西安交通大学

计算机网络原理专题实验报告

班级：	计算机72
学号：	2174410455
姓名：	徐子越

2019年11月28日

实验一:　http 和 tcp 抓包实验,基于 wireshark

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。

实验内容:

1>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；

2>综合分析截获的报文，概括 HTTP 协议的工作过程；

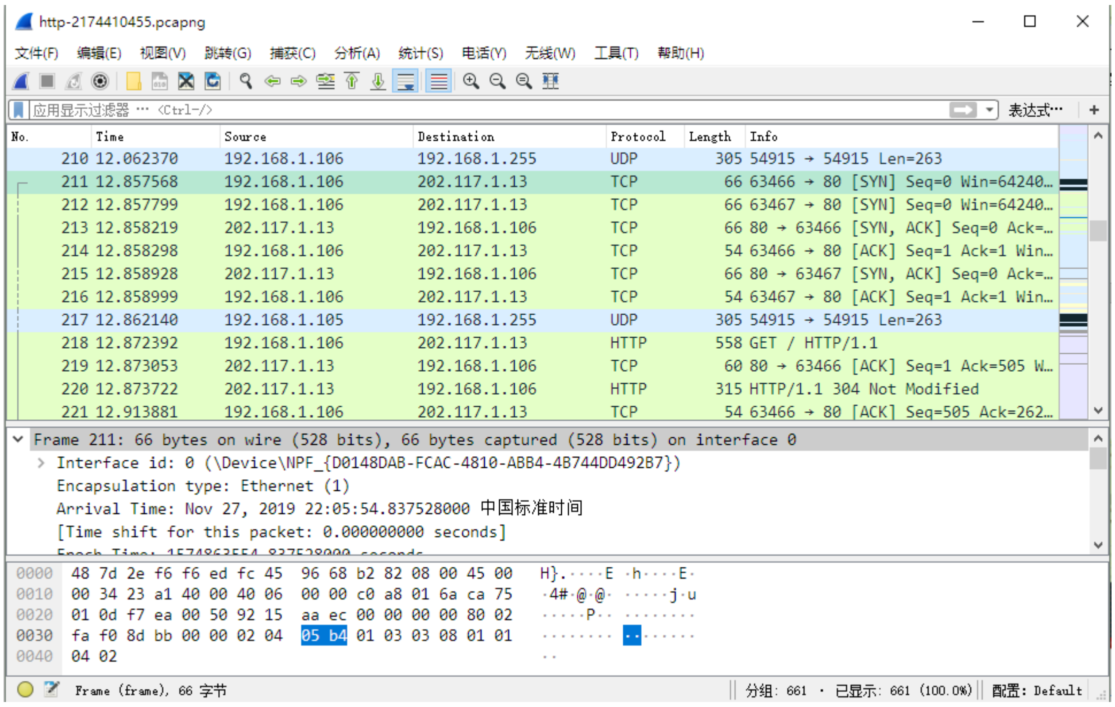
3>从截获报文中选择 TCP 建立连接和释放连接的报文，分析各个字段的值并概括 HTTP 协议的工作过程;

实验过程:

步骤一:下载 wireshark

步骤二:chrome 浏览器打开 www.xjtu.edu.cn,　wireshark 开始抓包.

西安交通大学
www.xjtu.edu.cn
设为首页 | English Version
西安交通大学
XI'AN JIAOTONG UNIVERSITY
在校生 | 教职工 | 校友 | 考生与访客 | 合作者 | 捐赠
首页 交大概况 院系设置 组织机构 本科书院 医学教育 科学研究 师资队伍 人才培养 国际交流 图书档案 校友服务 校园生活 信息服务
Universiteit Leiden Day
莱顿大学日
2019年11月29日
西安交通大学科学馆
传承与荣光 招生与培养 科研与人才 发展与展望 信息与服务


http-2174410455.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…

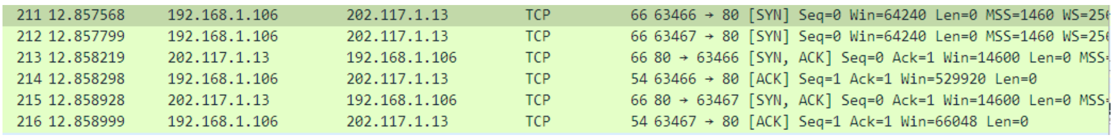
```
No.  Time        Source         Destination    Protocol  Length  Info
 210 12.062370   192.168.1.106  192.168.1.255  UDP       305 54915 → 54915 Len=263
 211 12.857568   192.168.1.106  202.117.1.13   TCP        66 63466 → 80 [SYN] Seq=0 Win=64240…
 212 12.857799   192.168.1.106  202.117.1.13   TCP        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: Nov 27, 2019 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
```

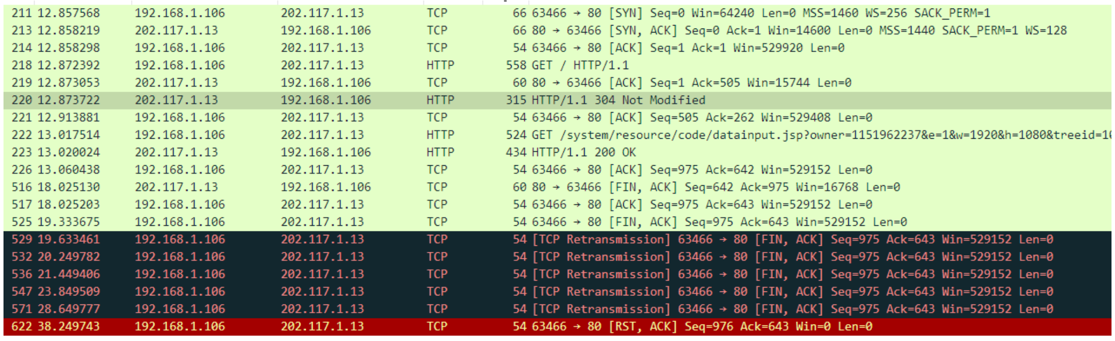
步骤三:停止抓包,根据 ip 地址找出 tcp 建立连接的包



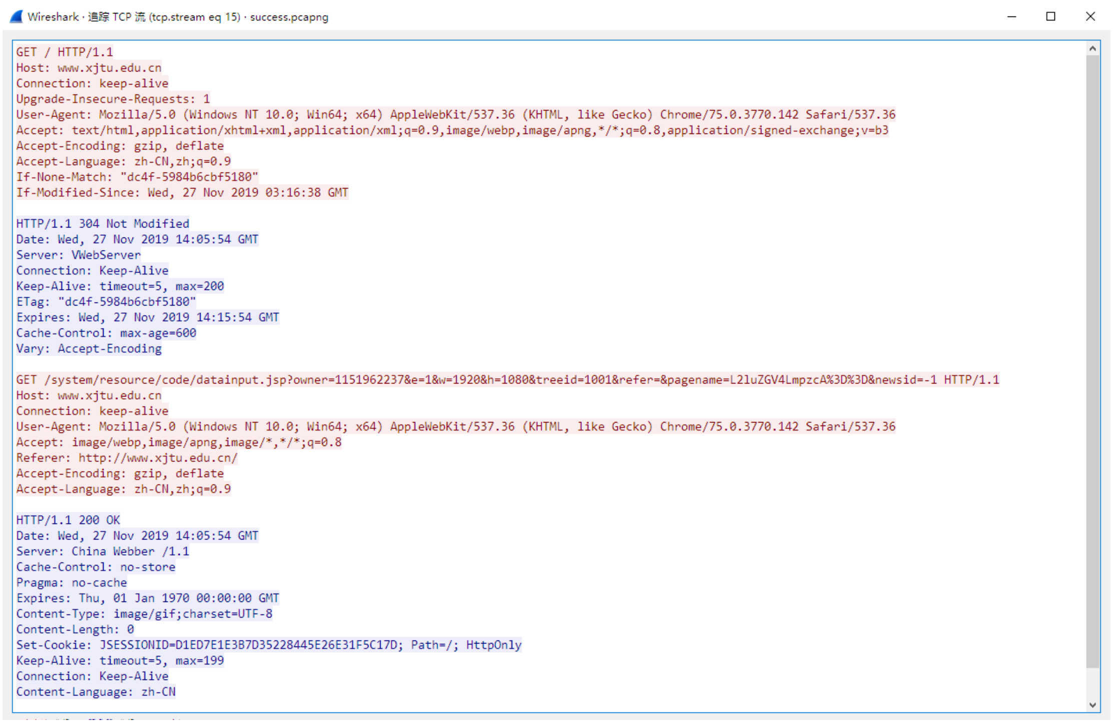
```
211 12.857568   192.168.1.106   202.117.1.13    TCP   66 63466 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=25
212 12.857799   192.168.1.106   202.117.1.13    TCP   66 63467 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=25
213 12.858219   202.117.1.13    192.168.1.106   TCP   66 80 → 63466 [SYN, ACK] 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
```



