西安交通大學

计算机网络原理
课内实验作业
（两次）

姓名：周海鹏

班级：计算机94

学号：2194313178

2022年12月9日

实验一： 基于 Boson Netsim 软件的路由器配置实验

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

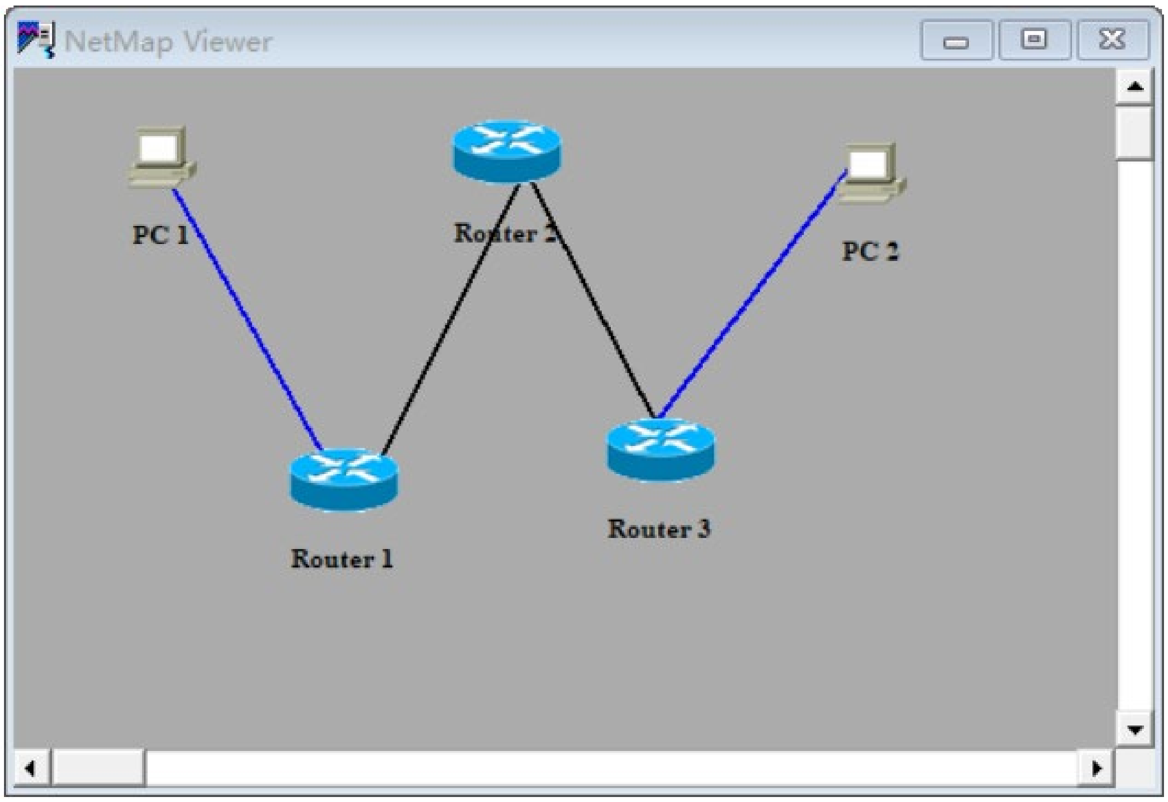
1. 自行构建一个网络拓扑，要求包括 3 个以上路由器，用于连接两个以太网，每个以太网至少包括 1 台主机；
2. 路由器配置RIP 或OSPF 来维护路由表；
3. 任意两个主机之间通过ping 命令能够连通；

三、实验步骤

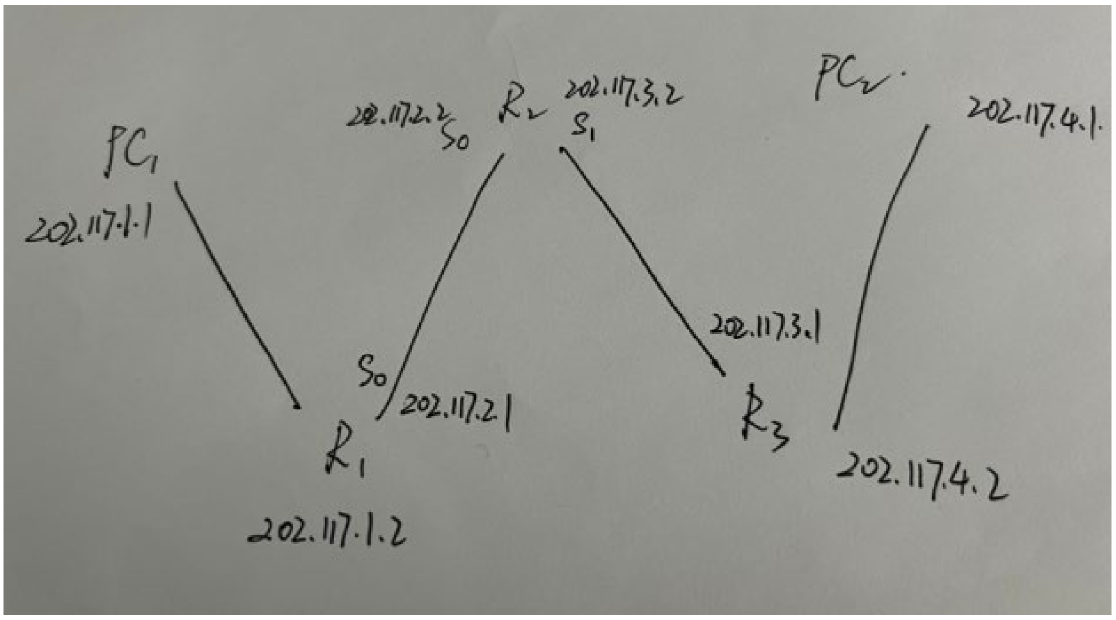
1. 连接拓扑图，两个主机通过三个路由器实现连接
2. 配置各个路由器的名称，端口IP 地址，子网掩码，封装格式及时钟频率
3. 配置各个主机的IP 地址，子网掩码，以及与其相连的路由器端口地址
4. 配置RIP 协议，实现路由选择及IP 分组转发

四、实验过程及结果

拓扑图：



具体ip如下：



先对路由器进行ip分配

R1:

enable

ip config

int e0

ipaddress202.117.1.2255.255.255.0

no shutdown

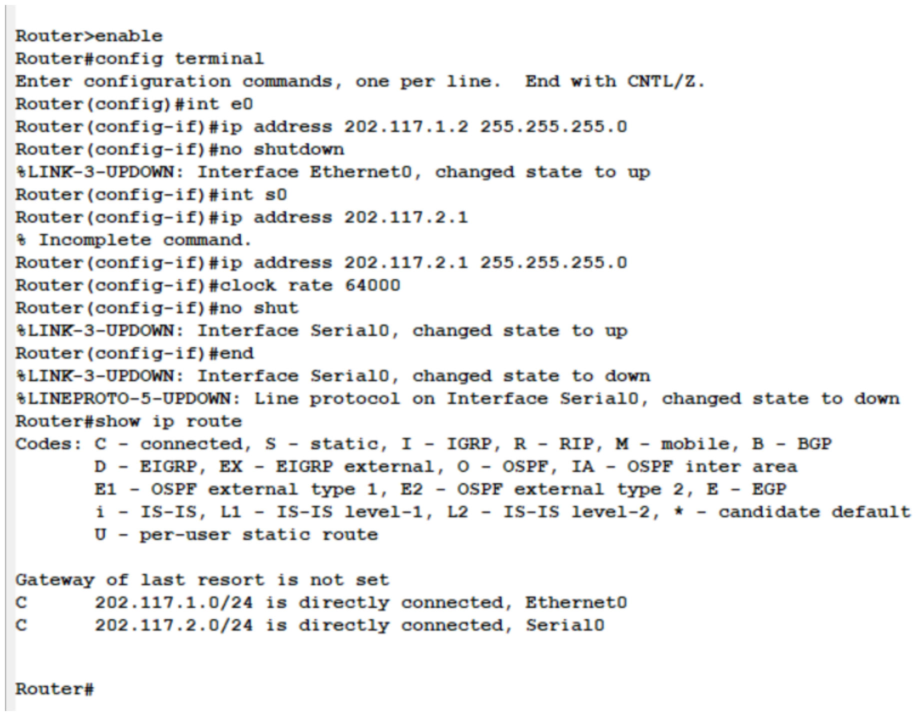
int s0

ip address202.117.2.1 255.255.255.0

clock rate 64000

no shut

end



```
Router>enable
Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int e0
Router(config-if)#ip address 202.117.1.2 255.255.255.0
Router(config-if)#no shutdown
%LINK-3-UPDOWN: Interface Ethernet0, changed state to up
Router(config-if)#int s0
Router(config-if)#ip address 202.117.2.1
% Incomplete command.
Router(config-if)#ip address 202.117.2.1 255.255.255.0
Router(config-if)#clock rate 64000
Router(config-if)#no shut
%LINK-3-UPDOWN: Interface Serial0, changed state to up
Router(config-if)#end
%LINK-3-UPDOWN: Interface Serial0, changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to down
Router#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      202.117.1.0/24 is directly connected, Ethernet0
C      202.117.2.0/24 is directly connected, Serial0


Router#
```

