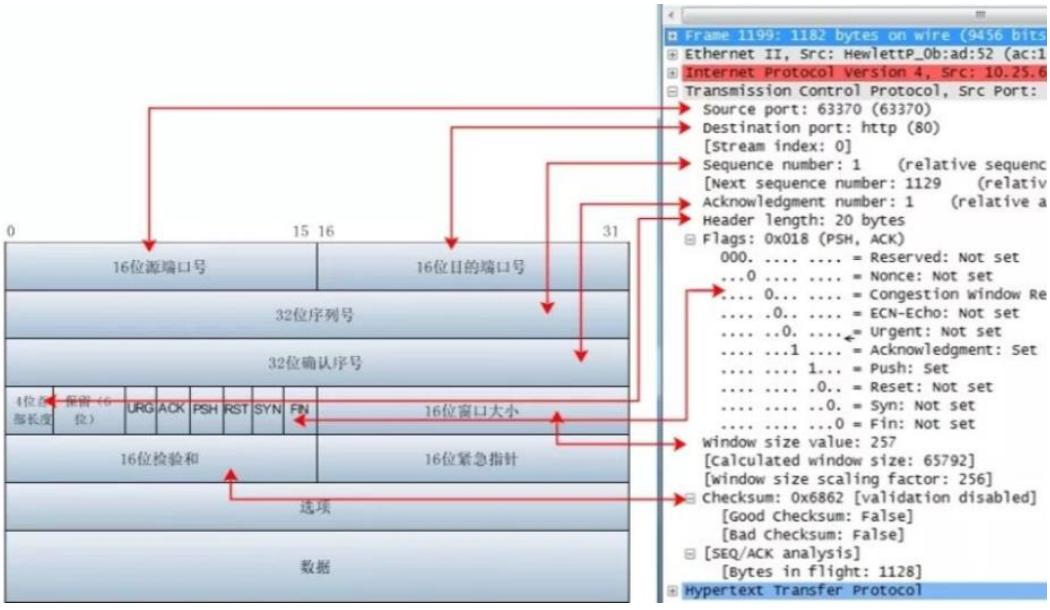
Ethernet II: 数据链路层以太网帧头部信息

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

Wireshark 捕获到的 TCP 包中的每个字段及对应含义如下图所示：



TCP 包中每个字段的含义

TCP 包中常用字段的含义如下：

源端口号：数据发起者的端口号，16bit

目的端口号：数据接收者的端口号，16bit

序号：32bit 的序列号，由发送方使用

确认序号：32bit 的确认号，是接收数据方期望收到发送方的下一个报文段的序号，因此确认序号应当是上次已成功收到数据字节序号加 1。

首部长度：首部中 32bit 字的数目，可表示 15*32bit=60 字节的首部。一般首部长度为 20 字节。

保留：6bit, 均为 0

紧急 URG：当 URG=1 时，表示报文段中有紧急数据，应尽快传送。

确认比特 ACK：ACK = 1 时代表这是一个确认的 TCP 包，取值 0 则不是确认包。

推送比特 PSH：当发送端 PSH=1 时，接收端尽快的交付给应用进程。

复位比特 RST：当 RST=1 时，表明 TCP 连接中出现严重差错，必须释放连接，再重新建立连接。

同步比特 SYN：在建立连接是用来同步序号。SYN=1， ACK=0 表示一个连接请求报文段。SYN=1，ACK=1 表示同意建立连接。

