计算机网络实验二

Web服务器配置，HTTP报文捕获

作者：鲁含章

学号：1811398

日期：2020年11月10日

目录

一、实验要求

1. 搭建Web服务器（自由选择系统），并制作简单Web页面，包含简单文本信息（至少包含专业、学号、姓名）。
2. 通过浏览器获取自己编写的Web页面，使用Wireshark捕获与Web服务器的交互过程，并进行简单分析说明。

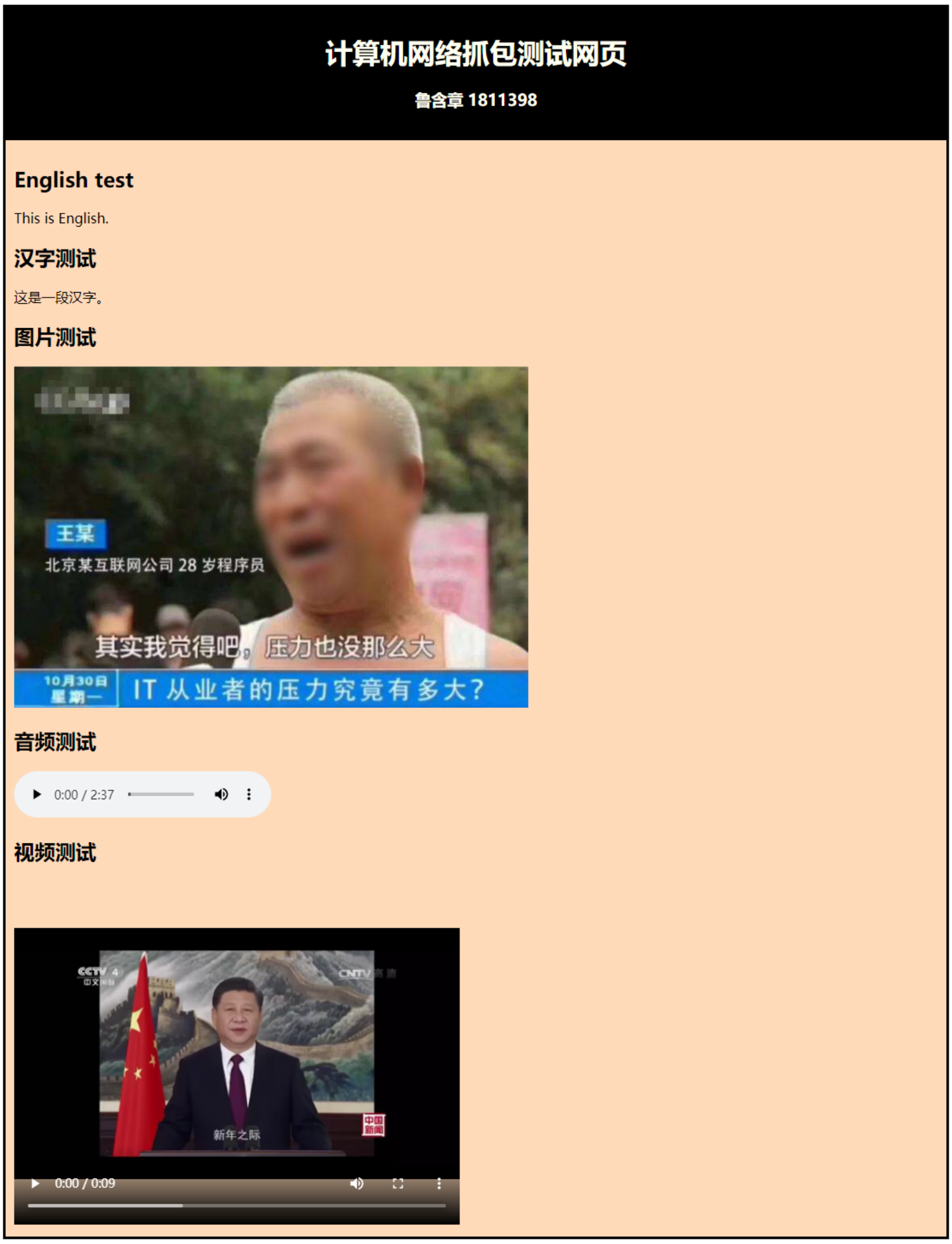
二、实验内容及结果

（一）搭建web服务器，制作简单Web页面

web页面中加入**图片、音频、视频**做不同传输测试。

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>抓包测试</title>
<style>
.center {
    margin: auto;
    width: 80%;
    border: 3px solid black;
    padding: 10px;
}
</style>
</head>

<body>
    <div class="center" align="center" style="color:ivory;background-color:black;">
        <h1>计算机网络抓包测试网页</h1>
        <h3>鲁含章 1811398</h3>
    </div>
    <div class="center" style="color:black;background-color:peachpuff;">
        <h2>English test</h2>
            This is English.