R2:

enable

config terminal

int e0

ip address 202.117.4.2 255.255.255.0

no shut

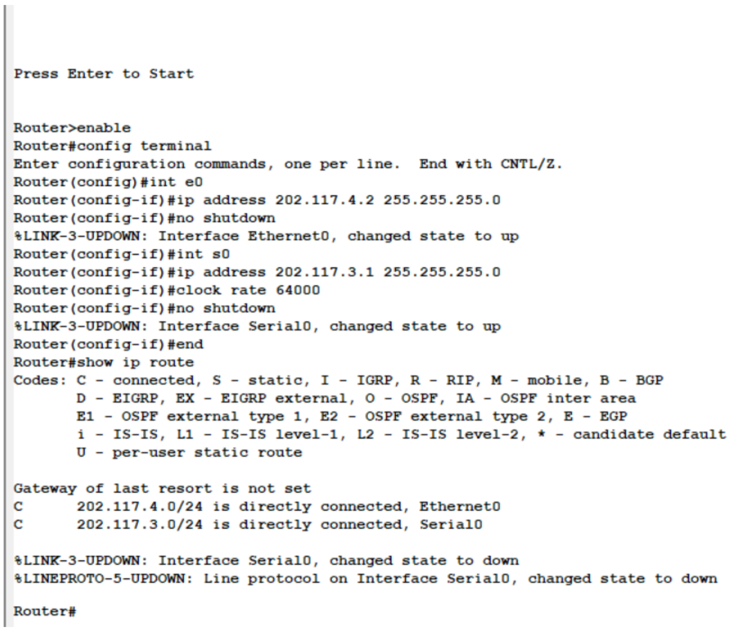
int s0

ip address 202.117.3.1 255.255.255.0

clock rate 64000

no shut

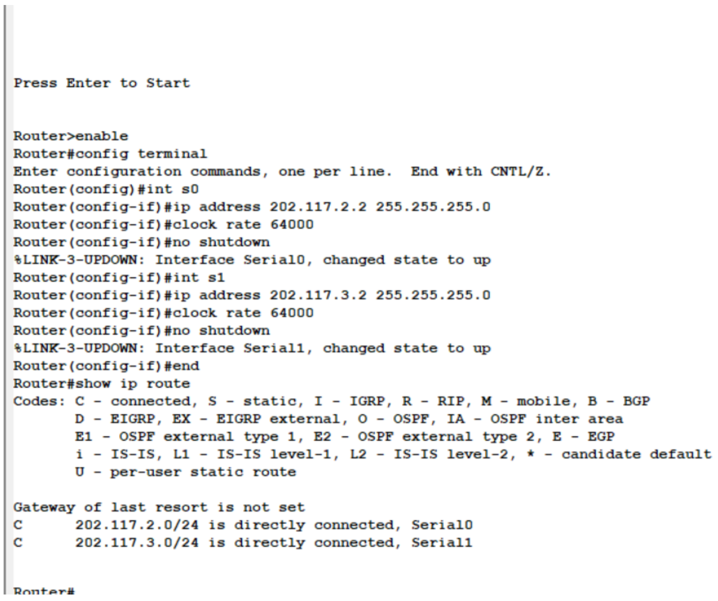
end



```
Press Enter to Start


Router>enable
Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int e0
Router(config-if)#ip address 202.117.4.2 255.255.255.0
Router(config-if)#no shutdown
%LINK-3-UPDOWN: Interface Ethernet0, changed state to up
Router(config-if)#int s0
Router(config-if)#ip address 202.117.3.1 255.255.255.0
Router(config-if)#clock rate 64000
Router(config-if)#no shutdown
%LINK-3-UPDOWN: Interface Serial0, changed state to up
Router(config-if)#end
Router#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      202.117.4.0/24 is directly connected, Ethernet0
C      202.117.3.0/24 is directly connected, Serial0

%LINK-3-UPDOWN: Interface Serial0, changed state to down
%LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to down

Router#
```

R3:

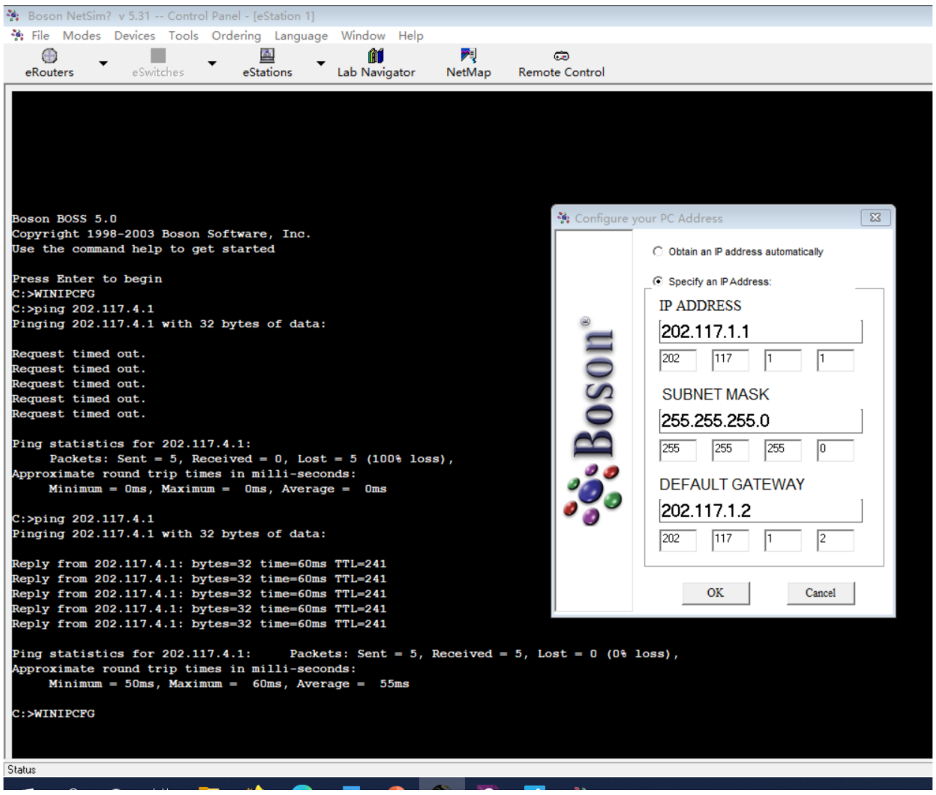
```
enable
config terminal
int s0
ip address 202.117.2.2 255.255.255.0
clock rate 64000
no shut
int s1
ip address 202.117.3.2 255.255.255.0
clock rate 64000
noshut
end
```



```
Press Enter to Start


Router>enable
Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#int s0
Router(config-if)#ip address 202.117.2.2 255.255.255.0
Router(config-if)#clock rate 64000
Router(config-if)#no shutdown
%LINK-3-UPDOWN: Interface Serial0, changed state to up
Router(config-if)#int s1
Router(config-if)#ip address 202.117.3.2 255.255.255.0
Router(config-if)#clock rate 64000
Router(config-if)#no shutdown
%LINK-3-UPDOWN: Interface Serial1, changed state to up
Router(config-if)#end
Router#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      202.117.2.0/24 is directly connected, Serial0
C      202.117.3.0/24 is directly connected, Serial1

Router#
```

接下来是主机

PC1：

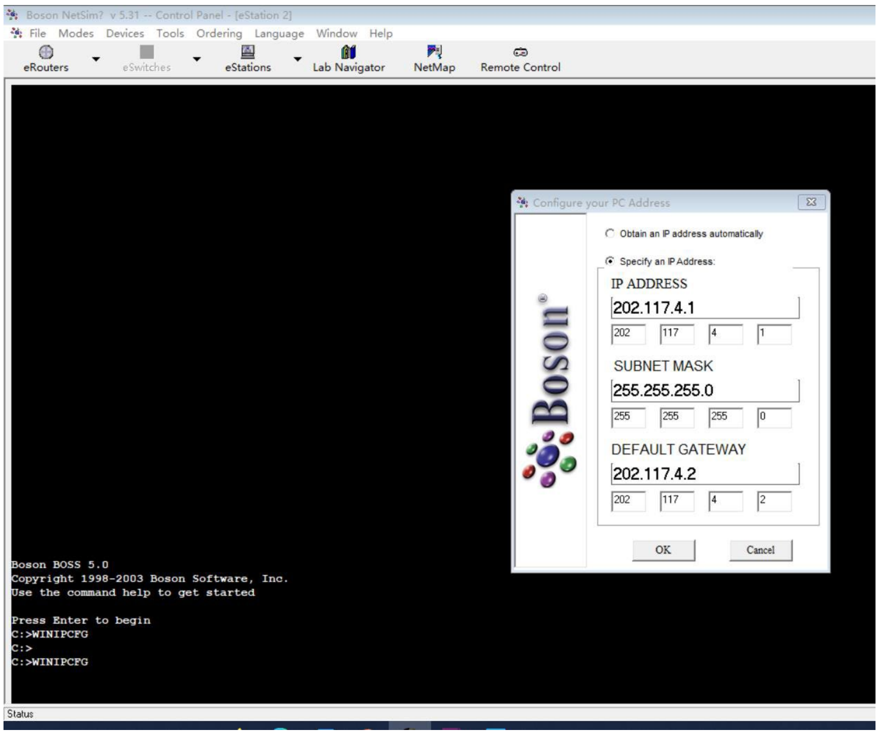

Boson NetSim? v 5.31 -- Control Panel - [eStation 1]
File Modes Devices Tools Ordering Language Window Help
eRouters
eSwitches
eStations
Lab Navigator
NetMap
Remote Control
Configure your PC Address
Boson
OK
Cancel
Status

```
Boson BOSS 5.0
Copyright 1998-2003 Boson Software, Inc.
Use the command help to get started

Press Enter to begin
C:>WINIPCFG
C:>ping 202.117.4.1
Pinging 202.117.4.1 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 202.117.4.1:
     Packets: Sent = 5, Received = 0, Lost = 5 (100% loss),
Approximate round trip times in milli-seconds:
     Minimum = 0ms, Maximum =  0ms, Average =  0ms

C:>ping 202.117.4.1
Pinging 202.117.4.1 with 32 bytes of data:

Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241

Ping statistics for 202.117.4.1:     Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
     Minimum = 50ms, Maximum =  60ms, Average =  55ms

C:>WINIPCFG
```