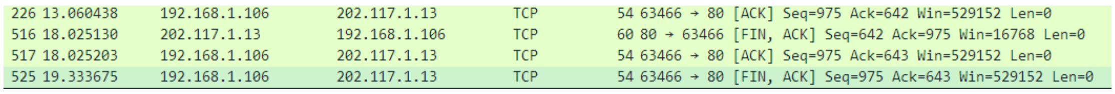
```
GET / HTTP/1.1
Host: www.xjtu.edu.cn
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/75.0.3770.142 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
If-None-Match: "dc4f-5984b6cbf5180"
If-Modified-Since: Wed, 27 Nov 2019 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: Wed, 27 Nov 2019 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: Wed, 27 Nov 2019 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: Wed, 27 Nov 2019 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.

步骤四:　　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: Wed, 27 Nov 2019 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；

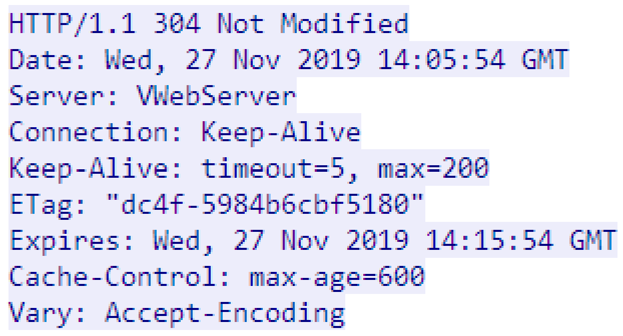
Accept-Encoding: gzip, deflate 是指浏览器接受的压缩格式；

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: Wed, 27 Nov 2019 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: Wed, 27 Nov 2019 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，那么服务器就会返回 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。

之后的 Date，Server，Connection，Keep-Alive 不再赘述。

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: Wed, 27 Nov 2019 14:15:54 GMT

Cache-Control: max-age=600

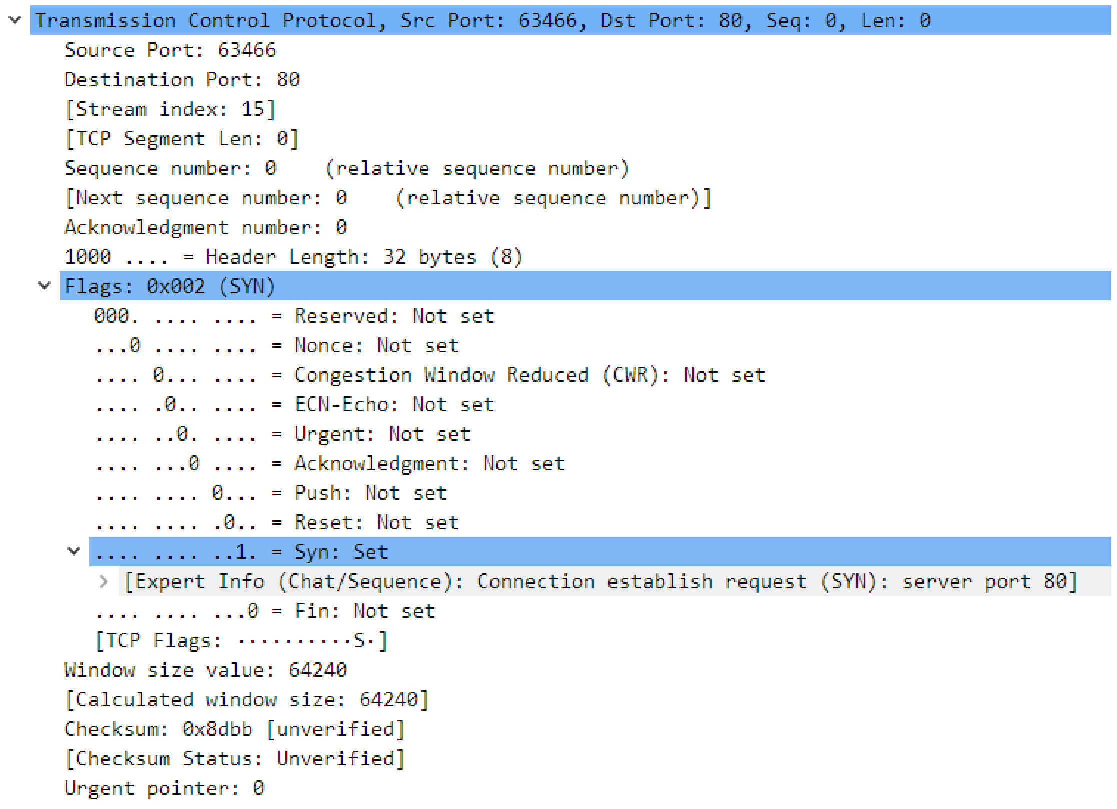
Vary: Accept-Encoding

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。

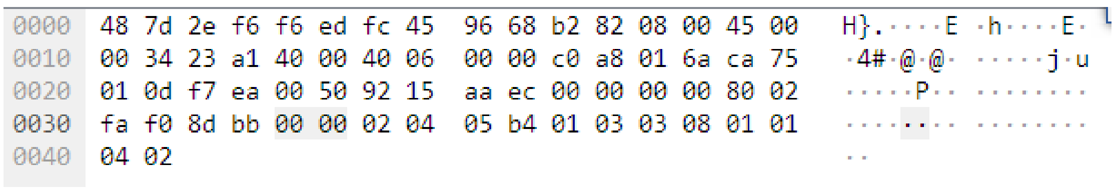
后面的相同，不在赘述。

步骤五：　　分析 tcp 报文

这里以三次握手的第一次为例来具体分析

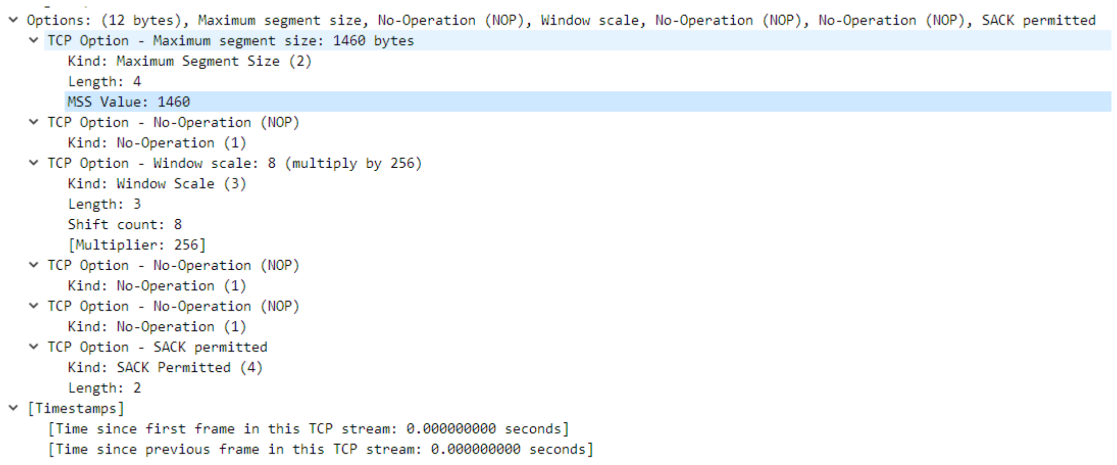


