

# 实验一：路由器基本配置

## 一、实验目的

- 1) 掌握路由器的基本知识；
- 2) 掌握路由器端口的配置；
- 3) 掌握路由协议的基本配置；
- 4) 熟悉使用 Boson Netsim 模拟器。

## 二、实验内容

- 1) 自行构建一个网络拓扑，要求包括 3 个以上路由器（路由器采用串行连接），用于连接两个以太网，每个以太网至少包括 1 台主机；
- 2) 完成路由器、主机等设备的配置，使用 RIP 或 OSPF 来维护路由器的路由表；
- 3) 实验配置完成后，两台主机要能够相互 ping 通。

## 三、实验步骤

- 1) 连接拓扑图；
- 2) 配置各个路由器的名称、端口 IP 地址、子网掩码、封装格式及时钟频率；
- 3) 配置各个主机的 IP 地址、子网掩码、以及与其相连的路由器端口地址；
- 4) 为各个路由器配置 RIP 协议。

## 四、实验过程及结果

- 1) 网络拓扑图：

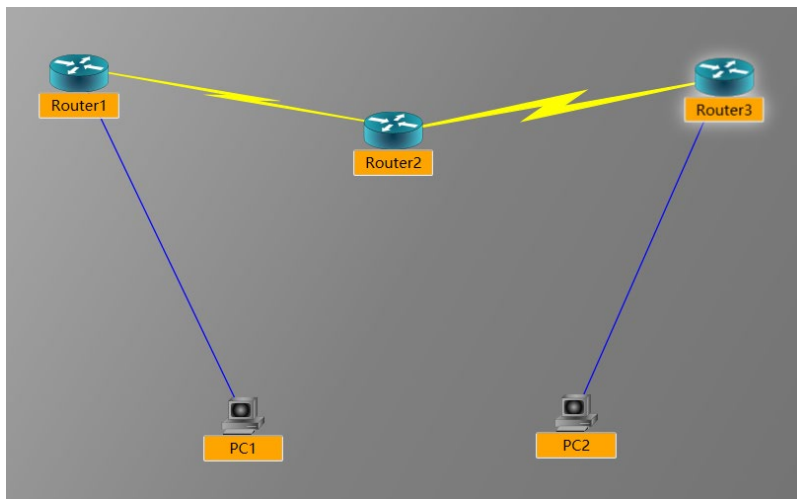


图 1-网络拓扑图

- 2) 路由器 ip 地址配置

R1: 左端口 192.168.1.1, 右端口 192.168.2.1

R2: 左端口 192.168.2.2, 右端口 192.168.3.1

R3: 左端口 192.168.3.2, 右端口 192.168.4.1

```

Press Enter to Start

Router>en
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname R1
R1(config)#int Ethernet0
R1(config-if)#ip address 192.168.1.1 255.255.255.0
R1(config-if)#no shut
00:01:12: %LINK-3-UPDOWN: Interface Ethernet0, changed state to up
00:01:13: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0, changed state to up
R1(config-if)#int Serial0
R1(config-if)#encapsulation hdlc
R1(config-if)#ip address 192.168.2.1 255.255.255.0
R1(config-if)#no shut
00:02:14: %LINK-3-UPDOWN: Interface Serial0, changed state to up
00:02:21: %LINK-3-UPDOWN: Interface Serial0, changed state to down
R1(config-if)#clock rate 64000
R1(config-if)#end
R1#

```

图 2-路由器 R1 配置 ip 地址

```

Press Enter to Start

Router>en
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname R2
R2(config)#int Serial0
R2(config-if)#encapsulation hdlc
R2(config-if)#ip address 192.168.2.2 255.255.255.0
R2(config-if)#no shut
00:06:03: %LINK-3-UPDOWN: Interface Serial0, changed state to up
00:06:05: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to up
R2(config-if)#clock rate 64000
R2(config-if)#int Serial0/0
R2(config-if)#encapsulation hdlc
R2(config-if)#ip address 192.168.3.1 255.255.255.0
R2(config-if)#no shut
00:07:00: %LINK-3-UPDOWN: Interface Serial0/0, changed state to up
00:07:01: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to up
00:07:07: %LINK-3-UPDOWN: Interface Serial0/0, changed state to down
00:07:07: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to down
R2(config-if)#clock rate 64000
R2(config-if)#end
R2#

```

图 3-路由器 R2 配置 ip 地址

```

Press Enter to Start

Router>en
Router#conf t
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname R3
R3(config)#int Ethernet0
R3(config-if)#ip address 192.168.4.1 255.255.255.0
R3(config-if)#no shut
00:09:03: %LINK-3-UPDOWN: Interface Ethernet0, changed state to up
00:09:04: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0, changed state to up
R3(config-if)#int Serial0/0
R3(config-if)#encapsulation hdlc
R3(config-if)#ip address 192.168.3.2 255.255.255.0
R3(config-if)#no shut
00:10:13: %LINK-3-UPDOWN: Interface Serial0/0, changed state to up
R3(config-if)#clock rate 64000
00:10:28: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0/0, changed state to up
R3(config-if)#end
R3#

```

图 4-路由器 R3 配置 ip 地址

### 3) 主机 ip 地址配置

PC1: ip 地址 192.168.1.2, 网关 192.168.1.1

PC2: ip 地址 192.168.4.2, 网关 192.168.4.1

```
Boson BOSS 5.0
Copyright 1998-2023 Boson Software, LLC.
Use the command help to get started

Press Enter to begin
C:>ipconfig /ip 192.168.1.2 255.255.255.0
C:>ipconfig /dg 192.168.1.1
C:>
```