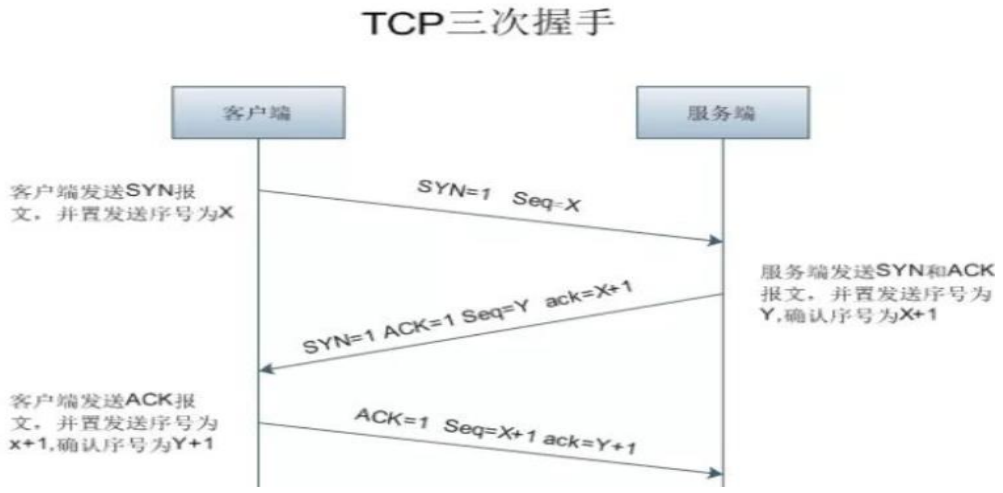
终止比特 FIN：FIN=1 时，表明此报文段的发送端的数据已经发送完毕，并要求释放传输连接。

2. TCP 协议中建立连接的“三次握手”

“三次握手”指建立一个 TCP 连接时，客户端和服务器总共发送 3 个包。在实验一的 socket 编程中，客户端执行 connect()时将触发三次握手。下图显示在建立 http 连接前先进行了三次 TCP 数据包的交互。

1	2020-11-13 11:16:47.048459	10.136.84.186	10.136.84.186	TCP	52	50746 → 8001 [SYN] Seq=0 Win=8192
2	2020-11-13 11:16:47.048581	10.136.84.186	10.136.84.186	TCP	52	8001 → 50746 [SYN, ACK] Seq=0 Ack=
3	2020-11-13 11:16:47.049064	10.136.84.186	10.136.84.186	TCP	44	50746 → 8001 [ACK] Seq=1 Ack=1 Win=
4	2020-11-13 11:16:47.058924	10.136.84.186	10.136.84.186	HTTP	576	GET / HTTP/1.1