```
v Transmission Control Protocol, Src Port: 63466, Dst Port: 80, Seq: 0, Len: 0
    Source Port: 63466
    Destination Port: 80
    [Stream index: 15]
    [TCP Segment Len: 0]
    Sequence number: 0    (relative sequence number)
    [Next sequence number: 0    (relative sequence number)]
    Acknowledgment 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                                              ··
```

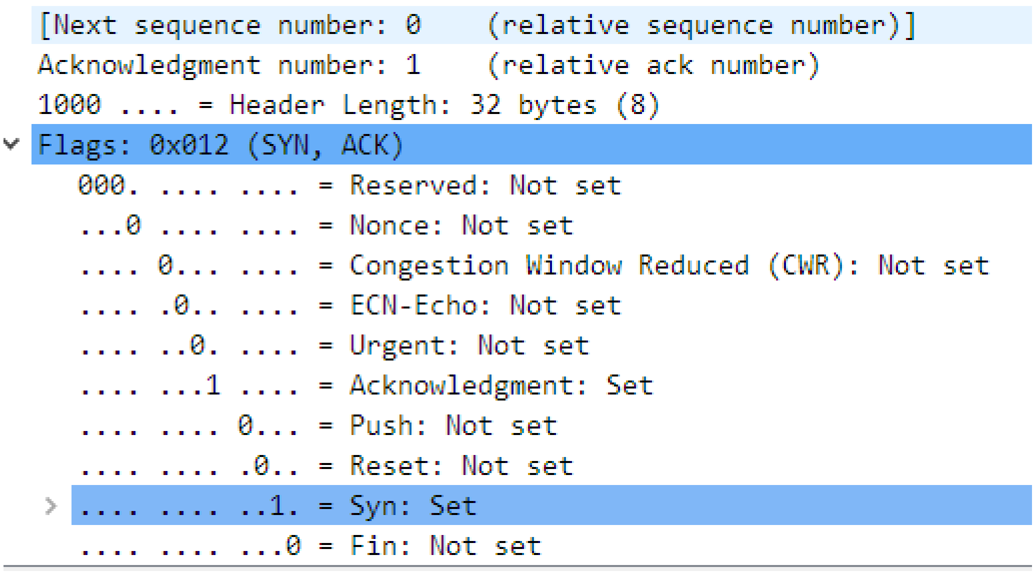
端口号为 63466，是客户端的 tcp 端口，目的端口为 80。这个 tcp 报文的序号为 0x9215aaec，确认序号为 0x0000。数据偏移为 8，是指报文头为 8x32B。保留字为 000 000。标志位之后 SYN 为 1，其他都是 0。接下来是窗口大小为 0xfaf0，校验和为 0x8dbb,紧急指针为 0.

任选项如下，其中最大报文长度为 1460B。

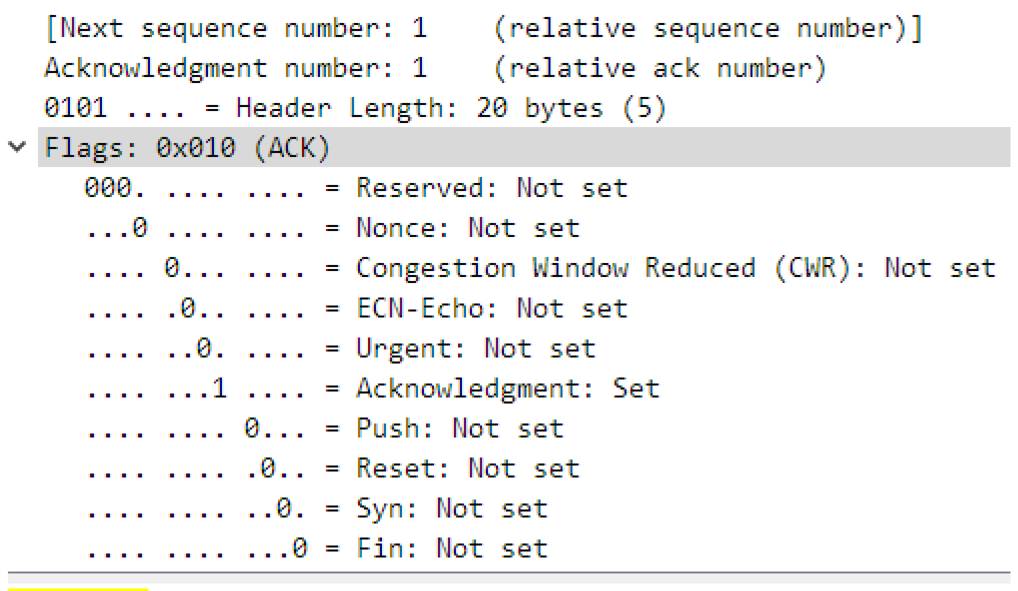

```
v Options: (12 bytes), Maximum segment size, No-Operation (NOP), Window scale, No-Operation (NOP), No-Operation (NOP), SACK permitted
  v TCP Option - Maximum segment size: 1460 bytes
      Kind: Maximum Segment Size (2)
      Length: 4
      MSS Value: 1460
  v TCP Option - No-Operation (NOP)
      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