图 5-主机 PC1 配置 ip 地址

```
Boson BOSS 5.0
Copyright 1998-2023 Boson Software, LLC.
Use the command help to get started

Press Enter to begin
C:>ipconfig /ip 192.168.4.2 255.255.255.0
C:>ipconfig /dg 192.168.4.1
C:>
```

图 6-主机 PC2 配置 ip 地址

#### 4) 路由器 RIP 协议配置

```
R1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#router rip
R1(config-router)#net 192.168.1.0
R1(config-router)#net 192.168.2.0
R1(config-router)#end
R1#
```

图 7-路由器 R1 配置 RIP 协议

```
R2#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R2(config)#router rip
R2(config-router)#net 192.168.2.0
R2(config-router)#net 192.168.3.0
R2(config-router)#end
R2#
```

图 8-路由器 R2 配置 RIP 协议

```
R3#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R3(config)#router rip
R3(config-router)#net 192.168.3.0
R3(config-router)#net 192.168.4.0
R3(config-router)#end
R3#
```

图 9-路由器 R3 配置 RIP 协议

#### 5) 双向 ping 通

```

C:>ping 192.168.4.2

Pinging 192.168.4.2 with 32 bytes of data:
Reply from 192.168.4.2: bytes=32 time=53ms TTL=241
Reply from 192.168.4.2: bytes=32 time=59ms TTL=241
Reply from 192.168.4.2: bytes=32 time=50ms TTL=241
Reply from 192.168.4.2: bytes=32 time=59ms TTL=241
Reply from 192.168.4.2: bytes=32 time=52ms TTL=241

Ping statistics for 192.168.4.2:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 50ms, Maximum = 59ms, Average = 55ms

C:>

```

图 10-主机 PC1ping 主机 PC2

```

C:>ping 192.168.1.2

Pinging 192.168.1.2 with 32 bytes of data:
Reply from 192.168.1.2: bytes=32 time=62ms TTL=241
Reply from 192.168.1.2: bytes=32 time=49ms TTL=241
Reply from 192.168.1.2: bytes=32 time=69ms TTL=241
Reply from 192.168.1.2: bytes=32 time=68ms TTL=241
Reply from 192.168.1.2: bytes=32 time=65ms TTL=241

Ping statistics for 192.168.1.2:
    Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 49ms, Maximum = 69ms, Average = 63ms

C:>

```

图 11-主机 PC2ping 主机 PC1

## 6) 路由表

```

R1#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route

Gateway of last resort is not set

C    192.168.1.0 is directly connected, Ethernet0
C    192.168.2.0 is directly connected, Serial0
R    192.168.3.0 [120/1] via 192.168.2.2, 00:02:41, Serial0
R    192.168.4.0 [120/2] via 192.168.2.2, 00:06:19, Serial0

R1#

```

图 12-路由器 R1 路由表

```

R2#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route

Gateway of last resort is not set

C    192.168.2.0 is directly connected, Serial0
C    192.168.3.0 is directly connected, Serial0/0
R    192.168.1.0 [120/1] via 192.168.2.1, 00:01:20, Serial0
R    192.168.4.0 [120/1] via 192.168.3.2, 00:01:33, Serial0/0

R2#

```

图 13-路由器 R2 路由表

```
R3#show ip route
Codes: C - connected, S - static, I - IGRP, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       E1 - OSPF external type 1, E2 - OSPF external type 2, E - EGP
       i - IS-IS, L1 - IS-IS level-1, L2 - IS-IS level-2, * - candidate default
       U - per-user static route

Gateway of last resort is not set

C    192.168.4.0 is directly connected, Ethernet0
C    192.168.3.0 is directly connected, Serial0/0
R    192.168.2.0 [120/1] via 192.168.3.1, 00:05:25, Serial0/0
R    192.168.1.0 [120/2] via 192.168.3.1, 00:02:40, Serial0/0

R3#
```

图 14-路由器 R3 路由表

## 五、心得体会

通过本次实验, 我掌握了 Boson Netsim 模拟器的使用方法, 了解了路由器的基本知识, 并学会了如何配置路由器端口 ip 地址和 RIP 路由协议, 成功实现了两台主机间的相互 ping 通。

# 实验二：分析 HTTP 协议

## 一、实验目的

- 1) 利用 wireshark 软件分析 HTTP 及其下层协议（TCP 协议）；
- 2) 了解网络中数据封装的概念；
- 3) 掌握 HTTP 及 TCP 协议的工作过程。