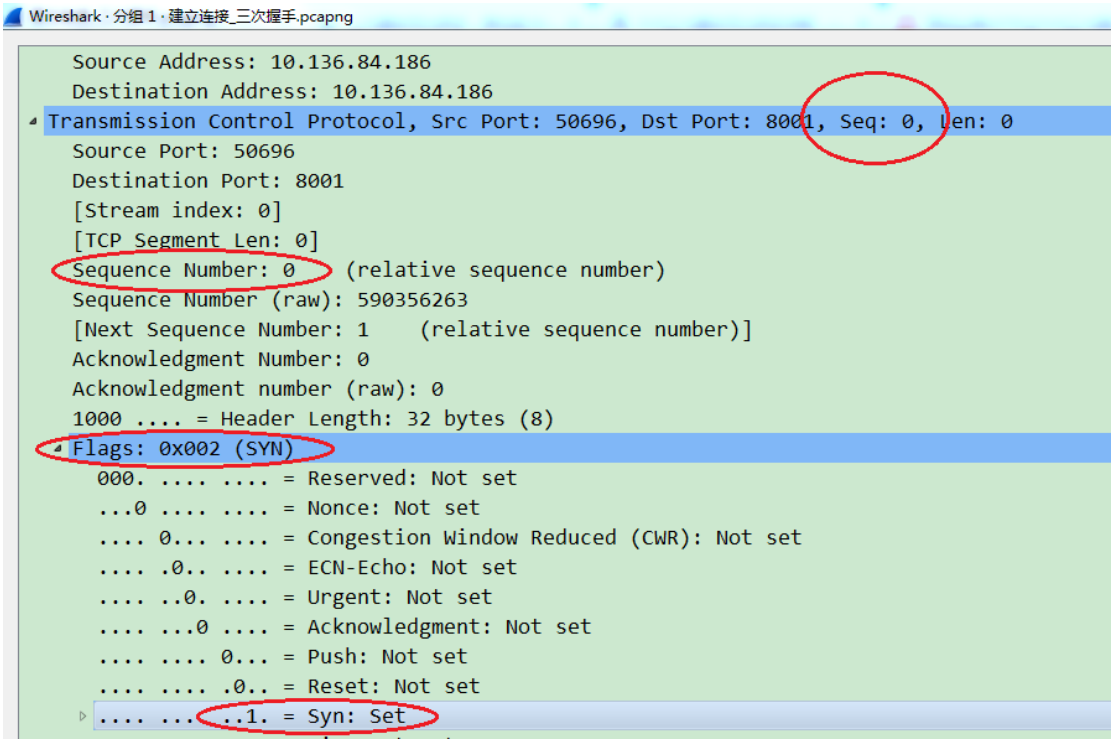
建立 TCP 连接时三次握手



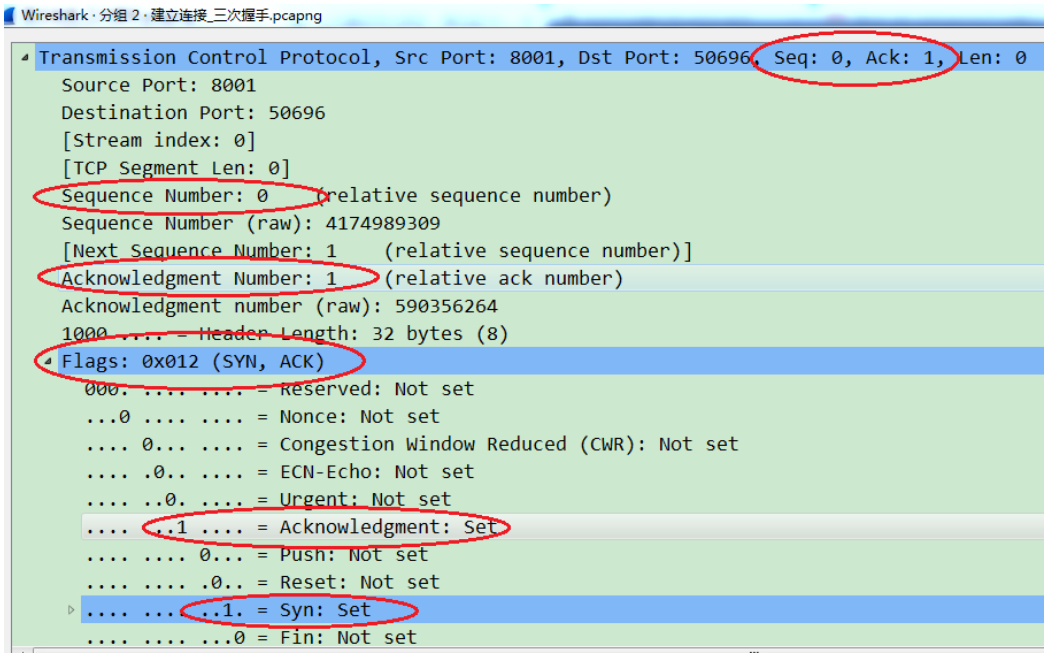
三次握手过程

下面分析三次握手过程，其中序列号显示的是相对数：

（1）第一次握手： 客户端向服务器发送连接请求包，标志位 SYN（同步序号）置为 1，序列号为 SeqX=0。



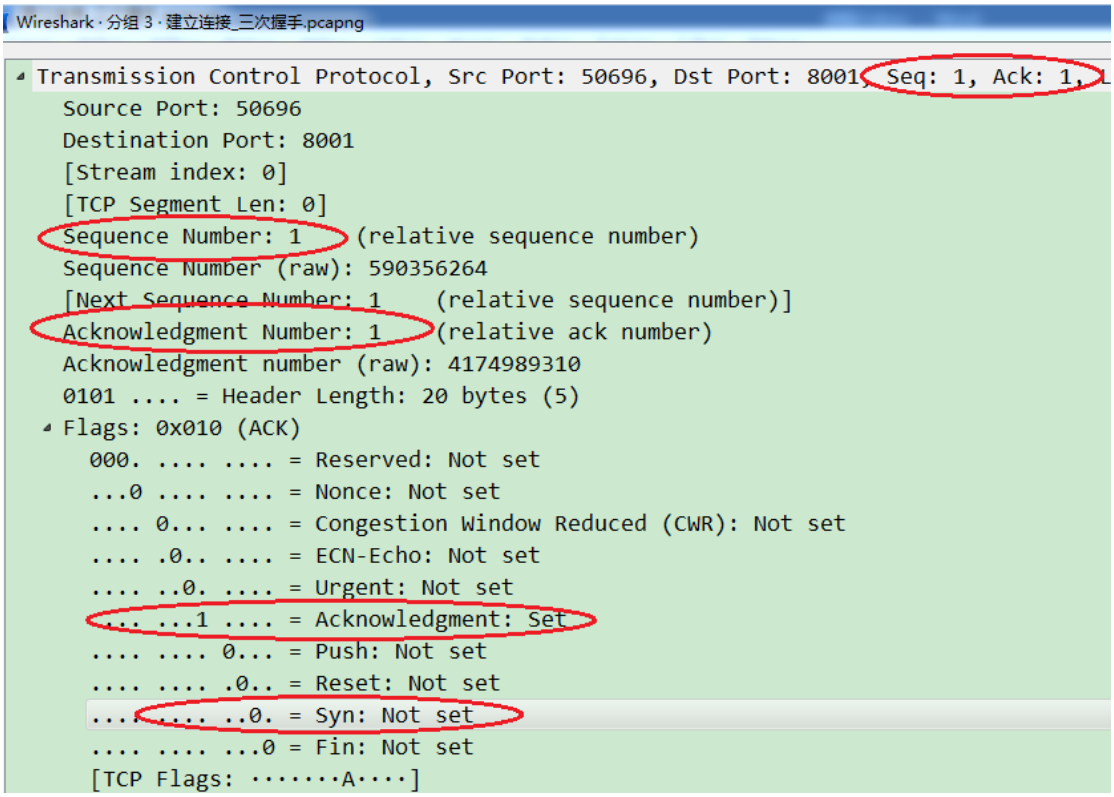
（2）第二次握手： 服务器收到客户端发过来报文，由 SYN=1 知道客户端要求建立联机。向客户端发送一个 SYN 和 ACK 都置为 1 的 TCP 报文，设置初始序列号 SeqY=0，将确认序号 AckNum 设置为客户的序列号加 1，即 AckNum=SeqX+1 = 0+1=1。



（3）第三次握手： 客户端收到服务器发来的包后检查确认序列号 AckNum（等于 SeqX+1=1）和标志位 ACK（等于 1）是否正确。

若正确，客户端再次发送确认包，ACK 标志位为 1，SYN 标志位为 0，发送序列号 SeqX=SeqX+1=1，确认序号 AckNum=SeqY+1=0+1=1。