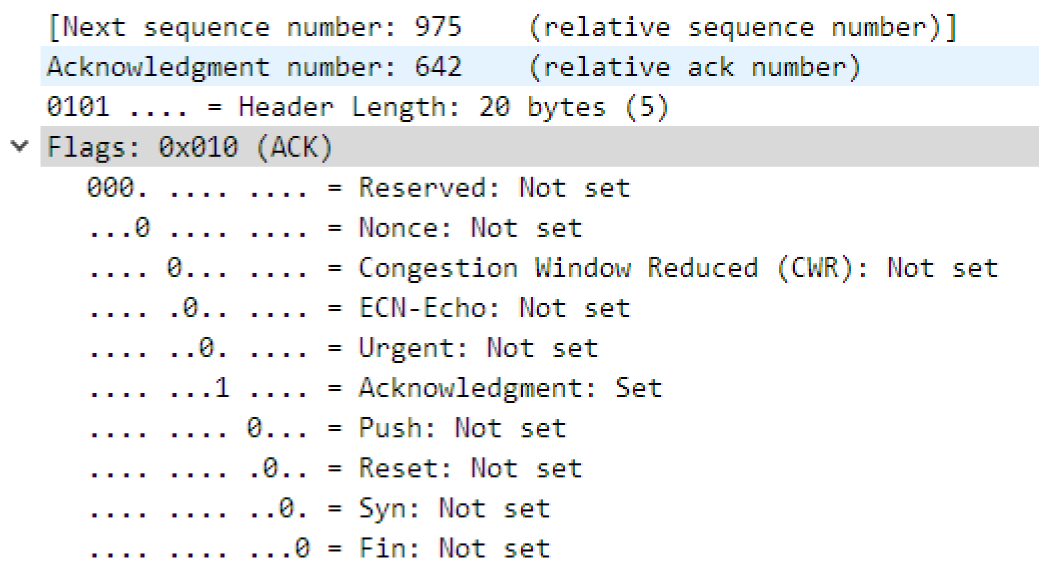

```
  [Next sequence number: 0    (relative sequence number)]
  Acknowledgment number: 1    (relative ack number)
  1000 .... = Header Length: 32 bytes (8)
v Flags: 0x012 (SYN, ACK)
     000. .... .... = Reserved: Not set
     ...0 .... .... = Nonce: Not set
     .... 0... .... = Congestion Window Reduced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



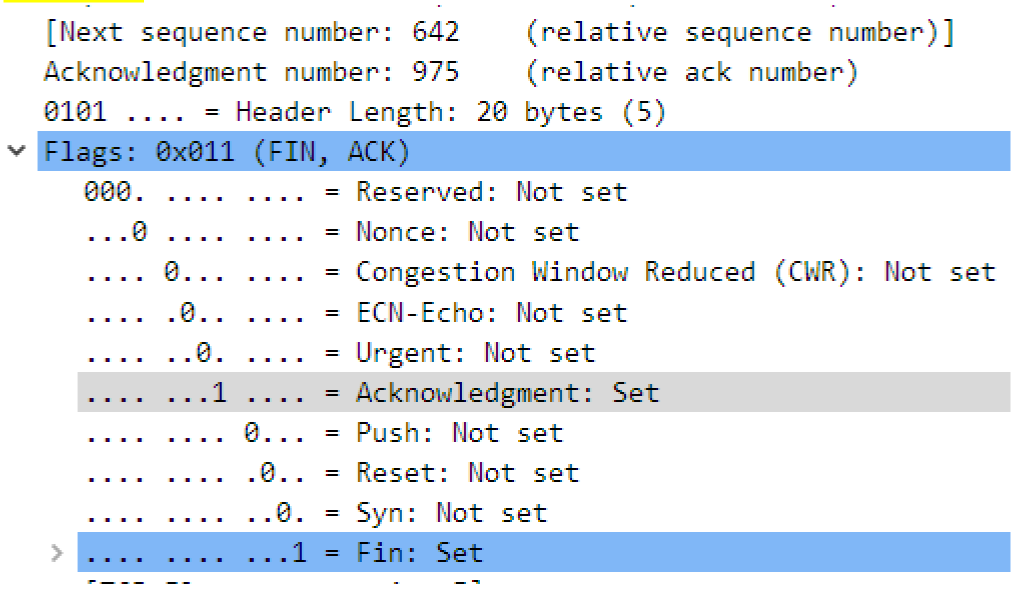
```
  [Next sequence number: 1    (relative sequence number)]
  Acknowledgment number: 1    (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
v Flags: 0x010 (ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (CWR):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



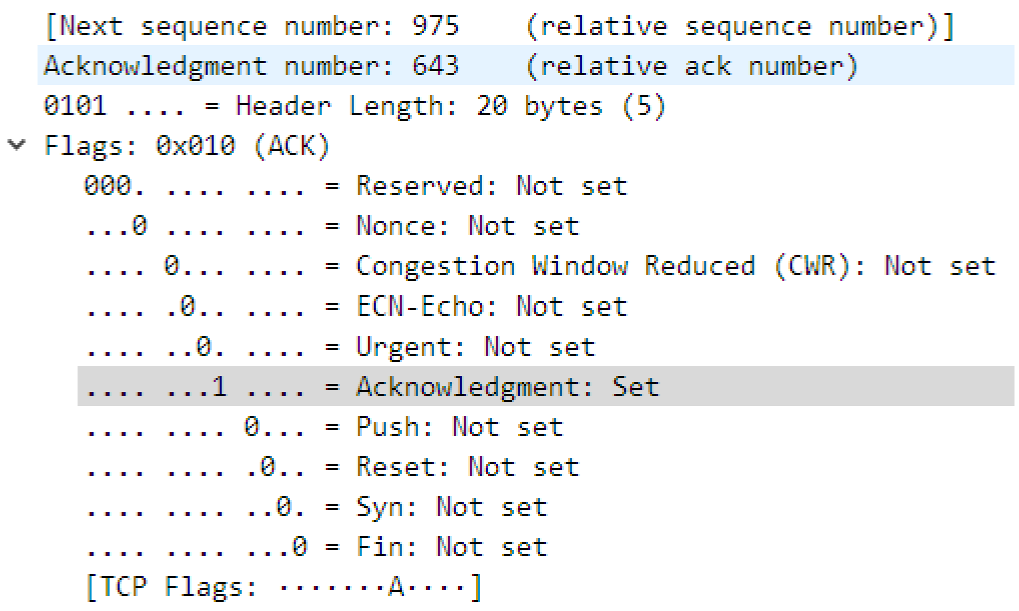
```
  [Next sequence number: 975    (relative sequence number)]
  Acknowledgment number: 642    (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
v Flags: 0x010 (ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced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



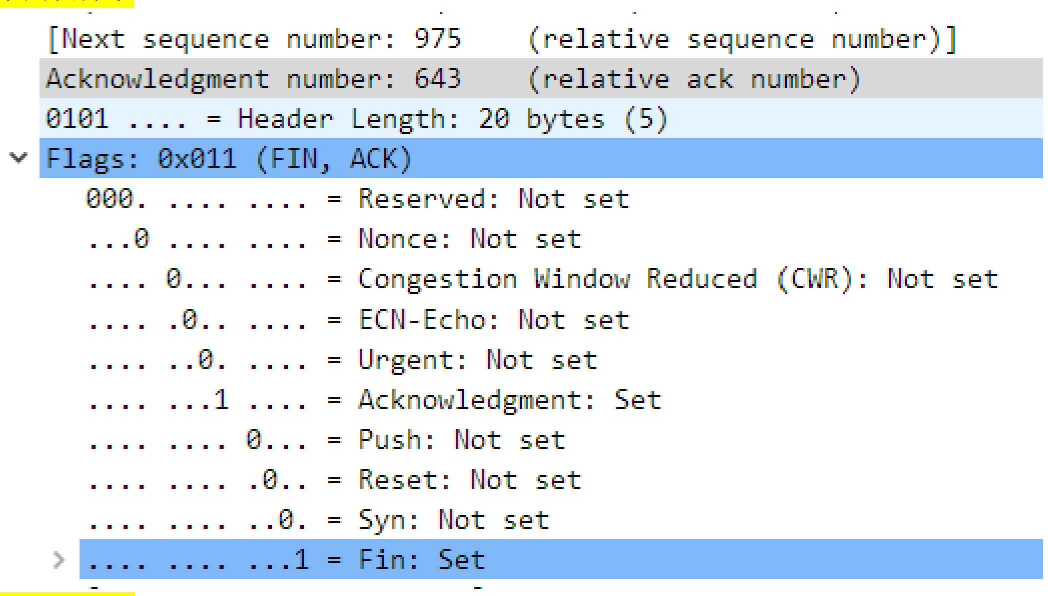
```
  [Next sequence number: 642    (relative sequence number)]
  Acknowledgment number: 975    (relative ack number)
  0101 .... = Header Length: 20 bytes (5)
v Flags: 0x011 (FIN, ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced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