## 二、实验内容

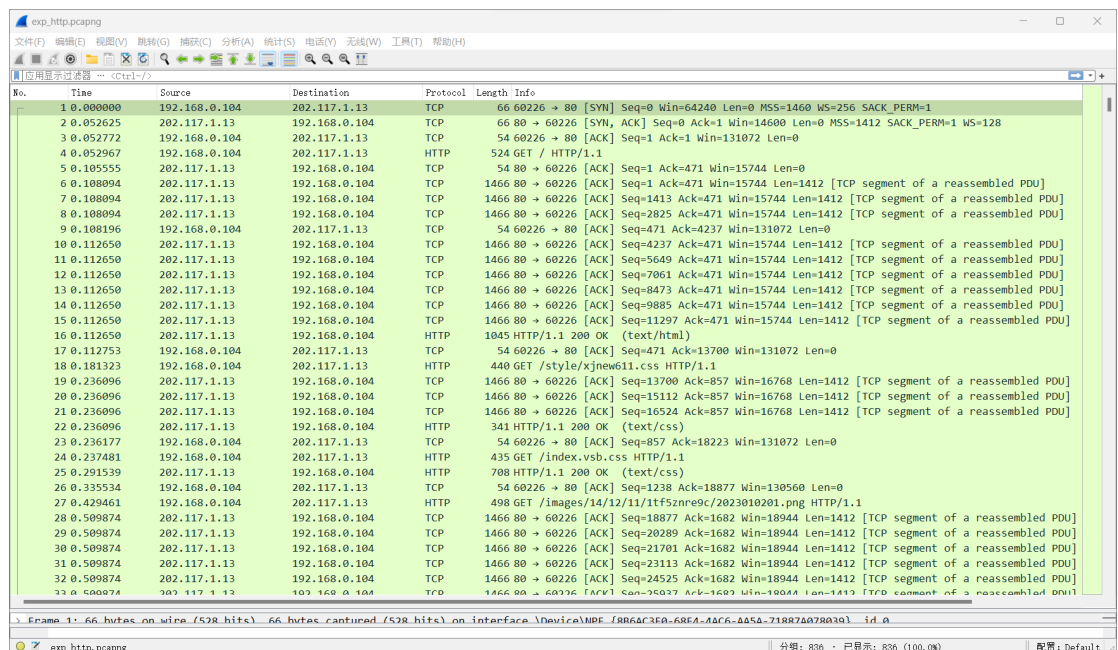
- 1) 启动 wireshark 软件，进行报文捕获；
- 2) 在浏览器访问 www.xjtu.edu.cn 页面（打开网页，浏览并关闭页面）；
- 3) 停止报文捕获，将捕获命名为“http—学号”；
- 4) 分析捕获报文。

## 三、实验步骤

- 1) 从捕获的报文中选择 HTTP 请求报文（即 get 报文）和 HTTP 应答报文，并分析各字段的值；
- 2) 综合分析捕获的报文，概括 HTTP 协议的工作过程；
- 3) 从捕获报文中选择 TCP 建立连接和释放连接的报文，分析各个字段的值并概括 TCP 协议的工作过程。

## 四、实验过程及结果

### 1) wireshark 捕获报文总览



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	192.168.0.104	202.117.1.13	TCP	66	60226 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2	0.052625	202.117.1.13	192.168.0.104	TCP	66	80 → 60226 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1412 SACK_PERM=1 WS=128
3	0.052772	192.168.0.104	202.117.1.13	TCP	54	60226 → 80 [ACK] Seq=1 Ack=1 Win=131072 Len=0
4	0.052967	192.168.0.104	202.117.1.13	HTTP	524	GET / HTTP/1.1
5	0.105555	202.117.1.13	192.168.0.104	TCP	54	80 → 60226 [ACK] Seq=1 Ack=471 Win=15744 Len=0
6	0.108094	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=1 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
7	0.108094	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=1413 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
8	0.108094	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=2825 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
9	0.108196	192.168.0.104	202.117.1.13	TCP	54	60226 → 80 [ACK] Seq=471 Ack=4237 Win=131072 Len=0
10	0.112650	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=4237 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
11	0.112650	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=5649 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
12	0.112650	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=7061 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
13	0.112650	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=8473 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
14	0.112650	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=9885 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
15	0.112650	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=11297 Ack=471 Win=15744 Len=1412 [TCP segment of a reassembled PDU]
16	0.112650	202.117.1.13	192.168.0.104	HTTP	1045	HTTP/1.1 200 OK (text/html)
17	0.112753	192.168.0.104	202.117.1.13	TCP	54	60226 → 80 [ACK] Seq=471 Ack=13700 Win=131072 Len=0
18	0.181323	192.168.0.104	202.117.1.13	HTTP	440	GET /style/xjnew611.css HTTP/1.1
19	0.236096	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=13700 Ack=857 Win=16768 Len=1412 [TCP segment of a reassembled PDU]
20	0.236096	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=15112 Ack=857 Win=16768 Len=1412 [TCP segment of a reassembled PDU]
21	0.236096	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=16524 Ack=857 Win=16768 Len=1412 [TCP segment of a reassembled PDU]
22	0.236096	202.117.1.13	192.168.0.104	HTTP	341	HTTP/1.1 200 OK (text/css)
23	0.236177	192.168.0.104	202.117.1.13	TCP	54	60226 → 80 [ACK] Seq=857 Ack=18223 Win=131072 Len=0
24	0.237481	192.168.0.104	202.117.1.13	HTTP	435	GET /index.vsb.css HTTP/1.1
25	0.291539	202.117.1.13	192.168.0.104	HTTP	708	HTTP/1.1 200 OK (text/css)
26	0.335534	192.168.0.104	202.117.1.13	TCP	54	60226 → 80 [ACK] Seq=1238 Ack=18877 Win=130560 Len=0
27	0.429461	192.168.0.104	202.117.1.13	HTTP	498	GET /images/14/12/11/11f52nre9c/2023010201.png HTTP/1.1
28	0.509874	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=18877 Ack=1682 Win=18944 Len=1412 [TCP segment of a reassembled PDU]
29	0.509874	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=20289 Ack=1682 Win=18944 Len=1412 [TCP segment of a reassembled PDU]
30	0.509874	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=21701 Ack=1682 Win=18944 Len=1412 [TCP segment of a reassembled PDU]
31	0.509874	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=23113 Ack=1682 Win=18944 Len=1412 [TCP segment of a reassembled PDU]
32	0.509874	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=24525 Ack=1682 Win=18944 Len=1412 [TCP segment of a reassembled PDU]
33	0.509874	202.117.1.13	192.168.0.104	TCP	1466	80 → 60226 [ACK] Seq=25037 Ack=1682 Win=18944 Len=1412 [TCP segment of a reassembled PDU]