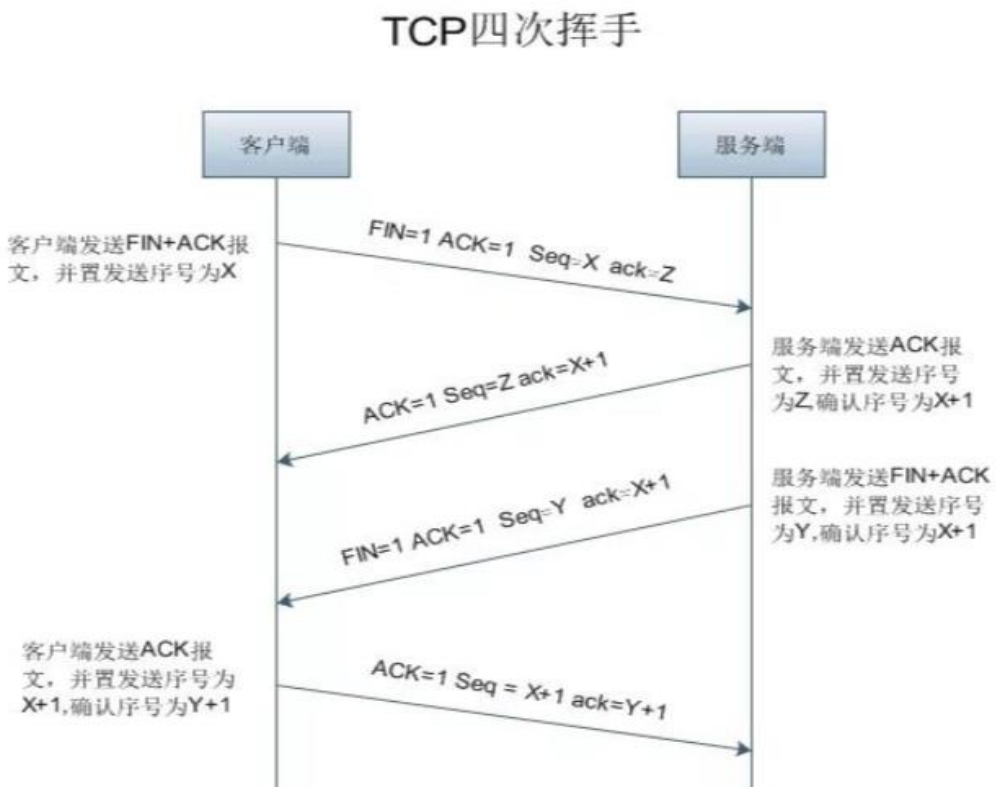
服务器收到后确认序号 AckNum=1 和 ACK=1 则连接建立成功，可以传送数据了。



三次握手过程中各个值的变化：

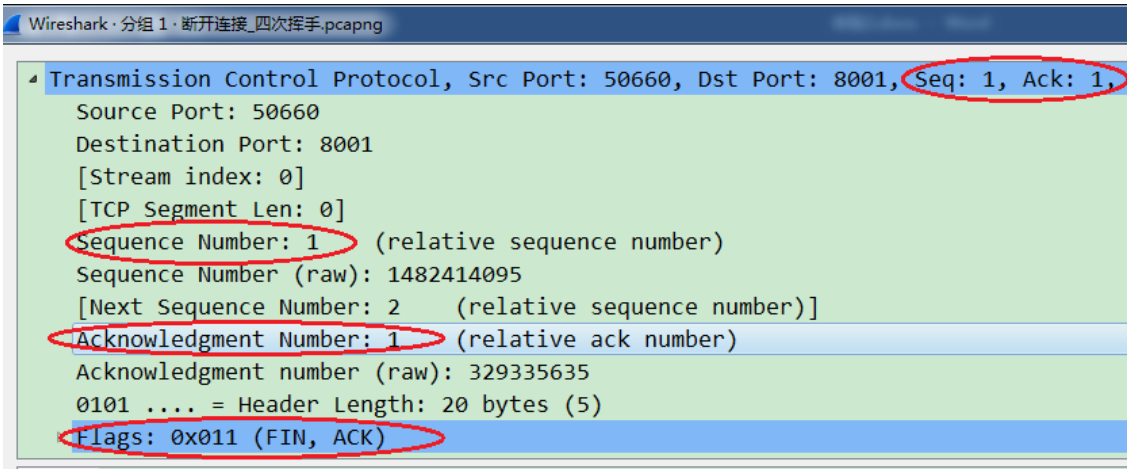
（1）客户端	SeqX=0	AckNum=0	SYN=1	ACK=0
（2）服务器	SeqY=0	AckNum=1	SYN=1	ACK=1
（3）客户端	SeqX=1	AckNum=1	SYN=0	ACK=1

3.TCP 协议中断开连接的“四次挥手”