( ) Obtain an IP address automatically
(●) Specify an IP Address:

IP ADDRESS: 202.117.1.1 (202 117 1 1)
SUBNET MASK: 255.255.255.0 (255 255 255 0)
DEFAULT GATEWAY: 202.117.1.2 (202 117 1 2)

PC2：


Boson NetSim? v 5.31 -- Control Panel - [eStation 2]
File Modes Devices Tools Ordering Language Window Help
eRouters
eSwitches
eStations
Lab Navigator
NetMap
Remote Control
Configure your PC Address
Boson
OK
Cancel
Status

( ) Obtain an IP address automatically
(●) Specify an IP Address:

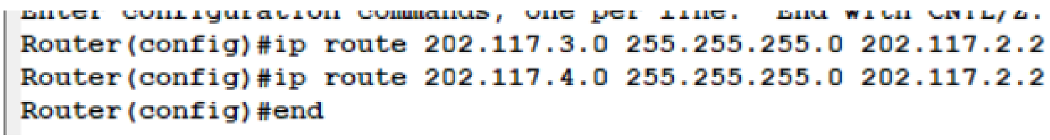
IP ADDRESS: 202.117.4.1 (202 117 4 1)
SUBNET MASK: 255.255.255.0 (255 255 255 0)
DEFAULT GATEWAY: 202.117.4.2 (202 117 4 2)

```
Boson BOSS 5.0
Copyright 1998-2003 Boson Software, Inc.
Use the command help to get started

Press Enter to begin
C:>WINIPCFG
C:>
C:>WINIPCFG
```

Bosonnetsim支持两种ip分配方式：RIP或OSPF。这里两种方式都进行测试。

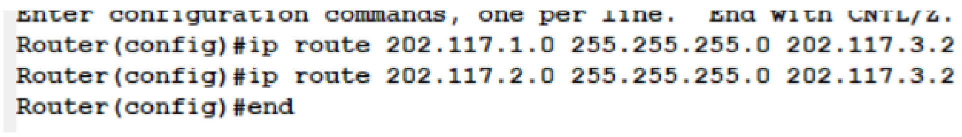
静态：

R1



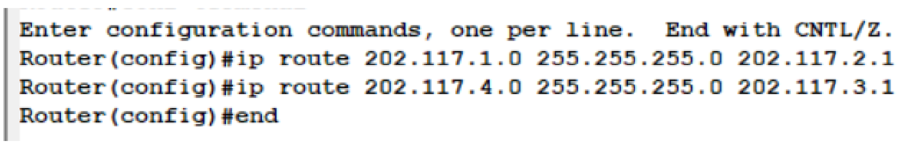
```
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip route 202.117.3.0 255.255.255.0 202.117.2.2
Router(config)#ip route 202.117.4.0 255.255.255.0 202.117.2.2
Router(config)#end
```

R2



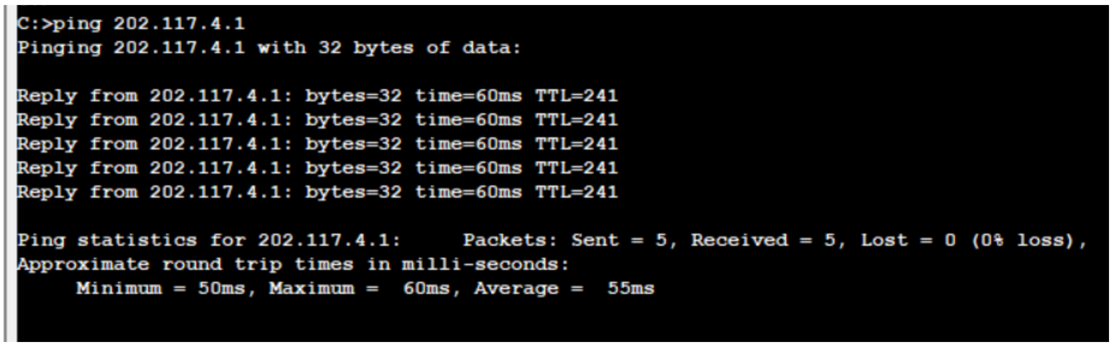
```
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip route 202.117.1.0 255.255.255.0 202.117.3.2
Router(config)#ip route 202.117.2.0 255.255.255.0 202.117.3.2
Router(config)#end
```

R3



```
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip route 202.117.1.0 255.255.255.0 202.117.2.1
Router(config)#ip route 202.117.4.0 255.255.255.0 202.117.3.1
Router(config)#end
```

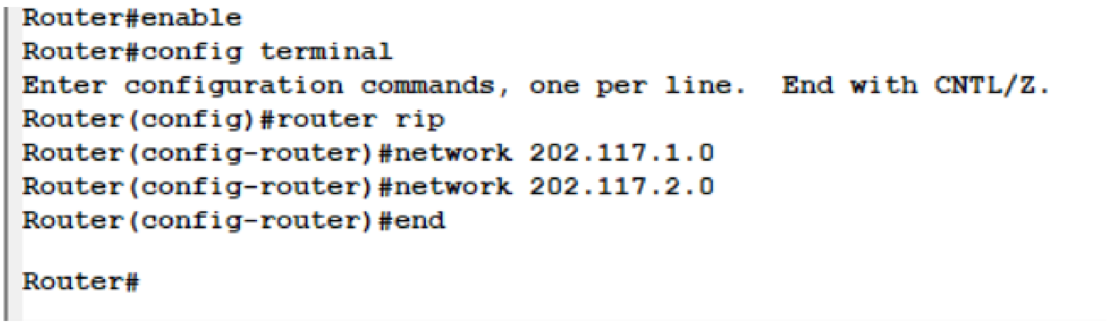
Ping结果：



```
C:>ping 202.117.4.1
Pinging 202.117.4.1 with 32 bytes of data:

Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241

Ping statistics for 202.117.4.1:     Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
     Minimum = 50ms, Maximum =  60ms, Average =  55ms
```

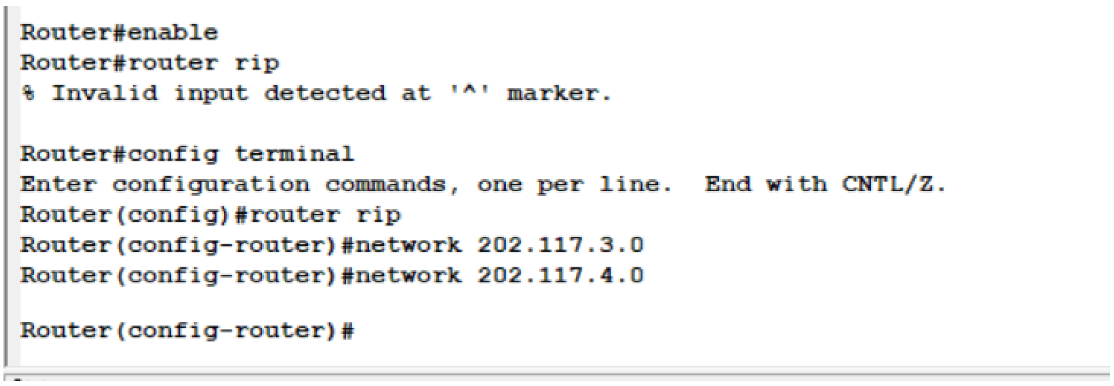
动态：

R1

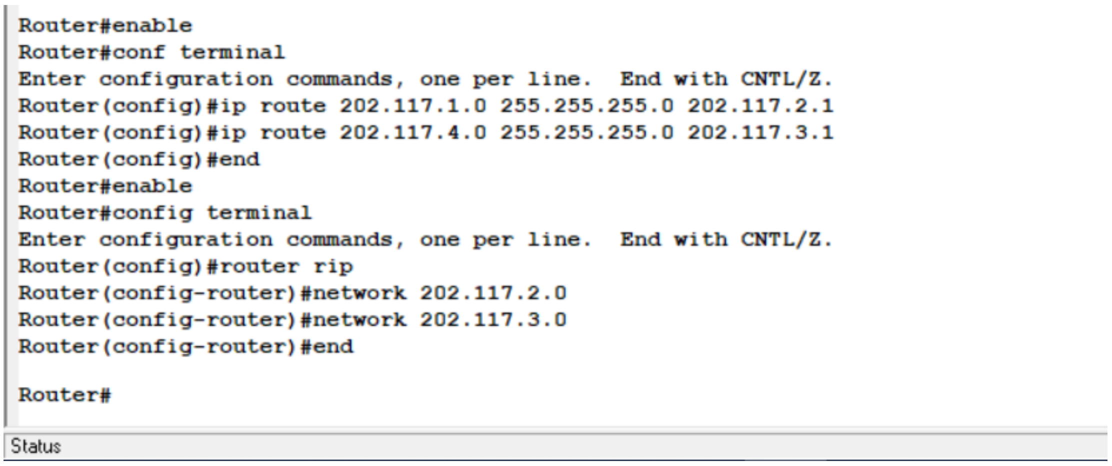


```
Router#enable
Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router rip
Router(config-router)#network 202.117.1.0
Router(config-router)#network 202.117.2.0
Router(config-router)#end

Router#
```