实验总结:

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

实验二：　路由器基本配置实验,基于 Boson NetSim

实验目的:

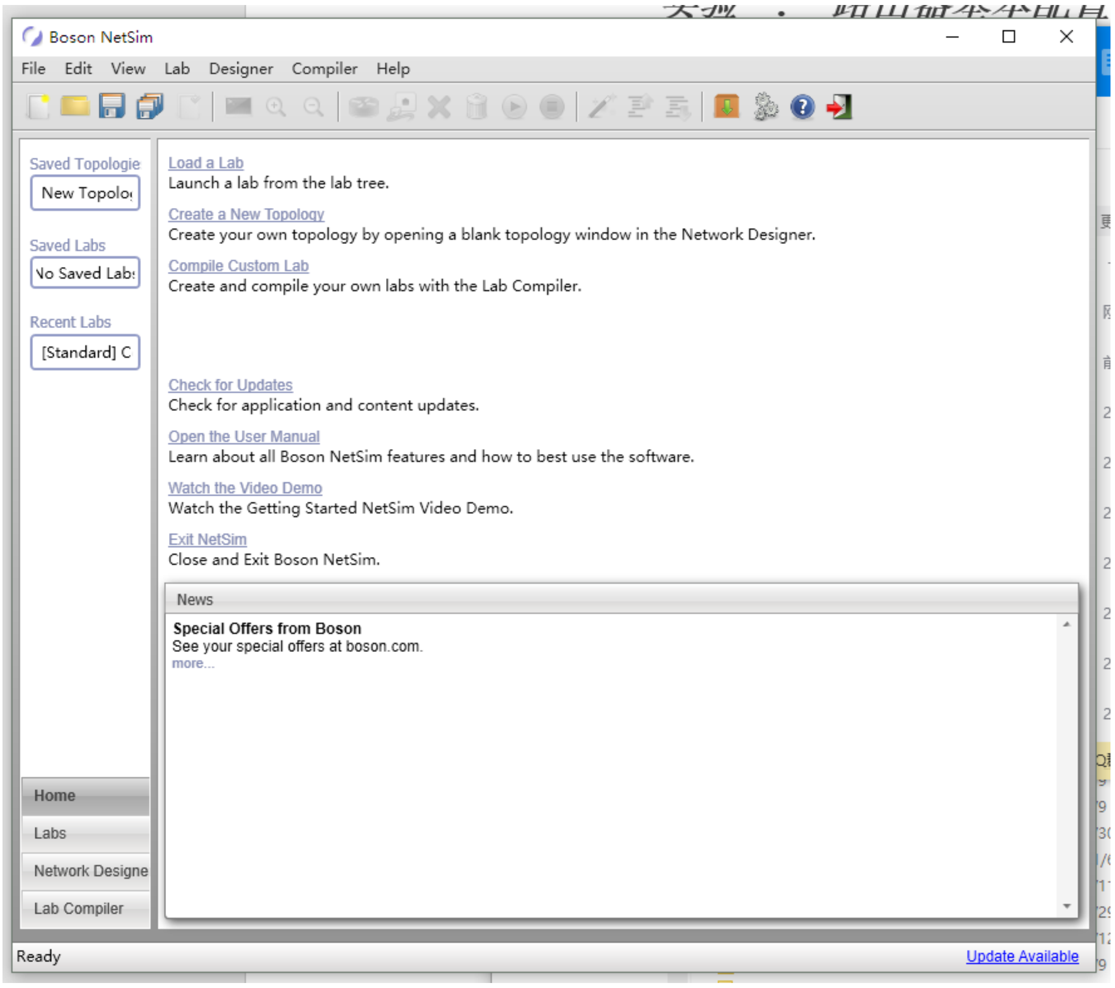
1>掌握路由器的基本知识
2>掌握路由器端口的配置
3>掌握路由协议的基本配置
4>熟悉使用 Boson Netsim 模拟器

实验内容:

1>使用 IOS 命令配置路由器
2>掌握握静态路由和动态路由（RIP、OSPF）的配置方法
3>本实验要求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；
4>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。
5>实验配置完成后，两台主机要能够相互 ping 通

实验过程:

步骤一　下载安装 NetSim

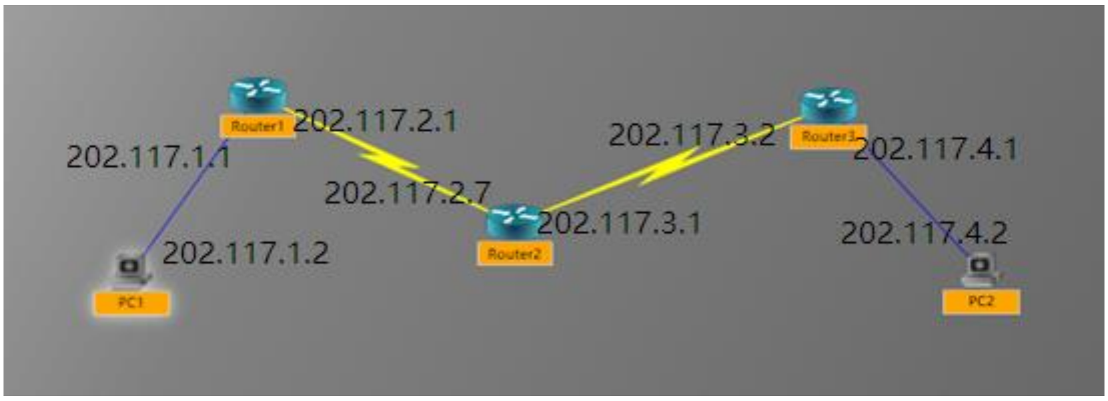

Boson NetSim
File Edit View Lab Designer Compiler Help

Saved Topologie
New Topolo
Saved Labs
No Saved Labs
Recent Labs
[Standard] C

Load a Lab
Launch a lab from the lab tree.
Create a New Topology
Create your own topology by opening a blank topology window in the Network Designer.
Compile Custom Lab
Create and compile your own labs with the Lab Compiler.
Check for Updates
Check for application and content updates.
Open the User Manual
Learn about all Boson NetSim features and how to best use the software.
Watch the Video Demo
Watch the Getting Started NetSim Video Demo.
Exit NetSim
Close and Exit Boson NetSim.

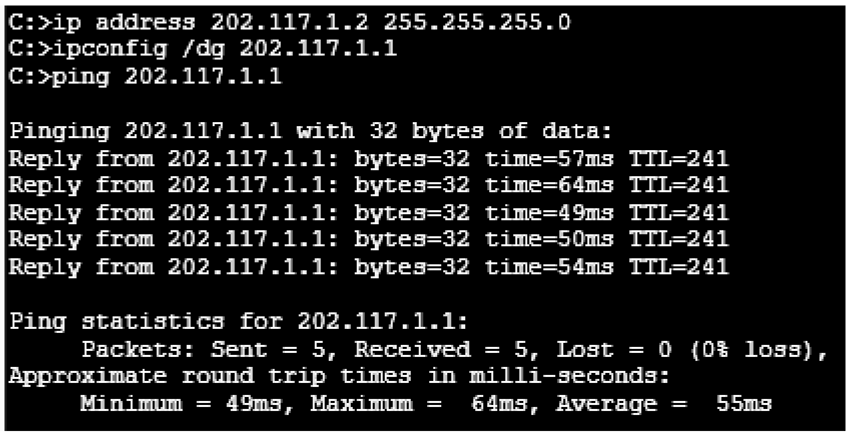
News
Special Offers from Boson
See your special offers at boson.com.
more...