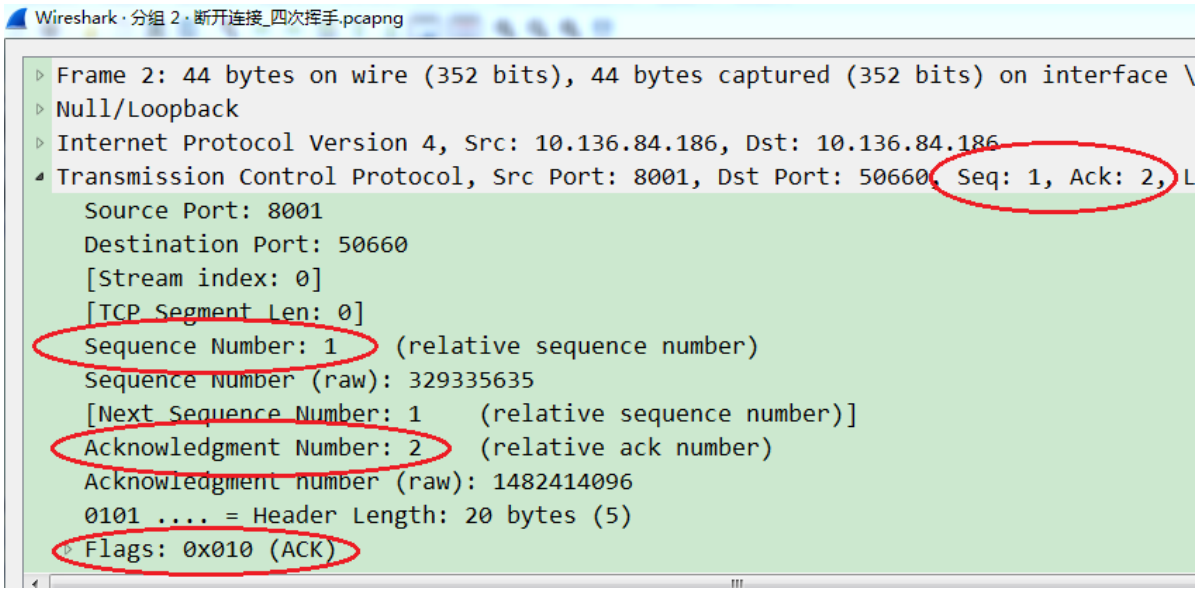


四次挥手过程

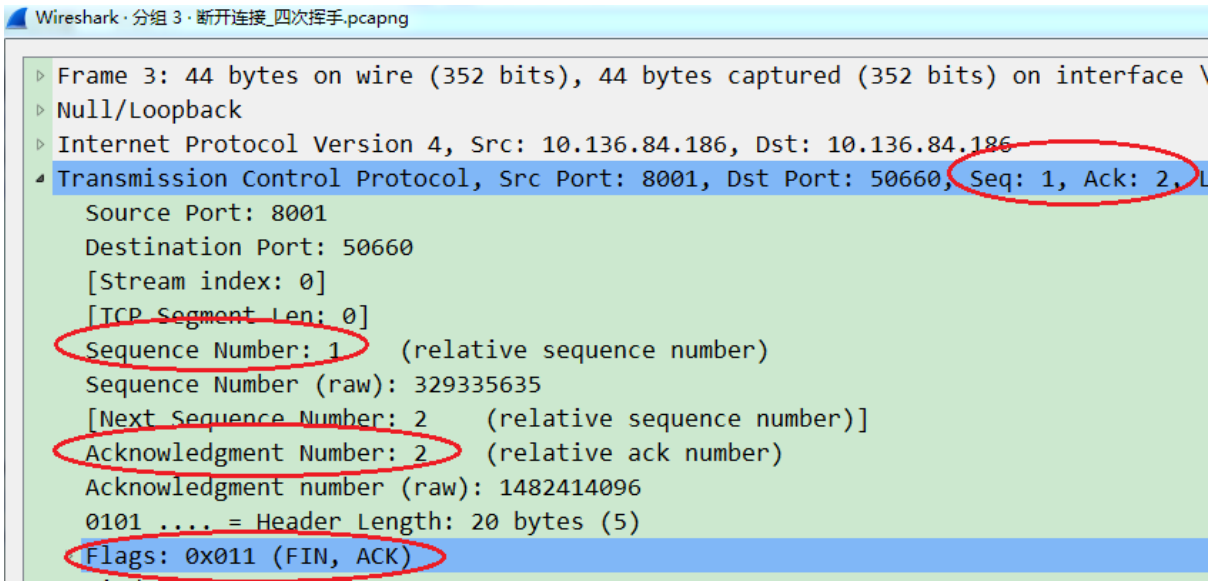
（1）第一次挥手： 客户端给服务器发送 TCP 包，用来关闭客户端到服务器的数据传送。将标志位 FIN 和 ACK 置为 1，序号为 SeqX=1，确认序号为 AckNum=1。