图 1-报文总览

2) HTTP 请求报文

4	0.052967	192.168.0.104	202.117.1.13	HTTP	524 GET / HTTP/1.1
Frame 4: 524 bytes on wire (4192 bits), 524 bytes captured (4192 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0 Ethernet II, Src: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9), Dst: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92) Internet Protocol Version 4, Src: 192.168.0.104, Dst: 202.117.1.13 Transmission Control Protocol, Src Port: 60226, Dst Port: 80, Seq: 1, Ack: 1, Len: 470 Hypertext Transfer Protocol					
GET / HTTP/1.1\r\n					
[Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]					
[GET / HTTP/1.1\r\n]					
[Severity level: Chat]					
[Group: Sequence]					
Request Method: GET					
Request URI: /					
Request Version: HTTP/1.1					
Host: www.xjtu.edu.cn\r\n					
Connection: keep-alive\r\n					
Upgrade-Insecure-Requests: 1\r\n					
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/108.0.0.0 Safari/537.36 Edg/108.0.1462.76\r\n					
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n					
Accept-Encoding: gzip, deflate\r\n					
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6\r\n					
\r\n					
[Full request URI: http://www.xjtu.edu.cn/]					
[HTTP request 1/12]					
[Response in frame: 16]					
[Next request in frame: 18]					

图 2-HTTP 请求报文

分析：HTTP 请求报文第一行为请求行，请求方法为 GET，URL 为“/”表示服务器存放网页的根目录，HTTP 版本号为 1.1，回车符与换行符（\r\n）表示请求行的结束。

Host 字段指明了请求将要发送到的服务器主机名和端口号，这里是 [www.xjtu.edu.cn](http://www.xjtu.edu.cn)，端口号默认为 80，这个头部行同样以\r\n 结束。

Connection 字段指定连接类型为 keep-alive，表示网络连接是持久的。

Upgrade-Insecure-Requests 字段表示客户端对加密和认证响应的偏好。

User-Agent 字段包含客户端运行的浏览器类型。

Accept 字段指定客户端能够接收的内容类型，内容类型的先后次序表示客户端接收的先后次序，这里能够接收的类型有 html、xml、webp 等。

Accept-Encoding 字段指定内容编码类型为 gzip 或 deflate。

Accept-Language 字段表明哪些语言客户端是能够理解，并且其区域的变体是优选的。

最后，用一个空行表示头部行的结束。

3) HTTP 应答报文

16	0.112650	202.117.1.13	192.168.0.104	HTTP	1045 HTTP/1.1 200 OK (text/html)
Frame 16: 1045 bytes on wire (8360 bits), 1045 bytes captured (8360 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0 Ethernet II, Src: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92), Dst: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9) Internet Protocol Version 4, Src: 202.117.1.13, Dst: 192.168.0.104 Transmission Control Protocol, Src Port: 80, Dst Port: 60226, Seq: 12709, Ack: 471, Len: 991 [10 Reassembled TCP Segments (13699 bytes): #6(1412), #7(1412), #8(1412), #10(1412), #11(1412), #12(1412), #13(1412), #14(1412), #15(1412), #16(991)] Hypertext Transfer Protocol					
HTTP/1.1 200 OK\r\n					
Date: Sun, 08 Jan 2023 13:30:33 GMT\r\n					
Server: *****\r\n					
X-Frame-Options: SAMEORIGIN\r\n					
Last-Modified: Sun, 08 Jan 2023 11:52:47 GMT\r\n					
Accept-Ranges: bytes\r\n					
Cache-Control: max-age=600\r\n					
Expires: Sun, 08 Jan 2023 13:40:33 GMT\r\n					
Vary: Accept-Encoding\r\n					
Content-Encoding: gzip\r\n					
ETag: "e4e9-5f1bf4a0d81c0-gzip"\r\n					
Content-Length: 13250\r\n					
Keep-Alive: timeout=5, max=100\r\n					
Connection: Keep-Alive\r\n					
Content-Type: text/html\r\n					
Content-Language: zh-CN\r\n					
\r\n					
[HTTP response 1/12]					
[Time since request: 0.059683000 seconds]					
[Request in frame: 4]					
[Next request in frame: 18]					
[Next response in frame: 22]					
[Request URI: http://www.xjtu.edu.cn/]					
Content-encoded entity body (gzip): 13250 bytes -> 58601 bytes					
File Data: 58601 bytes					
Line-based text data: text/html (1006 lines)					

图 3-HTTP 应答报文



分析：HTTP 应答报文第一行首先说明了 HTTP 协议版本号为 1.1，然后给出状态码 200 及状态语句 OK 表示请求已成功。200 响应默认是可缓存的。回车符与换行符（\r\n）表示请求行的结束。