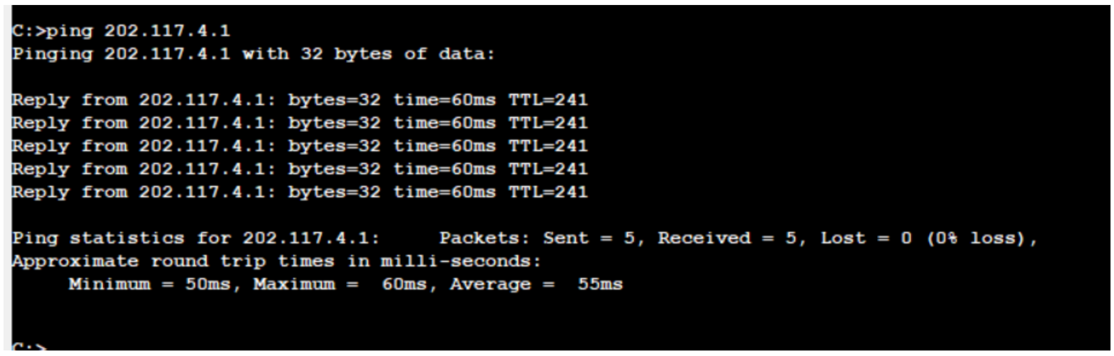
R2


```
Router#enable
Router#router rip
% Invalid input detected at '^' marker.

Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router rip
Router(config-router)#network 202.117.3.0
Router(config-router)#network 202.117.4.0

Router(config-router)#
```

R3


```
Router#enable
Router#conf terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#ip route 202.117.1.0 255.255.255.0 202.117.2.1
Router(config)#ip route 202.117.4.0 255.255.255.0 202.117.3.1
Router(config)#end
Router#enable
Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#router rip
Router(config-router)#network 202.117.2.0
Router(config-router)#network 202.117.3.0
Router(config-router)#end

Router#
```

Ping结果：


```
C:>ping 202.117.4.1
Pinging 202.117.4.1 with 32 bytes of data:

Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241
Reply from 202.117.4.1: bytes=32 time=60ms TTL=241

Ping statistics for 202.117.4.1:     Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
     Minimum = 50ms, Maximum =  60ms, Average =  55ms

C:>
```

五、总结

通过这次实验，我对于路由器配置了解更进了一步，之前只是在书上看了理论知识，实际操作下来有了更深的印象

实验二：　面向HTTP协议的抓包分析实验

一、实验目的

1. 利用ethereal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ethereal软件，进行报文截获
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ethereal 的报文截获，将截获命名为“http—学号”
4. 分析截获报文

三、实验要求

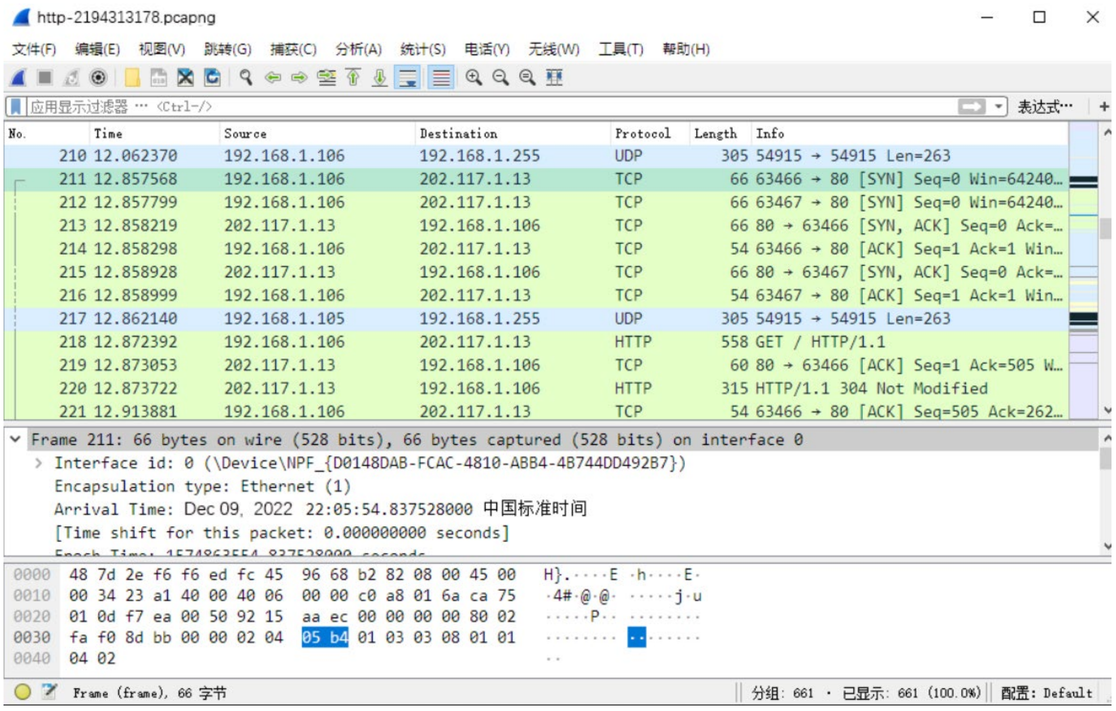
从截获的报文中选择HTTP请求报文（即get报文）和HTTP应答报文，并分析各字段的值；
综合分析截获的报文，概括HTTP协议的工作过程；
从截获报文中选择TCP建立连接和释放连接的报文，分析各个字段的值并概括HTTP协议的工作过程；

四、实验过程及结果

步骤一：启动软件抓包

Ethereal最新版为wireshark，这里报文截获使用wireshark软件
启动wireshark，关闭其他无关的软件避免额外的流量，减少干扰，开始抓包；启动浏览器访问 http://www.xjtu.edu.cn/

访问一段时间后，停止抓包，保存抓包结果 http—2194313178


http-2194313178.pcapng
文件(F) 编辑(E) 视图(V) 跳转(G) 捕获(C) 分析(A) 统计(S) 电话(Y) 无线(W) 工具(T) 帮助(H)
应用显示过滤器 … <Ctrl-/>
表达式…

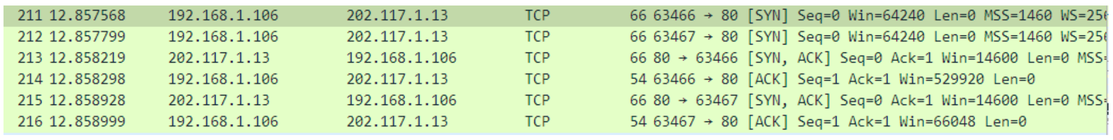
```
No.  Time        Source          Destination     Protocol  Length  Info
210 12.062370   192.168.1.106   192.168.1.255   UDP       305 54915 → 54915 Len=263
211 12.857568   192.168.1.106   202.117.1.13    TCP        66 63466 → 80 [SYN] Seq=0 Win=64240…
212 12.857799   192.168.1.106   202.117.1.13    TCP        66 63467 → 80 [SYN] Seq=0 Win=64240…
213 12.858219   202.117.1.13    192.168.1.106   TCP        66 80 → 63466 [SYN, ACK] Seq=0 Ack=…
214 12.858298   192.168.1.106   202.117.1.13    TCP        54 63466 → 80 [ACK] Seq=1 Ack=1 Win…
215 12.858928   202.117.1.13    192.168.1.106   TCP        66 80 → 63467 [SYN, ACK] Seq=0 Ack=…
216 12.858999   192.168.1.106   202.117.1.13    TCP        54 63467 → 80 [ACK] Seq=1 Ack=1 Win…
217 12.862140   192.168.1.105   192.168.1.255   UDP       305 54915 → 54915 Len=263
218 12.872392   192.168.1.106   202.117.1.13    HTTP      558 GET / HTTP/1.1
219 12.873053   202.117.1.13    192.168.1.106   TCP        60 80 → 63466 [ACK] Seq=1 Ack=505 W…
220 12.873722   202.117.1.13    192.168.1.106   HTTP      315 HTTP/1.1 304 Not Modified
221 12.913881   192.168.1.106   202.117.1.13    TCP        54 63466 → 80 [ACK] Seq=505 Ack=262…
```

```
Frame 211: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
  > Interface id: 0 (\Device\NPF_{D0148DAB-FCAC-4810-ABB4-4B744DD492B7})
    Encapsulation type: Ethernet (1)
    Arrival Time: Dec 09, 2022 22:05:54.837528000 中国标准时间
    [Time shift for this packet: 0.000000000 seconds]
```

```
0000  48 7d 2e f6 f6 ed fc 45  96 68 b2 82 08 00 45 00   H}.····E ·h····E·
0010  00 34 23 a1 40 00 40 06  00 00 c0 a8 01 6a ca 75   ·4#·@·@· ·····j·u
0020  01 0d f7 ea 00 50 92 15  aa ec 00 00 00 00 80 02   ·····P·· ········
0030  fa f0 8d bb 00 00 02 04  05 b4 01 03 03 08 01 01   ········ ········
0040  04 02                                              ··
```