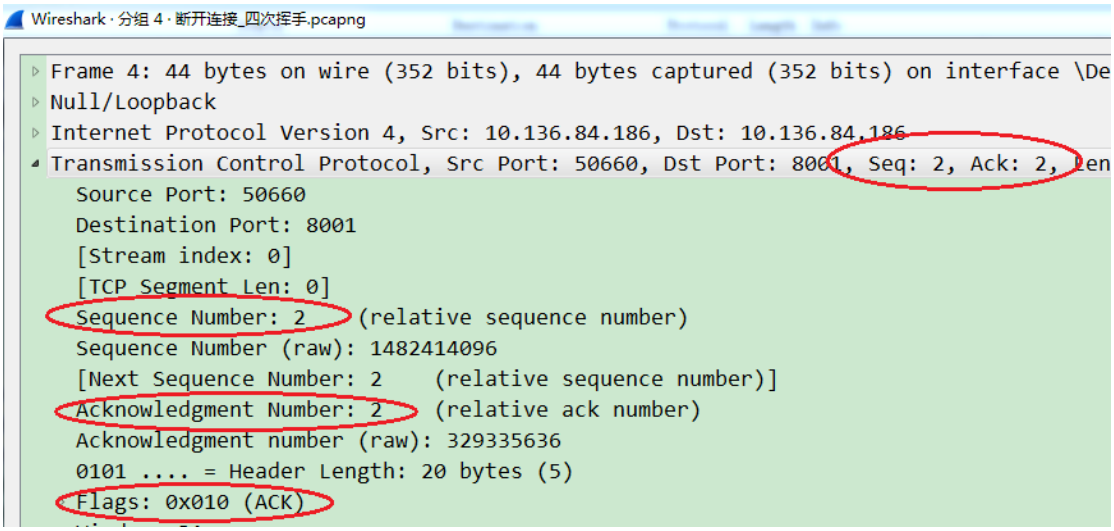
（2）第二次挥手：服务器收到 FIN 后，发回一个 ACK(标志位 ACK=1), 序号为收到的确认序号 SeqX =AckNum=1,确认序号为收到的序列号加 1，即 AckNum=SeqX+1=2。



（3）第三次挥手：服务器关闭与客户端的连接，发送一个 FIN。标志位 FIN 和 ACK 置为 1，序号为 SeqY=1，确认序号为 AckNum=2。



（4）第四次挥手：客户端收到服务器发送的 FIN 之后，发回 ACK 确认(标志位 ACK=1),序列号为收到的确认序号 SeqX=AckNum=2，确认序号为收到的序号加 1，即 AckNum=SeqY +1=2。



四次挥手过程中各个值的变化：

（1）客户端 SeqX=1	AckNum=1	FIN =1	ACK=1
（2）服务器 SeqY=1	AckNum=2	FIN =0	ACK=1
（3）服务器 SeqY=1	AckNum=2	FIN =1	ACK=1
（4）客户端 SeqX=2	AckNum=2	FIN =0	ACK=1