Date 字段表示消息发送的时间，这个头部行同样以\r\n 结束。

Server 字段表示服务器类型。

X-Frame-Options 字段指示浏览器是否应该加载一个 iframe 中的页面，值 SAMEORIGIN 表示页面只能被本站页面嵌入到 iframe 或者 frame 中。

Last-Modified 字段指明服务器对象的最后修订时间。

Accept-Ranges 字段用于告知客户端服务器是否能够处理范围请求，以指定获取服务器端某个部分的资源，当服务器可以处理范围请求时，指定为 bytes，反之则指定为 none。

Cache-Control 字段用以实现缓存机制，其值 max-age=600 表示如果缓存资源的缓存时间值小于 600 秒则使用缓存。

Expires 字段表示资源过期的日期，浏览器第一次请求资源，会保存 Expires 数据，下次再次请求该资源时，如果未过期，则不会向服务器发送请求，而是直接从缓存中获取。

Vary 字段用以告知下游的代理服务器，应当如何对以后的请求协议头进行匹配，以决定是否可使用已缓存的响应内容而不是重新从原服务器请求新的内容，其值 Accept-Encoding 表示当缓存服务器中相同请求的缓存数据的编码格式与当前请求的编码格式一致时返回缓存数据。

Content-Encoding 字段用于向接收方指示对象的类型，这里表示内容编码格式为 gzip。

ETag 字段表示资源的特定版本的标识符。它允许缓存更高效，并节省带宽，因为如果内容没有改变，Web 服务器不需要发送完整的响应。另一方面，如果内容发生了变化，etags 有助于防止资源的同时更新互相覆盖。

Content-Length 字段用来指明正文中返回对象的长度，这里是 13250bytes。

Keep-Alive 字段允许发送者提示关于如何连接，并且可以被用于设置超时时间以及请求的最大数量。其值 timeout=5 指示空闲连接必须保持打开的最短时间为 5 秒，max=100 指示在关闭它之前可以在此连接上发送的最大请求数为 100。

Connection 字段用于控制网络连接是否保持打开状态，其值 Keep-Alive 表示连接是持久的并且不关闭，从而允许对同一服务器的后续请求完成。

Content-Type 字段用于向接收方指示对象的类型，这里是 text/html。

Content-Language 字段用于指示正文中对象的语言，这里的 zh-CN 表示中文。

最后，用一个空行表示头部行的结束。

4) HTTP 协议的工作过程

域名解析 → TCP 三次握手建立连接 → 客户端发起 http 请求 → 服务器响应 http 请求并发送 Web 页面文件 → 浏览器解析 Web 页面文件并请求其中的资源（如 js、css、图片等） → 浏览器对页面进行渲染呈现给用户 → TCP 四次挥手释放连接

HTTP 协议如何使用 TCP 协议：当 HTTP 要传送一条报文时，会以流的形式将报文数据的内容通过一条打开的 TCP 连接按序传输。TCP 收到数据流之后，会将数据流分成被称作段的小数据块，并将段封装在 IP 分组中，通过因特网进行传输。

5) TCP 建立连接——三次握手

1 0.000000	192.168.0.104	202.117.1.13	TCP	66 60226 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
2 0.052625	202.117.1.13	192.168.0.104	TCP	66 80 → 60226 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1412 SACK_PERM=1 WS=128
3 0.052772	192.168.0.104	202.117.1.13	TCP	54 60226 → 80 [ACK] Seq=1 Ack=1 Win=131072 Len=0

图 4-三次握手



1 0.000000	192.168.0.104	202.117.1.13	TCP	66 60226 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK PERM=1
Frame 1: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0 Ethernet II, Src: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9), Dst: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92) Internet Protocol Version 4, Src: 192.168.0.104, Dst: 202.117.1.13 Transmission Control Protocol, Src Port: 60226, Dst Port: 80, Seq: 0, Len: 0				
Source Port: 60226 Destination Port: 80 [Stream index: 0] [Conversation completeness: Complete, WITH_DATA (31)] [TCP Segment Len: 0] Sequence Number: 0 (relative sequence number) Sequence Number (raw): 2878773094 [Next Sequence Number: 1 (relative sequence number)] Acknowledgment Number: 0 Acknowledgment number (raw): 0 1000 .... = Header Length: 32 bytes (8)				
▼ Flags: 0x002 (SYN)				
000. .... = Reserved: Not set ...0 .... = Nonce: Not set ... 0... = Congestion Window Reduced (CWR): Not set ... .0.. = ECN-Echo: Not set ... ..0. = Urgent: Not set ... ...0 = Acknowledgment: Not set ... .... 0... = Push: Not set ... ..... 0.. = Reset: Not set				
▶ .... ..1. = Syn: Set ... ..0 = Fin: Not set [TCP Flags: .....S.]				
Window: 64240 [Calculated window size: 64240] Checksum: 0x8cb9 [unverified] [Checksum Status: Unverified] Urgent Pointer: 0				
▶ Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted				
▶ [Timestamps]				

图 5-第一次握手

分析: Source Port 字段表示源端口为 60226, Destination Port 字段表示目的端口为 80。Sequence Number 表示序号为 0, Acknowledge Number 表示确认序号为 0 (无效, 因为 ACK=0)。