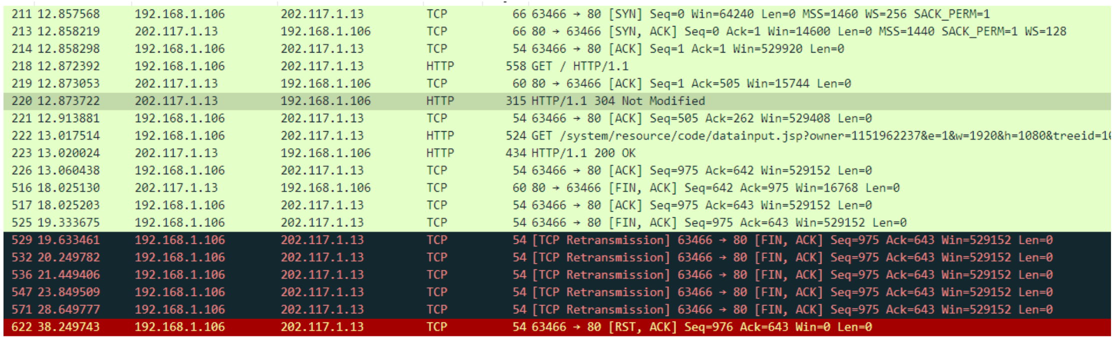
Frame (frame), 66 字节 | 分组: 661 · 已显示: 661 (100.0%) | 配置: Default

步骤二：根据ip找出tcp建立连接的包

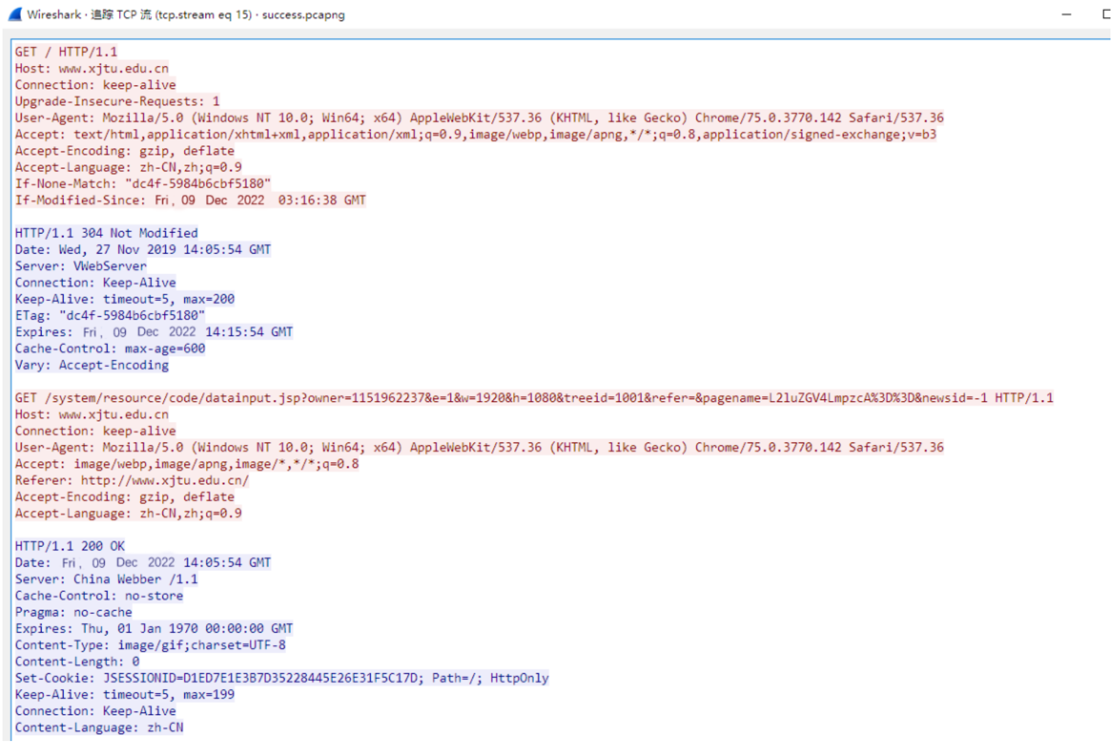


```
211 12.857568   192.168.1.106   202.117.1.13    TCP   66 63466 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=25
212 12.857799   192.168.1.106   202.117.1.13    TCP   66 63467 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=25
213 12.858219   202.117.1.13    192.168.1.106   TCP   66 80 → 63466 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS
214 12.858298   192.168.1.106   202.117.1.13    TCP   54 63466 → 80 [ACK] Seq=1 Ack=1 Win=529920 Len=0
215 12.858928   202.117.1.13    192.168.1.106   TCP   66 80 → 63467 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS
216 12.858999   192.168.1.106   202.117.1.13    TCP   54 63467 → 80 [ACK] Seq=1 Ack=1 Win=66048 Len=0
```

简要分析一下,在输入网页之后,chrome分别从两个不同端口发出两个tcp请求(如图前两个),每个请求有3次握手,一共6个包;至于chrome为什么发两个包,这是因为JSON Formatter Chrome这个扩展程序导致了请求两次的问题.
这个时候我们选择一个端口的包,右键点击跟踪tcp流,显示如下:

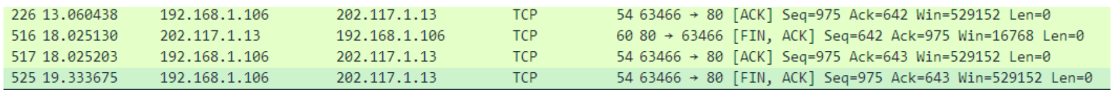


```
211 12.857568   192.168.1.106   202.117.1.13    TCP    66 63466 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
213 12.858219   202.117.1.13    192.168.1.106   TCP    66 80 → 63466 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1440 SACK_PERM=1 WS=128
214 12.858298   192.168.1.106   202.117.1.13    TCP    54 63466 → 80 [ACK] Seq=1 Ack=1 Win=529920 Len=0
218 12.872392   192.168.1.106   202.117.1.13    HTTP  558 GET / HTTP/1.1
219 12.873053   202.117.1.13    192.168.1.106   TCP    60 80 → 63466 [ACK] Seq=1 Ack=505 Win=15744 Len=0
220 12.873722   202.117.1.13    192.168.1.106   HTTP  315 HTTP/1.1 304 Not Modified
221 12.913881   192.168.1.106   202.117.1.13    TCP    54 63466 → 80 [ACK] Seq=505 Ack=262 Win=529408 Len=0
222 13.017514   192.168.1.106   202.117.1.13    HTTP  524 GET /system/resource/code/datainput.jsp?owner=1151962237&e=1&w=1920&h=1080&treeid=1
223 13.020024   202.117.1.13    192.168.1.106   HTTP  434 HTTP/1.1 200 OK
226 13.060438   192.168.1.106   202.117.1.13    TCP    54 63466 → 80 [ACK] Seq=975 Ack=642 Win=529152 Len=0
516 18.025130   202.117.1.13    192.168.1.106   TCP    60 80 → 63466 [FIN, ACK] Seq=642 Ack=975 Win=16768 Len=0
517 18.025203   192.168.1.106   202.117.1.13    TCP    54 63466 → 80 [ACK] Seq=975 Ack=643 Win=529152 Len=0
525 19.333675   192.168.1.106   202.117.1.13    TCP    54 63466 → 80 [FIN, ACK] Seq=975 Ack=643 Win=529152 Len=0
529 19.633461   192.168.1.106   202.117.1.13    TCP    54 [TCP Retransmission] 63466 → 80 [FIN, ACK] Seq=975 Ack=643 Win=529152 Len=0
532 20.249782   192.168.1.106   202.117.1.13    TCP    54 [TCP Retransmission] 63466 → 80 [FIN, ACK] Seq=975 Ack=643 Win=529152 Len=0
536 21.449406   192.168.1.106   202.117.1.13    TCP    54 [TCP Retransmission] 63466 → 80 [FIN, ACK] Seq=975 Ack=643 Win=529152 Len=0
547 23.849509   192.168.1.106   202.117.1.13    TCP    54 [TCP Retransmission] 63466 → 80 [FIN, ACK] Seq=975 Ack=643 Win=529152 Len=0
571 28.649777   192.168.1.106   202.117.1.13    TCP    54 [TCP Retransmission] 63466 → 80 [FIN, ACK] Seq=975 Ack=643 Win=529152 Len=0
622 38.249743   192.168.1.106   202.117.1.13    TCP    54 63466 → 80 [RST, ACK] Seq=976 Ack=643 Win=0 Len=0
```