步骤二　设计拓扑图,如下



步骤三　主机和路由器配置如下:

PC1:

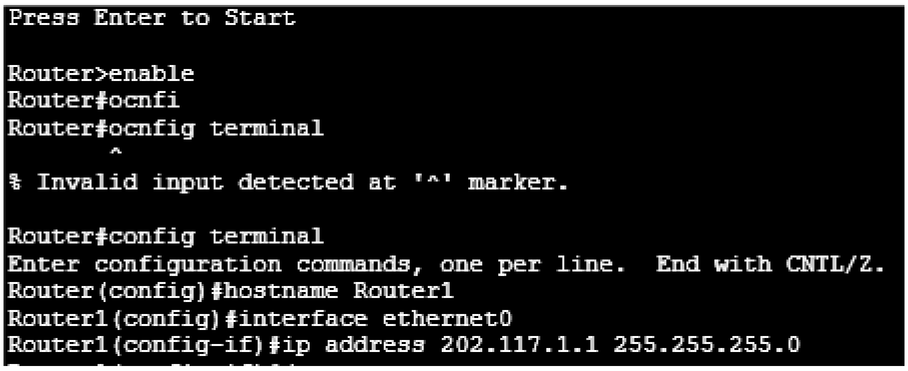


```
C:>ip address 202.117.1.2 255.255.255.0
C:>ipconfig /dg 202.117.1.1
C:>ping 202.117.1.1

Pinging 202.117.1.1 with 32 bytes of data:
Reply from 202.117.1.1: bytes=32 time=57ms TTL=241
Reply from 202.117.1.1: bytes=32 time=64ms TTL=241
Reply from 202.117.1.1: bytes=32 time=49ms TTL=241
Reply from 202.117.1.1: bytes=32 time=50ms TTL=241
Reply from 202.117.1.1: bytes=32 time=54ms TTL=241

Ping statistics for 202.117.1.1:
     Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
     Minimum = 49ms, Maximum =  64ms, Average =  55ms
```

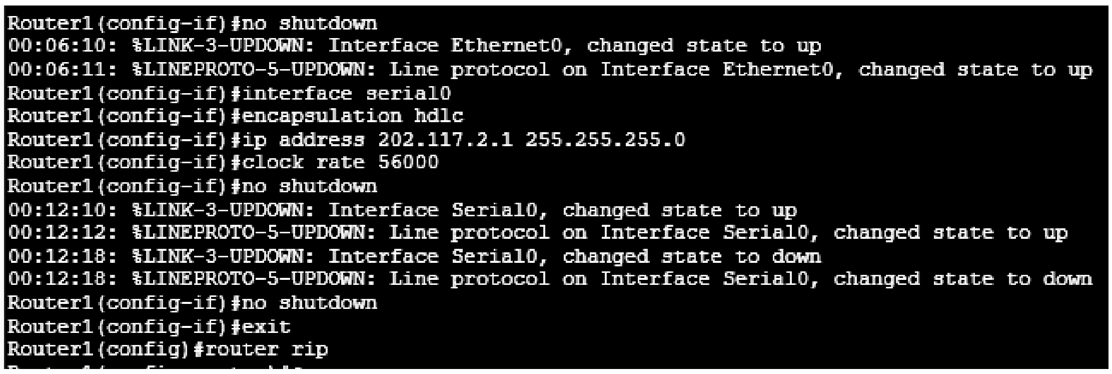
Router1:



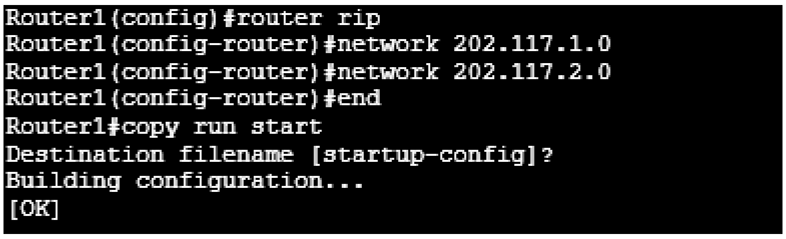
```
Press Enter to Start

Router>enable
Router#ocnfi
Router#ocnfig terminal
       ^
% Invalid input detected at '^' marker.

Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname Router1
Router1(config)#interface ethernet0
Router1(config-if)#ip address 202.117.1.1 255.255.255.0
```

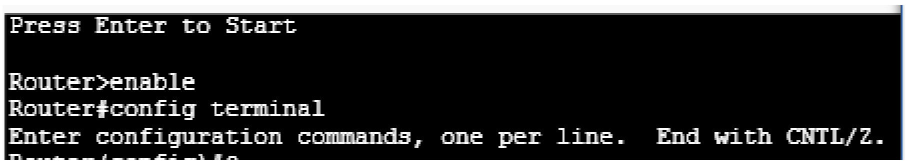


```
Router1(config-if)#no shutdown
00:06:10: %LINK-3-UPDOWN: Interface Ethernet0, changed state to up
00:06:11: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0, changed state to up
Router1(config-if)#interface serial0
Router1(config-if)#encapsulation hdlc
Router1(config-if)#ip address 202.117.2.1 255.255.255.0
Router1(config-if)#clock rate 56000
Router1(config-if)#no shutdown
00:12:10: %LINK-3-UPDOWN: Interface Serial0, changed state to up
00:12:12: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to up
00:12:18: %LINK-3-UPDOWN: Interface Serial0, changed state to down
00:12:18: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to down
Router1(config-if)#no shutdown
Router1(config-if)#exit
Router1(config)#router rip
```



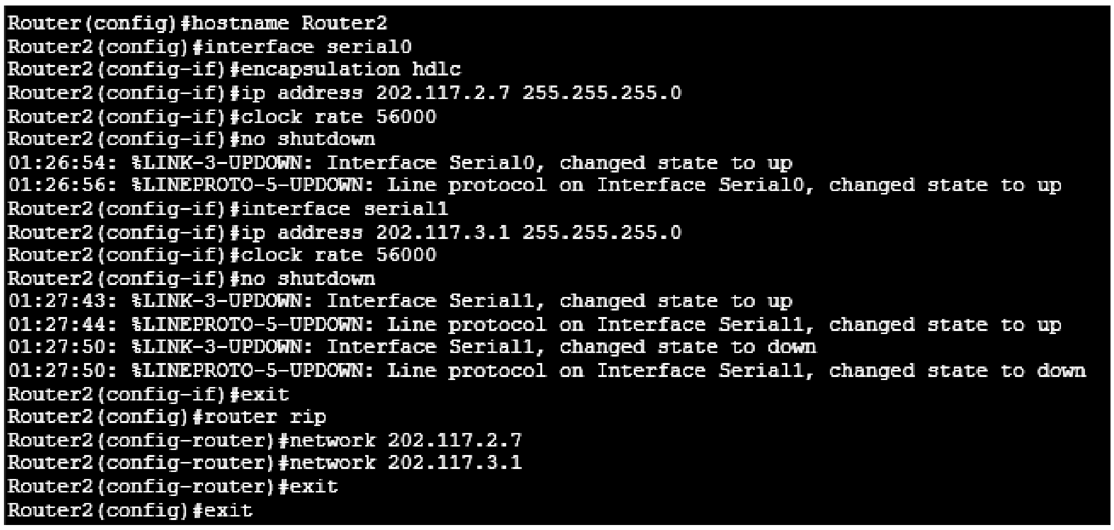
```
Router1(config)#router rip
Router1(config-router)#network 202.117.1.0
Router1(config-router)#network 202.117.2.0
Router1(config-router)#end
Router1#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
```