数据偏移字段为 1000 (4 位二进制, 单位为 4 字节), 表示报文头部总长度为 32 字节。保留字段占 6 位, 未使用。

标志位字段中 SYN 置 1, 表示要求建立连接。

Window 字段表示窗口, 用于控制对方所能发送的数据量, 这里为 64240 字节。

Checksum 字段表示校验和, 用于对 TCP 报文整体进行校验。

Urgent Pointer 字段表示紧急指针, 用于指出窗口中紧急数据的位置。

选项字段中最大报文长度值 MSS=1460 字节, 窗口缩放因子 WS=2^8=256, SACK PERM=1 允许 TCP 单独确认非连续的片段, 用于告知真正丢失的包, 只重传丢失的片段。

2 0.052625	202.117.1.13	192.168.0.104	TCP	66 80 → 60226 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1412 SACK PERM=1 WS=128
Frame 2: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0 Ethernet II, Src: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92), Dst: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9) Internet Protocol Version 4, Src: 202.117.1.13, Dst: 192.168.0.104 Transmission Control Protocol, Src Port: 80, Dst Port: 60226, Seq: 0, Ack: 1, Len: 0				
Source Port: 80 Destination Port: 60226 [Stream index: 0] [Conversation completeness: Complete, WITH_DATA (31)] [TCP Segment Len: 0] Sequence Number: 0 (relative sequence number) Sequence Number (raw): 2633659284 [Next Sequence Number: 1 (relative sequence number)] Acknowledgment Number: 1 (relative ack number) Acknowledgment number (raw): 2878773095 1000 .... = Header Length: 32 bytes (8)				
▼ Flags: 0x012 (SYN, ACK)				
000. .... = Reserved: Not set ...0 .... = Nonce: Not set ... 0... = Congestion Window Reduced (CWR): Not set ... .0.. = ECN-Echo: Not set ... ..0. = Urgent: Not set ... ..1. = Acknowledgment: Set ... .... 0... = Push: Not set ... ..... 0.. = Reset: Not set				
▶ .... ..1. = Syn: Set ... ..0 = Fin: Not set [TCP Flags: .....A..S.]				
Window: 14600 [Calculated window size: 14600] Checksum: 0x6a76 [unverified] [Checksum Status: Unverified] Urgent Pointer: 0				
▶ Options: (12 bytes), Maximum segment size, No-Operation (NOP), No-Operation (NOP), SACK permitted, No-Operation (NOP), Window scale				
▶ [Timestamps]				

图 6-第二次握手

分析: Source Port 字段表示源端口为 80, Destination Port 字段表示目的端口为 60226。  
Sequence Number 表示序号为 0, Acknowledge Number 表示确认序号为 1 (有效, 因为 ACK=1)。

数据偏移字段为 1000 (4 位二进制, 单位为 4 字节), 表示报文头部总长度为 32 字节。  
保留字段占 6 位, 未使用。

**标志位字段中 SYN 与 ACK 置 1, 表示对客户端发来的 SYN 报文段进行确认。**

Window 字段表示窗口, 用于控制对方所能发送的数据量, 这里为 14600 字节。

Checksum 字段表示校验和, 用于对 TCP 报文整体进行校验。

Urgent Pointer 字段表示紧急指针, 用于指出窗口中紧急数据的位置。

选项字段中最大报文长度值 MSS=1412 字节, 窗口缩放因子 WS=2<sup>7</sup>=128, SACK PERM=1 允许 TCP 单独确认非连续的片段, 用于告知真正丢失的包, 只重传丢失的片段。

```

3 0.052772      192.168.0.104      202.117.1.13      TCP      54 60226 → 80 [ACK] Seq=1 Ack=1 Win=131072 Len=0
Frame 3: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0
Ethernet II, Src: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9), Dst: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92)
Internet Protocol Version 4, Src: 192.168.0.104, Dst: 202.117.1.13
Transmission Control Protocol, Src Port: 60226, Dst Port: 80, Seq: 1, Ack: 1, Len: 0
  Source Port: 60226
  Destination Port: 80
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 1 (relative sequence number)
  Sequence Number (raw): 2878773095
  [Next Sequence Number: 1 (relative sequence number)]
  Acknowledgment Number: 1 (relative ack number)
  Acknowledgment number (raw): 2633659285
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x010 (ACK)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    ....0... = Congestion Window Reduced (CWR): Not set
    ....0... = ECN-Echo: Not set
    ....0... = Urgent: Not set
    ....1... = Acknowledgment: Set
    ....0... = Push: Not set
    ....0... = Reset: Not set
    ....0... = Syn: Not set
    ....0... = Fin: Not set
  [TCP Flags: .....A....]
  Window: 512
  [Calculated window size: 131072]
  [Window size scaling factor: 256]
  Checksum: 0x8cad [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > [Timestamps]
```

图 7-第三次握手

分析: Source Port 字段表示源端口为 60226, Destination Port 字段表示目的端口为 80。  
Sequence Number 表示序号为 1, Acknowledge Number 表示确认序号为 1 (有效, 因为 ACK=1)。

数据偏移字段为 0101 (4 位二进制, 单位为 4 字节), 表示报文头部总长度为 20 字节。  
保留字段占 6 位, 未使用。

**标志位字段中 ACK 置 1, 用于通知服务器双方已完成连接的建立。**

Window 字段表示窗口, 用于控制对方所能发送的数据量, 这里为 512 字节 (乘以窗口缩放因子 256 后, 实际大小为 131072 字节)。

Checksum 字段表示校验和, 用于对 TCP 报文整体进行校验。

Urgent Pointer 字段表示紧急指针, 用于指出窗口中紧急数据的位置。

选项字段为空。

三次握手总结: 首先, 主机 A 发送一个 SYN 位被置 1 的连接请求报文, 当主机 B 收到

这个报文后，发送 SYN 和 ACK 位均被置 1 的报文来对第一个 SYN 报文段进行确认。最后，主机 A 再发送一个 ACK 位被置 1 的确认报文，用来通知主机 B 双方已完成连接的建立。