Wireshark · 追踪 TCP 流 (tcp.stream eq 15) · success.pcapng

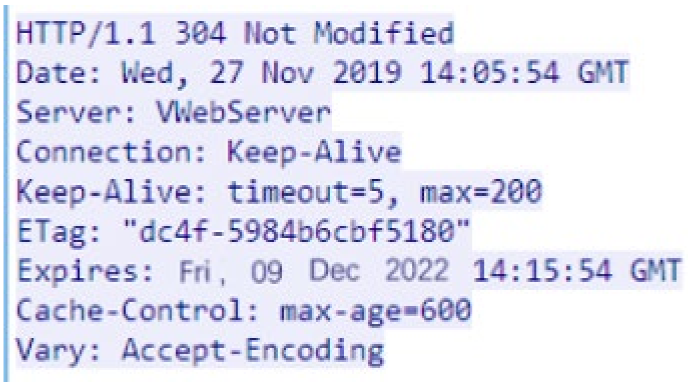
```
GET / HTTP/1.1
Host: www.xjtu.edu.cn
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
If-None-Match: "dc4f-5984b6cbf5180"
If-Modified-Since: Fri, 09 Dec 2022 03:16:38 GMT

HTTP/1.1 304 Not Modified
Date: Wed, 27 Nov 2019 14:05:54 GMT
Server: VWebServer
Connection: Keep-Alive
Keep-Alive: timeout=5, max=200
ETag: "dc4f-5984b6cbf5180"
Expires: Fri, 09 Dec 2022 14:15:54 GMT
Cache-Control: max-age=600
Vary: Accept-Encoding

GET /system/resource/code/datainput.jsp?owner=1151962237&e=1&w=1920&h=1080&treeid=1001&refer=&pagename=L2luZGV4LmpzcA%3D%3D&newsid=-1 HTTP/1.1
Host: www.xjtu.edu.cn
Connection: keep-alive
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept: image/webp,image/apng,image/*,*/*;q=0.8
Referer: http://www.xjtu.edu.cn/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9

HTTP/1.1 200 OK
Date: Fri, 09 Dec 2022 14:05:54 GMT
Server: China Webber /1.1
Cache-Control: no-store
Pragma: no-cache
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Content-Type: image/gif;charset=UTF-8
Content-Length: 0
Set-Cookie: JSESSIONID=D1ED7E1E3B7D35228445E26E31F5C17D; Path=/; HttpOnly
Keep-Alive: timeout=5, max=199
Connection: Keep-Alive
Content-Language: zh-CN
```

这时候我们可以清楚地看见3次握手4次挥手,并且注意到:4次握手中的先后顺序问题,是服务器先发送的两次挥手



```
226 13.060438    192.168.1.106    202.117.1.13     TCP    54 63466 → 80 [ACK] Seq=975 Ack=642 Win=529152 Len=0
516 18.025130    202.117.1.13     192.168.1.106    TCP    60 80 → 63466 [FIN, ACK] Seq=642 Ack=975 Win=16768 Len=0
517 18.025203    192.168.1.106    202.117.1.13     TCP    54 63466 → 80 [ACK] Seq=975 Ack=643 Win=529152 Len=0
525 19.333675    192.168.1.106    202.117.1.13     TCP    54 63466 → 80 [FIN, ACK] Seq=975 Ack=643 Win=529152 Len=0
```

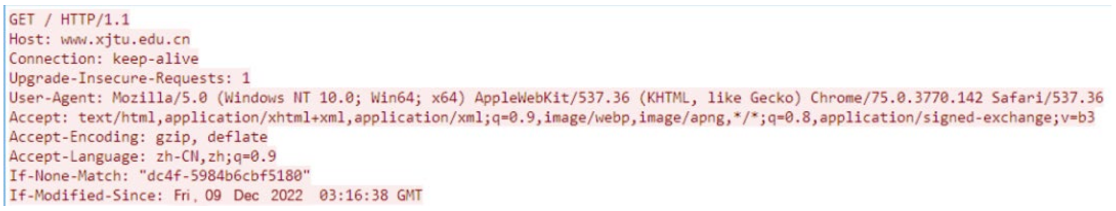
这里是因为这个http有最长连接时间5和命令次数200的限制,如下:



```
HTTP/1.1 304 Not Modified
Date: Wed, 27 Nov 2019 14:05:54 GMT
Server: VWebServer
Connection: Keep-Alive
Keep-Alive: timeout=5, max=200
ETag: "dc4f-5984b6cbf5180"
Expires: Fri, 09 Dec 2022 14:15:54 GMT
Cache-Control: max-age=600
Vary: Accept-Encoding
```

我打开网页之后等待的时间超过了5秒,所以是server先发断开连接的。

步骤三：http包分析



```
GET / HTTP/1.1
Host: www.xjtu.edu.cn
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
If-None-Match: "dc4f-5984b6cbf5180"
If-Modified-Since: Fri, 09 Dec 2022 03:16:38 GMT
```

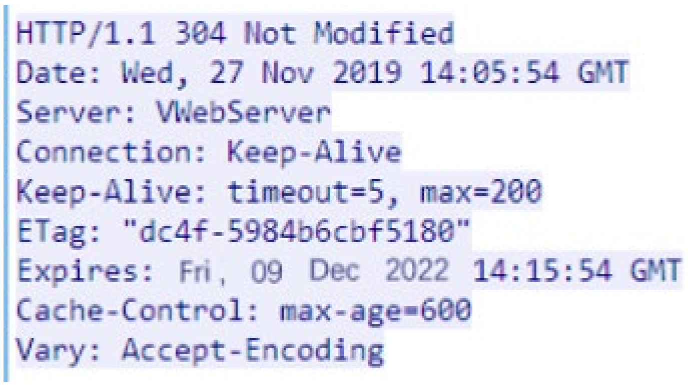
分析如下：

GET请求/节点用http/1.1协议；

主机为www.xjtu.edu.cn；

连接方式为keep-alive，就是连接不放，持续连接；
下面的Upgrade-Insecure-Requests：1是W3C 工作组考虑到了升级HTTPS的艰难，在2015 年 4 月份就出了一个Upgrade Insecure Requests的草案，他的作用就是让浏览器自动升级请求，可以将http升级为https；
User-Agent 是 指 客 户 端 使 用 的 是 Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36；
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8 ,application/signed-exchange;v=b3是指浏览器接受的响应格式，为test/html；
Accept-Encoding: gzip, deflate是指浏览器接受的压缩格式；
Accept-Language: zh-CN,zh;q=0.9是指浏览器接受的压缩格式；
If-None-Match: "dc4f-5984b6cbf5180"中If-None-Match是一个条件式请求首部对于GET和HEAD请求方法来说，当且仅当服务器上没有任何资源的ETag属性值与这个首部中列出的相匹配的时候，服务器端会才返回所请求的资源，响应码为200。对于其他方法来说，当且仅当最终确认没有已存在的资源的ETag属性值与这个首部中所列出的相匹配的时候，才会对请求进行相应的处理；
If-Modified-Since: Fri, 09 Dec 2022 03:16:38 GMT中If-Modified-Since是一个条件式请求首部，服务器只在所请求的资源在给定的日期时间之后对内容进行过修改的情况下才会将资源返回，状态码为200。如果请求的资源从那时起未经修改，那么返回一个不带有消息主体的304响应，而Last-Modified首部中会带有上次修改时间。不同于If-Unmodified-Since，If-Modified-Since只可以用在GET或HEAD请求中。

接下来分析http应答报文：



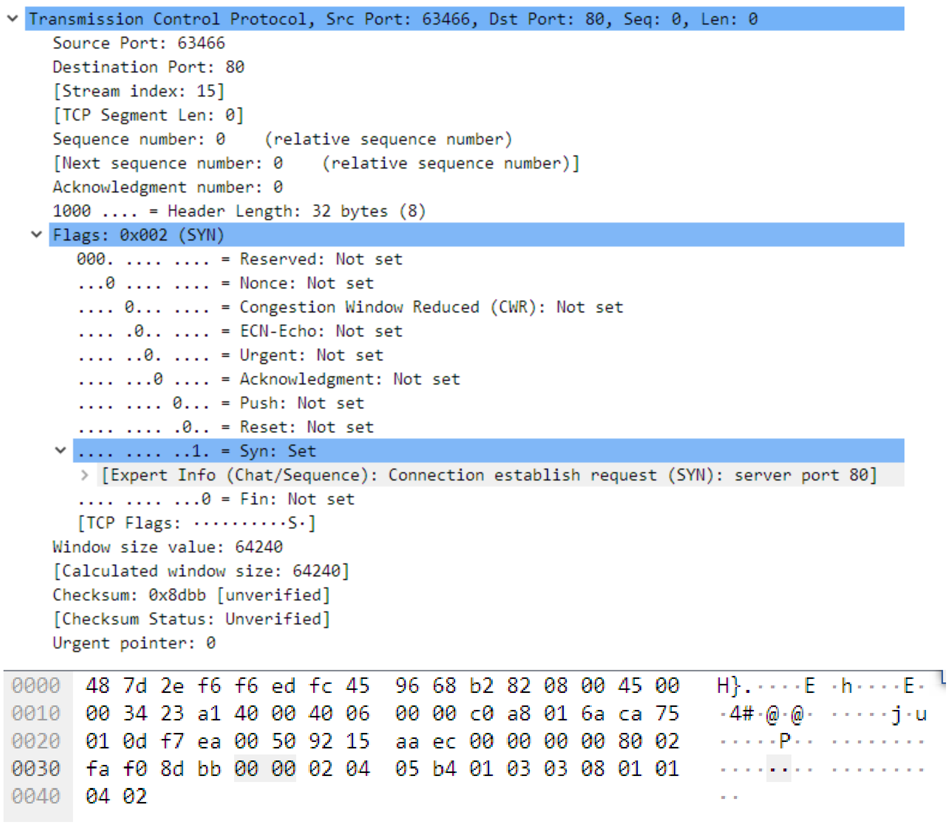
```
HTTP/1.1 304 Not Modified
Date: Wed, 27 Nov 2019 14:05:54 GMT
Server: VWebServer
Connection: Keep-Alive
Keep-Alive: timeout=5, max=200
ETag: "dc4f-5984b6cbf5180"
Expires: Fri, 09 Dec 2022 14:15:54 GMT
Cache-Control: max-age=600
Vary: Accept-Encoding
```