4.http 请求的简单分析

（1）http 请求的完整过程：

三次握手建立 TCP 连接。

建立 TCP 完成后，客户端向服务器发送请求命令，比如 GET https://www.baidu.com?name=xx&addr=xx HTTP1.1。

客户端发送请求头信息，发送完 header 后接着发送一个空白行，GET 请求没有数据，POST 请求要发送 body 数据。

服务器接收到以上信息后，开始处理，之后开始应答。

服务器返回响应头信息，再发送一个空白行。

服务器向客户端发送数据。

四次挥手关闭 TCP 连接。

（2）http 请求报文

http 请求报文包括请求行、请求头部、空行和请求数据四个部分。

```
Hypertext Transfer Protocol
  GET /main/s?user=people|2016people|fuceng&db=people&border=0&local=y
  Host: pmm.people.com.cn\r\n
  Connection: keep-alive\r\n
  User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (
  Accept: */*\r\n
  Referer: http://www.people.com.cn/\r\n
  Accept-Encoding: gzip, deflate\r\n
  Accept-Language: zh-CN,zh;q=0.9\r\n
  Cookie: ALLYESID4=1471C0DB9908C553; _people_ip_new_code=300000; wdci
  \r\n
```

请求行包括： 请求方法（GET），URL(包括参数信息)，协议版本信息（HTTP/1.1）。

请求头部(Header)是一个个的键值对，如：Host:…，User-Agent:…。

空行表示 header 和请求数据的分隔。

请求数据：GET 方法没有携带数据， POST 方法会携带一个 body。

（3）http 响应报文

http 响应报文包括状态行、响应头、空行、数据(响应体)。

```
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Fri, 13 Nov 2020 13:34:54 GMT\r\n
  Server: Apache/2.2.22 (Unix)\r\n
  P3P: CP="OTI PSA OUR"\r\n
  Cache-Control: no-store, no-cache\r\n
  Expires: -1\r\n
  Content-Length: 88\r\n
  Content-Type: text/html\r\n
  Connection: Keep-alive\r\n
  Via: 1.1 ID-0314217204545346 uproxy-2\r\n
  \r\n
  [HTTP response 6/7]
```

状态行包括：HTTP 版本号（如 HTTP/1.1），状态码（200,304,404 等）和状态值（OK, Not Modified 等）。

响应头类似请求头，是一系列键值对，如：Date:…,Server:…。

空白行来分隔 header 和数据

响应体：响应的数据，如 html 文件。

四、总结与展望

1.实验中的简单问题和解决方法

（1）搭建服务器过程中出现网页无法访问问题——修改网页文件权限（增加一个 Everyone 用户具有所有权限）即可解决。

（2）本地环回数据不经过路由器，难以捕获——使用 Npcap 功能，在 Wireshark 中选择 Adapter for loopback traffic capture 可以方便地捕获本地环回的数据包。

（3）无线网络连接会变化导致本地 IPv4 地址变化，自己的网页容易无法访问——不设置 IP 地址，直接用 localhost 或 127.0.0.1。

（4）开始访问网页会出现"GET /favicon.ico HTTP/1.1" 404——可以在网页文件中加入 favicon.ico 图标，对网页没有影响。

http 返回状态码为 304（表示已缓存，最近未修改过），如何返回 200——清空浏览器浏览记录，将缓存清空。

2.其他思考

在捕获时观察到“Keep-Alive”。

HTTP 协议的 Keep-Alive 目的是短时间内连接复用，希望可以短时间内在同一个连接上进行多次请求/响应。

TCP 的 KeepAlive 机制用于检测连接错误。TCP 的 keepalive 目的是保持客户端和服务端的连接，一方会不定期发送心跳包给另一方，当一方断掉的时候，没有断掉的定时发送几次心跳包，如果间隔发送几次，对方都返回的是 RST，而不是 ACK，就释放当前连接。这样防止了断开连接而不发送 FIN 的情况，避免影响服务器资源。

3.反思改进

本次实验学会了搭建 Web 服务器，之后可以尝试其他方式如使用 Apache 搭建，并尝试局域网内其他设备的访问（本次实验中偶尔能成功）。

学习了使用 Wireshark 捕获数据包并分析浏览器与服务器的交互过程，除了本机环回数据，也尝试分析了访问其他网站的捕获数据。

通过观察交互过程，分析各个字段和标志位值的变化，加深了对 TCP/IP 协议、http 报文格式等内容的理解，特别是“三次握手”“四次挥手”。

Wireshark 是通用的抓包工具，其他的还有 http 抓包工具 Fiddler 等，这些工具都有助于理解网络协议，今后还要继续实践。