6) TCP 释放连接——四次挥手

833	8.355836	202.117.1.13	192.168.0.104	TCP	54	80 → 60226	[FIN, ACK] Seq=982956 Ack=5409 Win=27520 Len=0
834	8.356012	192.168.0.104	202.117.1.13	TCP	54	60226 → 80	[ACK] Seq=5409 Ack=982957 Win=131072 Len=0
835	8.673394	192.168.0.104	202.117.1.13	TCP	54	60226 → 80	[FIN, ACK] Seq=5409 Ack=982957 Win=131072 Len=0
836	8.725133	202.117.1.13	192.168.0.104	TCP	54	80 → 60226	[ACK] Seq=982957 Ack=5410 Win=27520 Len=0

图 8-四次挥手

```
Frame 833: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0
Ethernet II, Src: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92), Dst: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9)
Internet Protocol Version 4, Src: 202.117.1.13, Dst: 192.168.0.104
Transmission Control Protocol, Src Port: 80, Dst Port: 60226, Seq: 982956, Ack: 5409, Len: 0
  Source Port: 80
  Destination Port: 60226
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 982956 (relative sequence number)
  Sequence Number (raw): 2634642240
  [Next Sequence Number: 982957 (relative sequence number)]
  Acknowledgment Number: 5409 (relative ack number)
  Acknowledgment number (raw): 2878778503
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x011 (FIN, ACK)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    ....0... = Congestion Window Reduced (CWR): Not set
    ....0... = ECN-Echo: Not set
    ....0... = Urgent: Not set
    ....1... = Acknowledgment: Set
    ....0... = Push: Not set
    ....0... = Reset: Not set
    ....0... = Syn: Not set
    ....1... = Fin: Set
  [TCP Flags: .....A...F]
  Window: 215
  [Calculated window size: 27520]
  [Window size scaling factor: 128]
  Checksum: 0xce6e [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  [Timestamps]
```

图 9-第一次挥手

分析: Source Port 字段表示源端口为 80, Destination Port 字段表示目的端口为 60226。Sequence Number 表示序号为 982956, Acknowledge Number 表示确认序号为 5409 (有效, 因为 ACK=1)。

数据偏移字段为 0101 (4 位二进制, 单位为 4 字节), 表示报文头部总长度为 20 字节。保留字段占 6 位, 未使用。

**标志位字段中 FIN 与 ACK 置 1, 表示释放从服务器到客户端的连接。**

Window 字段表示窗口, 用于控制对方所能发送的数据量, 这里为 215 字节 (乘以窗口缩放因子 128 后, 实际大小为 27520 字节)。

Checksum 字段表示校验和, 用于对 TCP 报文整体进行校验。

Urgent Pointer 字段表示紧急指针, 用于指出窗口中紧急数据的位置。

选项字段为空。

```

Frame 834: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0
Ethernet II, Src: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9), Dst: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92)
Internet Protocol Version 4, Src: 192.168.0.104, Dst: 202.117.1.13
Transmission Control Protocol, Src Port: 60226, Dst Port: 80, Seq: 5409, Ack: 982957, Len: 0
  Source Port: 60226
  Destination Port: 80
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 5409 (relative sequence number)
  Sequence Number (raw): 2878778503
  [Next Sequence Number: 5409 (relative sequence number)]
  Acknowledgment Number: 982957 (relative ack number)
  Acknowledgment number (raw): 2634642241
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x010 (ACK)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    ....0... = Congestion Window Reduced (CWR): Not set
    ....0... = ECN-Echo: Not set
    ....0... = Urgent: Not set
    ....1... = Acknowledgment: Set
    ....0... = Push: Not set
    ....0... = Reset: Not set
    ....0... = Syn: Not set
    ....0... = Fin: Not set
    [TCP Flags: .....A....]
  Window: 512
  [Calculated window size: 131072]
  [Window size scaling factor: 256]
  Checksum: 0x8cad [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]

```

图 10-第二次挥手

分析: Source Port 字段表示源端口为 60226, Destination Port 字段表示目的端口为 80。Sequence Number 表示序号为 5409, Acknowledge Number 表示确认序号为 982957 (有效, 因为 ACK=1)。