这里304返回值意思当客户端缓存了目标资源但不确定该缓存资源是否是最新版本的时候，就会发送一个条件请求，客户端会提供给服务器一个If-Modified-Since请求头，其值为服务器上次返回的Last-Modified响应头中的日期值，还会提供一个If-None-Match请求头，值为服务器上次返回的ETag响应头的值；服务器会读取到这两个请求头中的值，判断出客户端缓存的资源是否是最新的，如果是的话，服务器就会返回HTTP/304 Not Modified响应，但没有响应体。客户端收到304响应后，就会从缓存中读取对应的资源。
另一种情况是，如果服务器认为客户端缓存的资源已经过期了，那么服务器就会返HTTP/200 OK响应，响应体就是该资源当前最新的内容.客户端收到200响应后，就会用新的响应体覆盖掉旧的缓存资源。
只有在客户端缓存了对应资源且该资源的响应头中包含了Last-Modified或ETag的情况下，才可能发送条件请求。如果这两个头都不存在，则必须无条件(unconditionally)请求该资源，服务器也就必须返回完整的资源数据。

Etag由服务器端生成，客户端通过If-Match或者说If-None-Match这个条件判断请求来验证资源是否修改。常见的是使用If-None-Match.请求一个文件的流程可能如下：
====第一次请求===
1.客户端发起HTTP GET 请求一个文件；
2.服务器处理请求，返回文件内容和一堆Header，当然包括Etag(例如"2e681a-6-5d044840")(假设服务器支持Etag生成和已经开启了Etag).状态码200
====第二次请求===
1.客户端发起HTTP GET 请求一个文件，注意这个时候客户端同时发送一个If-None-Match头，这个头的内容就是第一次请求时服务器返回的Etag：2e681a-6-5d044840
2.服务器判断发送过来的Etag和计算出来的Etag匹配，因此If-None-Match为False，不返回200，返回304，客户端继续使用本地缓存。在完全匹配If-Modified-Since和If-None-Match即检查完修改时间和Etag之后，服务器才能返回304；
Expires: Fri, 27 Dec 2022 14:15:54 GMT
Cache-Control: max-age=600
Vary: Accept-Encoding
虽然上面的判断资源缓存是否存在以及是否更新节约资源下载时间,但是这已然发起了一次http请求。那么有别的方法可以免去这些http请求，就是服务器端的Expires和Cache-Control；其中Cache-Control可以控制缓存周期，而Expires是添加该资源过期的时间，用来和客户端时间对比，如果到期就要更新。

步骤四：分析tcp报文

这里以三次握手的第一次为例来具体分析

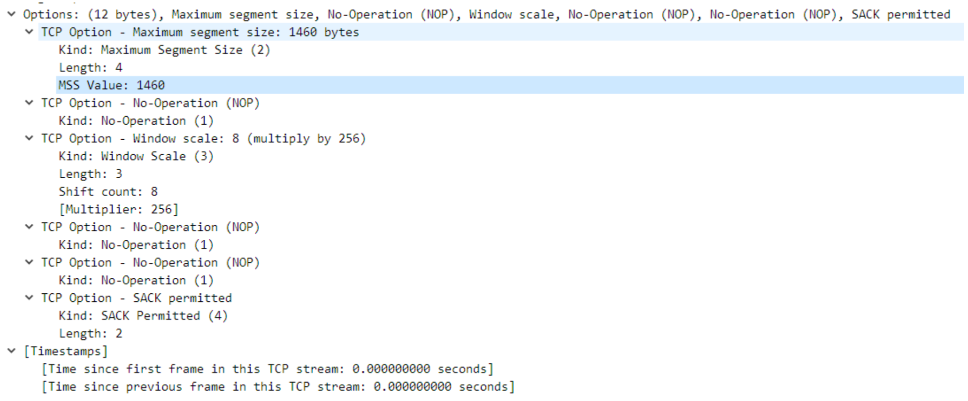


```
v Transmission Control Protocol, Src Port: 63466, Dst Port: 80, Seq: 0, Len: 0
    Source Port: 63466
    Destination Port: 80
    [Stream index: 15]
    [TCP Segment Len: 0]
    Sequence number: 0    (relative sequence number)
    [Next sequence number: 0    (relative sequence number)]
    Acknowledgment number: 0
    1000 .... = Header Length: 32 bytes (8)
  v 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
    v .... .... ..1. = Syn: Set
      > [Expert Info (Chat/Sequence): Connection establish request (SYN): server port 80]
      .... .... ...0 = Fin: Not set
      [TCP Flags: ··········S·]
    Window size value: 64240
    [Calculated window size: 64240]
    Checksum: 0x8dbb [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0

0000  48 7d 2e f6 f6 ed fc 45  96 68 b2 82 08 00 45 00   H}.····E ·h····E·
0010  00 34 23 a1 40 00 40 06  00 00 c0 a8 01 6a ca 75   ·4#·@·@· ·····j·u
0020  01 0d f7 ea 00 50 92 15  aa ec 00 00 00 00 80 02   ·····P·· ········
0030  fa f0 8d bb 00 00 02 04  05 b4 01 03 03 08 01 01   ········ ········
0040  04 02                                              ··
```

端口号为63466，是客户端的tcp端口，目的端口为80。这个tcp报文的序号为0x9215aaec，确认序号为0x0000。数据偏移为8，是指报文头为8x32B。保留字为000 000。

标志位之后SYN为1，其他都是0。接下来是窗口大小为0xfaf0，校验和为0x8dbb,紧急指针为0.

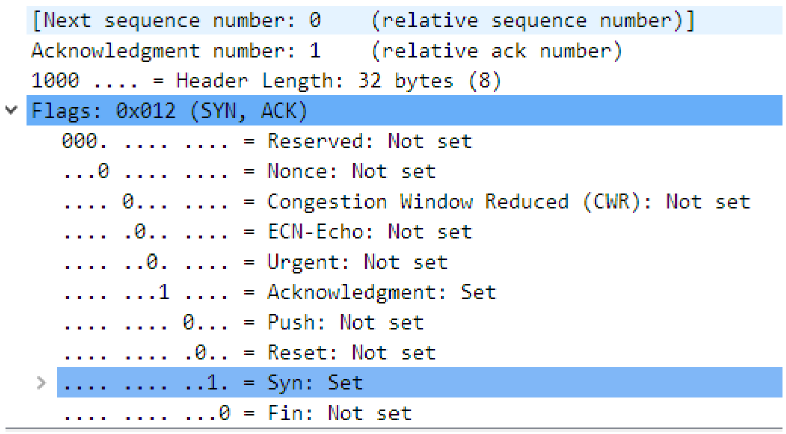
任选项如下，其中最大报文长度为1460B。



```
v Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
  v TCP Option - Maximum segment size: 1460 bytes
      Kind: Maximum Segment Size (2)
      Length: 4
      MSS Value: 1460
  v TCP Option - No-Operation (NOP)
      Kind: No-Operation (1)
  v TCP Option - Window scale: 8 (multiply by 256)
      Kind: Window Scale (3)
      Length: 3
      Shift count: 8
      [Multiplier: 256]
  v TCP Option - No-Operation (NOP)
      Kind: No-Operation (1)
  v TCP Option - No-Operation (NOP)
      Kind: No-Operation (1)
  v TCP Option - SACK permitted
      Kind: SACK Permitted (4)
      Length: 2
v [Timestamps]
    [Time since first frame in this TCP stream: 0.000000000 seconds]
    [Time since previous frame in this TCP stream: 0.000000000 seconds]
```

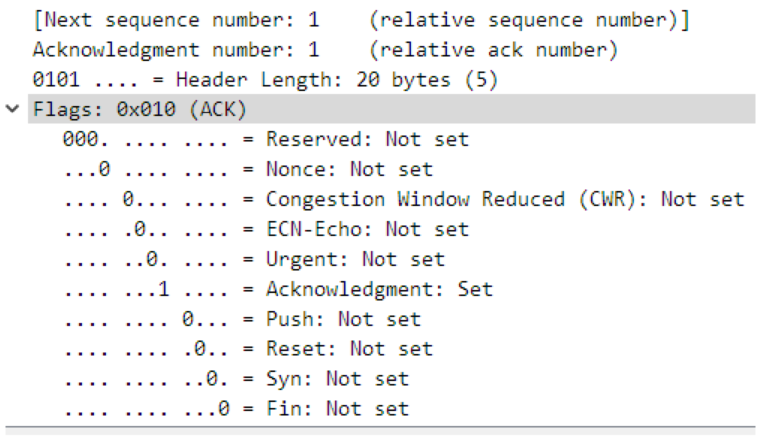
然后其他的握手和挥手不再赘述，它们基本上是滑动窗口、序号、确认序号以及标志位的不同。