Router2:



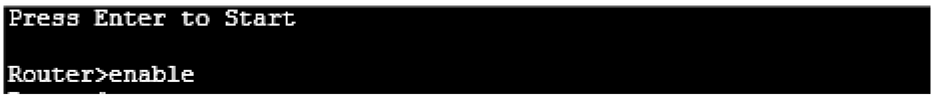
```
Press Enter to Start

Router>enable
Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
```



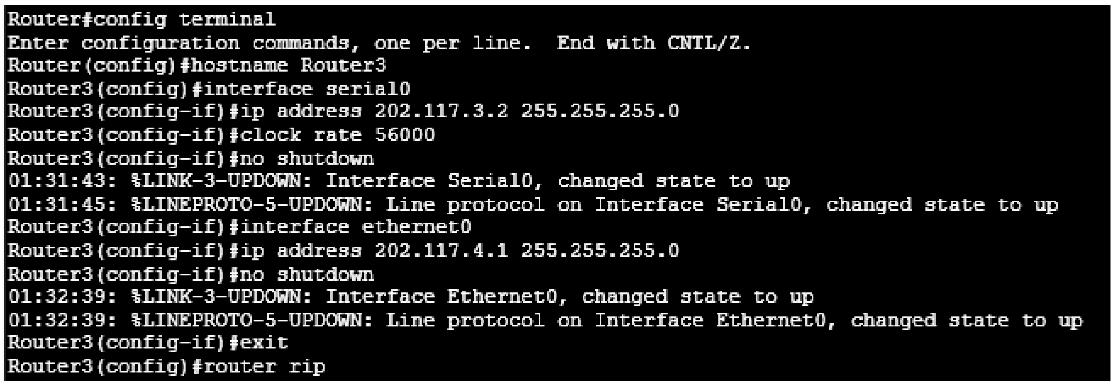
```
Router(config)#hostname Router2
Router2(config)#interface serial0
Router2(config-if)#encapsulation hdlc
Router2(config-if)#ip address 202.117.2.7 255.255.255.0
Router2(config-if)#clock rate 56000
Router2(config-if)#no shutdown
01:26:54: %LINK-3-UPDOWN: Interface Serial0, changed state to up
01:26:56: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to up
Router2(config-if)#interface serial1
Router2(config-if)#ip address 202.117.3.1 255.255.255.0
Router2(config-if)#clock rate 56000
Router2(config-if)#no shutdown
01:27:43: %LINK-3-UPDOWN: Interface Serial1, changed state to up
01:27:44: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1, changed state to up
01:27:50: %LINK-3-UPDOWN: Interface Serial1, changed state to down
01:27:50: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial1, changed state to down
Router2(config-if)#exit
Router2(config)#router rip
Router2(config-router)#network 202.117.2.7
Router2(config-router)#network 202.117.3.1
Router2(config-router)#exit
Router2(config)#exit
```

Router3:

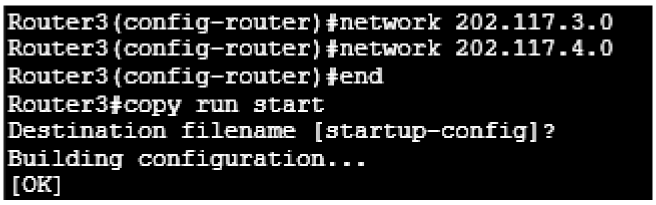


```
Press Enter to Start

Router>enable
```

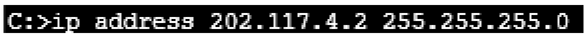


```
Router#config terminal
Enter configuration commands, one per line.  End with CNTL/Z.
Router(config)#hostname Router3
Router3(config)#interface serial0
Router3(config-if)#ip address 202.117.3.2 255.255.255.0
Router3(config-if)#clock rate 56000
Router3(config-if)#no shutdown
01:31:43: %LINK-3-UPDOWN: Interface Serial0, changed state to up
01:31:45: %LINEPROTO-5-UPDOWN: Line protocol on Interface Serial0, changed state to up
Router3(config-if)#interface ethernet0
Router3(config-if)#ip address 202.117.4.1 255.255.255.0
Router3(config-if)#no shutdown
01:32:39: %LINK-3-UPDOWN: Interface Ethernet0, changed state to up
01:32:39: %LINEPROTO-5-UPDOWN: Line protocol on Interface Ethernet0, changed state to up
Router3(config-if)#exit
Router3(config)#router rip
```

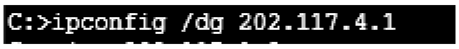


```
Router3(config-router)#network 202.117.3.0
Router3(config-router)#network 202.117.4.0
Router3(config-router)#end
Router3#copy run start
Destination filename [startup-config]?
Building configuration...
[OK]
```

PC2:



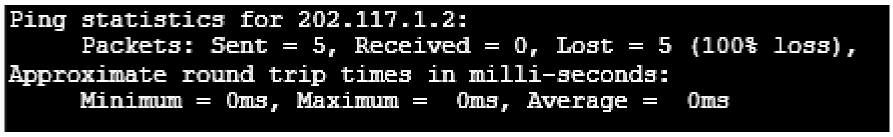
```
C:>ip address 202.117.4.2 255.255.255.0
```



```
C:>ipconfig /dg 202.117.4.1
```

步骤四：测试

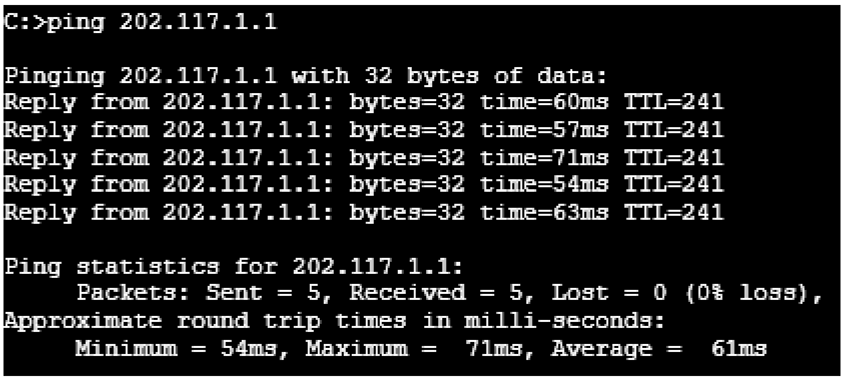
PC2 ping PC1



```
Ping statistics for 202.117.1.2:
     Packets: Sent = 5, Received = 0, Lost = 5 (100% loss),
Approximate round trip times in milli-seconds:
     Minimum = 0ms, Maximum =  0ms, Average =  0ms
```

这里失败了,开始查原因:

测试 PC2 ping Router1



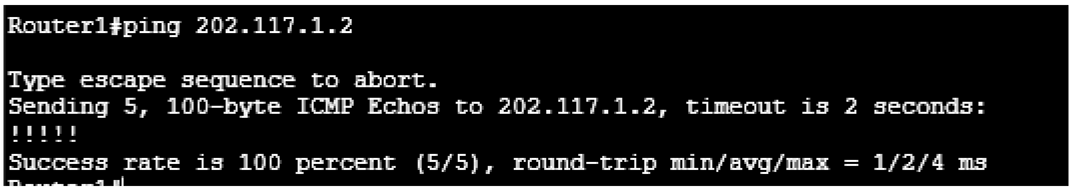
```
C:>ping 202.117.1.1

Pinging 202.117.1.1 with 32 bytes of data:
Reply from 202.117.1.1: bytes=32 time=60ms TTL=241
Reply from 202.117.1.1: bytes=32 time=57ms TTL=241
Reply from 202.117.1.1: bytes=32 time=71ms TTL=241
Reply from 202.117.1.1: bytes=32 time=54ms TTL=241
Reply from 202.117.1.1: bytes=32 time=63ms TTL=241

Ping statistics for 202.117.1.1:
     Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
     Minimum = 54ms, Maximum =  71ms, Average =  61ms
```

这里发现可以 ping 通,所以问题应该出在 Router1 和 PC1 之间

测试 Router1 ping PC1

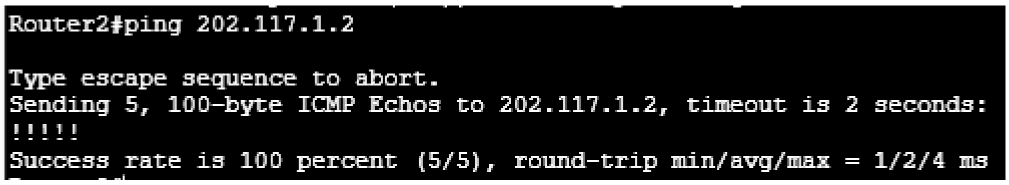


```
Router1#ping 202.117.1.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 202.117.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

居然成功了,这就奇怪了

那么测试 Router2 ping PC1

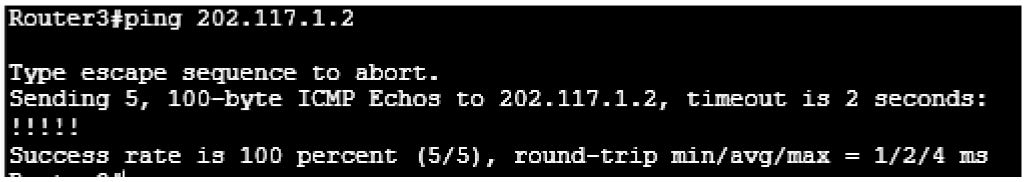


```
Router2#ping 202.117.1.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 202.117.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

居然又成功了,不对,怀疑原本就没问题

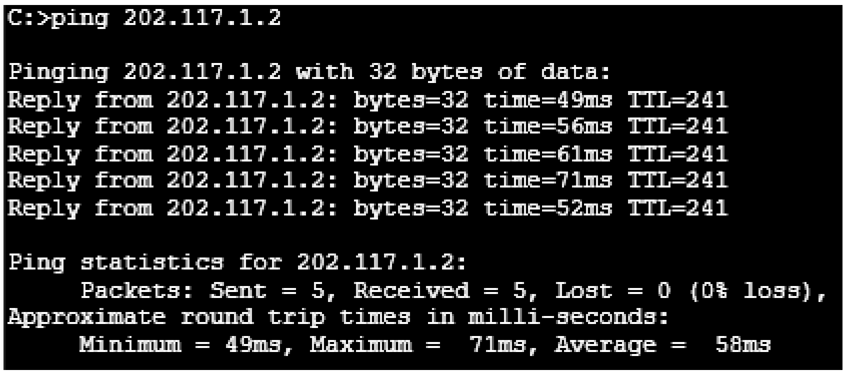
为了严谨,还是测试 Router3 ping PC1



```
Router3#ping 202.117.1.2

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 202.117.1.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 1/2/4 ms
```

成功了,最后重新测试 PC2 ping PC1



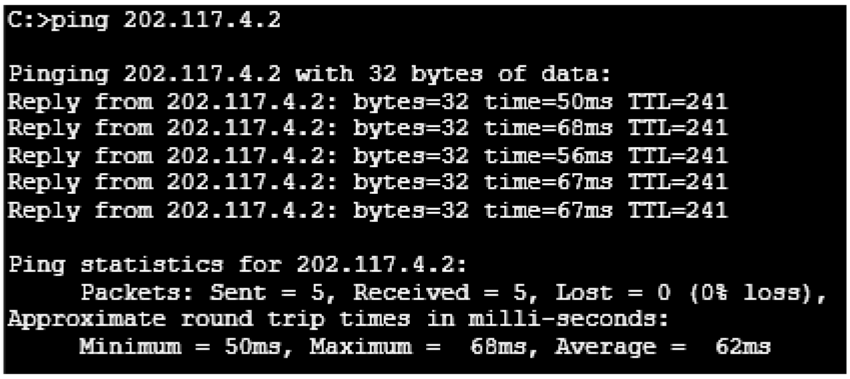
```
C:>ping 202.117.1.2

Pinging 202.117.1.2 with 32 bytes of data:
Reply from 202.117.1.2: bytes=32 time=49ms TTL=241
Reply from 202.117.1.2: bytes=32 time=56ms TTL=241
Reply from 202.117.1.2: bytes=32 time=61ms TTL=241
Reply from 202.117.1.2: bytes=32 time=71ms TTL=241
Reply from 202.117.1.2: bytes=32 time=52ms TTL=241

Ping statistics for 202.117.1.2:
     Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
     Minimum = 49ms, Maximum =  71ms, Average =  58ms
```

果然又没错了

再测试 PC1 ping PC2



```
C:>ping 202.117.4.2

Pinging 202.117.4.2 with 32 bytes of data:
Reply from 202.117.4.2: bytes=32 time=50ms TTL=241
Reply from 202.117.4.2: bytes=32 time=68ms TTL=241
Reply from 202.117.4.2: bytes=32 time=56ms TTL=241
Reply from 202.117.4.2: bytes=32 time=67ms TTL=241
Reply from 202.117.4.2: bytes=32 time=67ms TTL=241

Ping statistics for 202.117.4.2:
     Packets: Sent = 5, Received = 5, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
     Minimum = 50ms, Maximum =  68ms, Average =  62ms
```

结果也成功了

推测是 NetSim 软件问题,我们设置本来没有问题的.最后也没出问题.

实验总结

指令总结

指令总结	指令描述
clock rate clock-rate	sets the clock rate for a Data Communications Equipment (DCE) interface
configure terminal	enters global configuration mode from privileged EXEC mode
enable	enters privileged exec mode
end	ends and exits configuration mode
exit	exits one level in the menu structure
hostname host-name	sets the device name
interface type number	changes from global configuration mode to interface configuration mode
ip address ip-address subnet-mask	assigns an IP address to an interface

network network-address	activates the specified routing protocol on the specified network
ping ip-address	sends an Internet Control Message Protocol (ICMP) echo request to the specified address
router eigrp autonomous-system-number	enters router configuration mode for EIGRP
show ip interface brief	displays a brief summary of interface status and configuration
show ip route	displays the IP routing table
show running-config	displays the active configuration file
shutdown; no shutdown	disables an interface; enables an interface

以下显示本次实验所设置的 ip 地址以及掩码和网关:

NAME	IP	GATE	MASK	PROTECOL
PC1	202.117.1.2	202.117.1.1	255.255.255.0	
Router1	202.117.1.1\202.117.2.1		255.255.255.0	Rip
Router1	202.117.2.7\202.117.3.1		255.255.255.0	Rip
Router1	202.117.3.2\202.117.4.1		255.255.255.0	Rip
PC2	202.117.4.2	202.117.4.1	255.255.255.0	

计算机 72 班 徐子越 2174410455

日期 2019/11/28