数据偏移字段为 0101 (4 位二进制, 单位为 4 字节), 表示报文头部总长度为 20 字节。保留字段占 6 位, 未使用。

**标志位字段中 ACK 置 1, 表示客户端的 TCP 进程对 FIN 报文段进行确认。**

Window 字段表示窗口, 用于控制对方所能发送的数据量, 这里为 512 字节 (乘以窗口缩放因子 256 后, 实际大小为 131072 字节)。

Checksum 字段表示校验和, 用于对 TCP 报文整体进行校验。

Urgent Pointer 字段表示紧急指针, 用于指出窗口中紧急数据的位置。

选项字段为空。

```

Frame 835: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0
Ethernet II, Src: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9), Dst: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92)
Internet Protocol Version 4, Src: 192.168.0.104, Dst: 202.117.1.13
Transmission Control Protocol, Src Port: 60226, Dst Port: 80, Seq: 5409, Ack: 982957, Len: 0
  Source Port: 60226
  Destination Port: 80
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 5409 (relative sequence number)
  Sequence Number (raw): 2878778503
  [Next Sequence Number: 5410 (relative sequence number)]
  Acknowledgment Number: 982957 (relative ack number)
  Acknowledgment number (raw): 2634642241
  0101 .... = Header Length: 20 bytes (5)
  Flags: 0x011 (FIN, ACK)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    ....0... = Congestion Window Reduced (CWR): Not set
    ....0... = ECN-Echo: Not set
    ....0... = Urgent: Not set
    ....1... = Acknowledgment: Set
    ....0... = Push: Not set
    ....0... = Reset: Not set
    ....0... = Syn: Not set
    > ....1... = Fin: Set
    > [TCP Flags: .....A...F]
  Window: 512
  [Calculated window size: 131072]
  [Window size scaling factor: 256]
  Checksum: 0x8cad [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > [Timestamps]

```

图 11-第三次挥手

分析: Source Port 字段表示源端口为 60226, Destination Port 字段表示目的端口为 80。  
Sequence Number 表示序号为 5409, Acknowledge Number 表示确认序号为 982957 (有效, 因为 ACK=1)。

数据偏移字段为 0101 (4 位二进制, 单位为 4 字节), 表示报文头部总长度为 20 字节。  
保留字段占 6 位, 未使用。

**标志位字段中 FIN 与 ACK 置 1, 表示释放从客户端到服务器的连接。**

Window 字段表示窗口, 用于控制对方所能发送的数据量, 这里为 512 字节 (乘以窗口缩放因子 256 后, 实际大小为 131072 字节)。

Checksum 字段表示校验和, 用于对 TCP 报文整体进行校验。

Urgent Pointer 字段表示紧急指针, 用于指出窗口中紧急数据的位置。

选项字段为空。

```
Frame 836: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface \Device\NPF_{8B6AC3F0-68F4-4AC6-AA5A-71887A078039}, id 0
Ethernet II, Src: MercuryC_cc:6e:92 (8c:f2:28:cc:6e:92), Dst: IntelCor_53:49:a9 (f8:e4:e3:53:49:a9)
Internet Protocol Version 4, Src: 202.117.1.13, Dst: 192.168.0.104
Transmission Control Protocol, Src Port: 80, Dst Port: 60226, Seq: 982957, Ack: 5410, Len: 0
  Source Port: 80
  Destination Port: 60226
  [Stream index: 0]
  [Conversation completeness: Complete, WITH_DATA (31)]
  [TCP Segment Len: 0]
  Sequence Number: 982957 (relative sequence number)
  Sequence Number (raw): 2634642241
  [Next Sequence Number: 982957 (relative sequence number)]
  Acknowledgment Number: 5410 (relative ack number)
  Acknowledgment number (raw): 2878778504
  0101 .... = Header Length: 20 bytes (5)
  ✓ Flags: 0x010 (ACK)
    000. .... = Reserved: Not set
    ...0 .... = Nonce: Not set
    ....0... = Congestion Window Reduced (CWR): Not set
    ....0... = ECN-Echo: Not set
    ....0... = Urgent: Not set
    ....1... = Acknowledgment: Set
    ....0... = Push: Not set
    ....0... = Reset: Not set
    ....0... = Syn: Not set
    ....0... = Fin: Not set
  [TCP Flags: .....A....]
  Window: 215
  [Calculated window size: 27520]
  [Window size scaling factor: 128]
  Checksum: 0xce6d [unverified]
  [Checksum Status: Unverified]
  Urgent Pointer: 0
  > [Timestamps]
  > [SEQ/ACK analysis]
```

图 12-第四次挥手

分析: Source Port 字段表示源端口为 80, Destination Port 字段表示目的端口为 60226。  
Sequence Number 表示序号为 982957, Acknowledge Number 表示确认序号为 5410 (有效, 因为 ACK=1)。

数据偏移字段为 0101 (4 位二进制, 单位为 4 字节), 表示报文头部总长度为 20 字节。  
保留字段占 6 位, 未使用。

**标志位字段中 ACK 置 1, 表示服务器的 TCP 进程对 FIN 报文段进行确认。**

Window 字段表示窗口, 用于控制对方所能发送的数据量, 这里为 215 字节 (乘以窗口缩放因子 128 后, 实际大小为 27520 字节)。

Checksum 字段表示校验和, 用于对 TCP 报文整体进行校验。

Urgent Pointer 字段表示紧急指针, 用于指出窗口中紧急数据的位置。

选项字段为空。

四次挥手总结: TCP 采用对称的连接释放方式, 即对每个方向的连接单独释放。当关闭一个方向的连接时, 连接释放的发起方发送一个 FIN 标志位被置 1 的 TCP 报文, 然后接收

方对 FIN 报文段进行确认，发送 ACK 报文，并通知应用程序，整个通信会话已结束。当连接的两个方向都已关闭时，该连接的两个端点的 TCP 进程将删除这个连接记录。

## 五、心得体会

通过本次实验，我掌握了使用 wireshark 进行抓包的方法，并通过抓包学习了 HTTP 请求与响应报文的格式，以及 TCP 协议建立连接时的三次握手过程和释放连接时的四次挥手过程。