第二次握手：



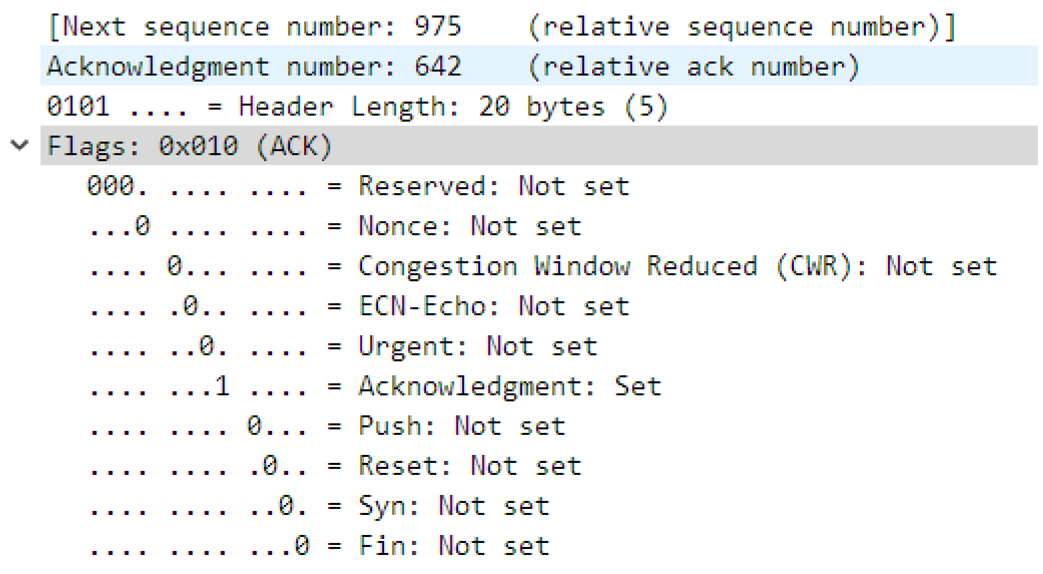
```
  [Next sequence number: 0    (relative sequence number)]
  Acknowledgment number: 1    (relative ack number)
  1000 .... = Header Length: 32 bytes (8)
v Flags: 0x012 (SYN, ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (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
```

第三次握手：



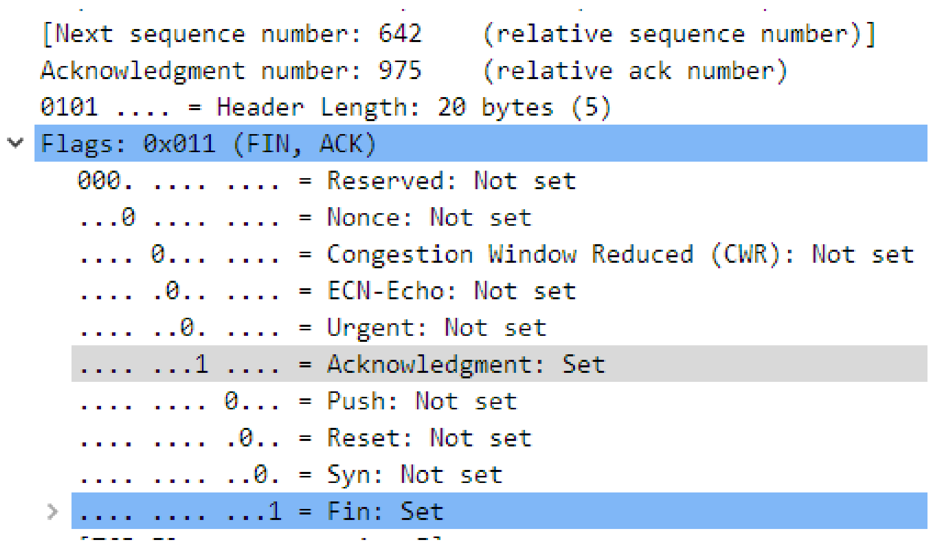
```
  [Next sequence number: 1    (relative sequence number)]
  Acknowledgment number: 1    (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
v Flags: 0x010 (ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (CWR): 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
```

第一次挥手：



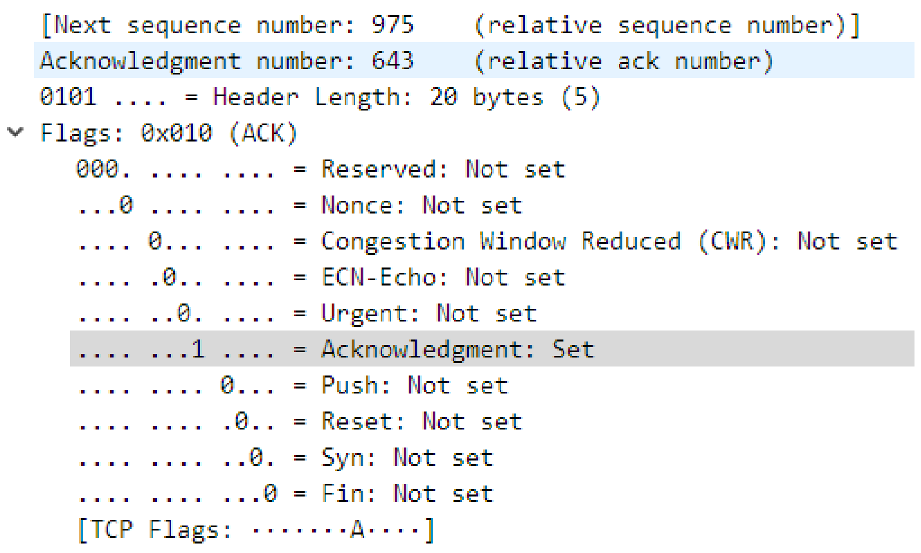
```
  [Next sequence number: 975    (relative sequence number)]
  Acknowledgment number: 642    (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
v Flags: 0x010 (ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (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
```

第二次挥手：



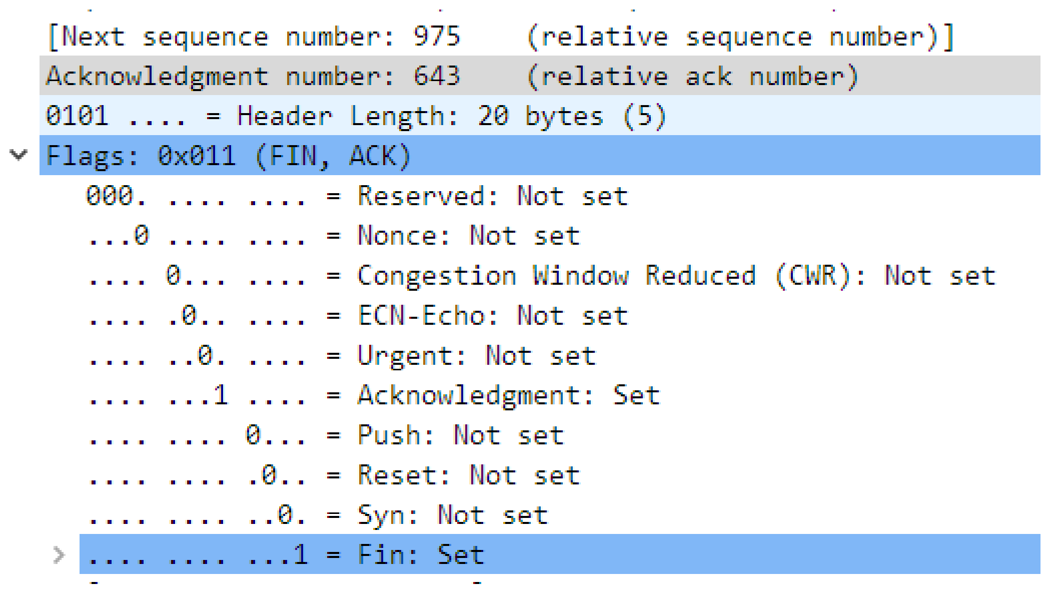
```
  [Next sequence number: 642    (relative sequence number)]
  Acknowledgment number: 975    (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
v Flags: 0x011 (FIN, ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (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
```

第三次挥手：



```
  [Next sequence number: 975    (relative sequence number)]
  Acknowledgment number: 643    (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
v 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····]
```

第四次挥手：



```
[Next sequence number: 975    (relative sequence number)]
Acknowledgment number: 643    (relative ack number)
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
```

五、总结

Tcp的工作过程：

三次握手建立连接。主机1向主机2发送syn=1的tcp报文，主机2收到之后向1发送syn=1，ack=1的报文来确认，主机1收到之后再向2发送syn=0，ack=1的报文进行确认。

http工作过程：

1.对www.baidu.com这个网址进行DNS域名解析，得到对应的IP地址
2.根据这个IP，找到对应的服务器，发起TCP的三次握手
3.建立TCP连接后发起HTTP请求
4.服务器响应HTTP请求，浏览器得到html代码
5.浏览器解析html代码，并请求html代码中的资源（如js、css图片等）（先得到html代码，才能去找这些资源）
6.浏览器对页面进行渲染呈现给用户

这次实验让我对于tcp连接的三次握手四次握手有了更加深刻的了解。我也开始对http有了更加具体的印象而非抽象的概念。Tcp工作在传输层而http工作在应用层，所以两者之间有着SAP的联系使得二者某种程度上是密不可分的。