        <h2>汉字测试</h2>
            这是一段汉字。</br>

        <h2>图片测试</h2>
            <img src="source/1.jpg" alt="" width="600">

        <h2>音频测试</h2>
            <audio controls="controls">
                <source src="source/test.mp3" type="audio/mpeg">
            Your browser does not support the audio tag.
            </audio>

        <h2>视频测试</h2>
            <video width="520" height="400" controls="controls" loop="loop">
                <source src="source/movie.mp4" type="video/mp4">
            Your browser does not support the video tag.
            </video>

    </div>
</body>
</html>
```

因为windows下该版本的wireshark无法直接抓取本地回环数据包，故将网页搭载到github的远程服务器，访问https://hzkztech.github.io/ 可以得到如下测试页面。



计算机网络抓包测试网页

鲁含章 1811398

English test

This is English.

汉字测试

这是一段汉字。

图片测试

王某
北京某互联网公司 28 岁程序员
其实我觉得吧，压力也没那么大
10月30日 星期一
IT 从业者的压力究竟有多大?

音频测试

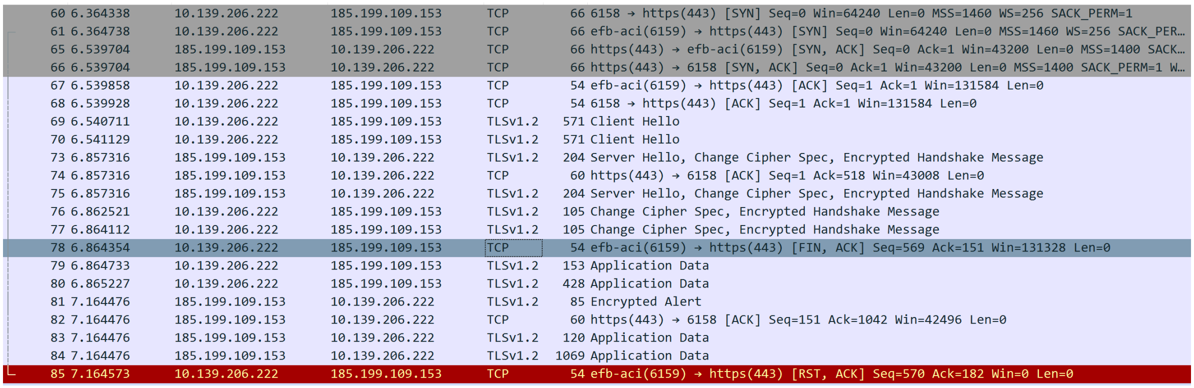
视频测试

CCTV 4
CNTV
新年之际
0:00 / 0:09

（二）使用Wireshark捕获交互过程

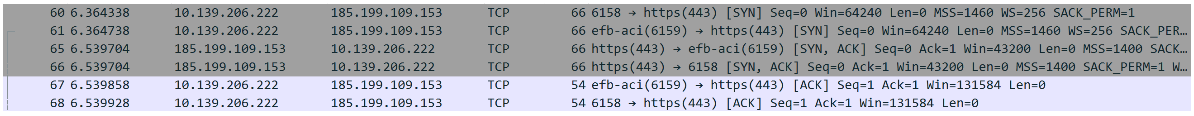
访问https://hzkztech.github.io/，该域名被解析为ip地址185.199.109.153。本站点使用https协议，其中对于数据的传输是TLSv1.2 协议的加密方式，通过wireshark无法直接获取确定内容，不过可以观察其在不同用户端动作之后的交互过程。

对使用 wireshark 捕获结果，使用`ip.addr == 185.199.109.153`过滤得到与本相关其中编号78以前是握手阶段，78号之后是握手后双方使用商议好的秘钥进行通讯。



```
60 6.364338  10.139.206.222   185.199.109.153  TCP       66 6158 → https(443) [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
61 6.364738  10.139.206.222   185.199.109.153  TCP       66 efb-aci(6159) → https(443) [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PER…
65 6.539704  185.199.109.153  10.139.206.222   TCP       66 https(443) → efb-aci(6159) [SYN, ACK] Seq=0 Ack=1 Win=43200 Len=0 MSS=1400 SACK…
66 6.539704  185.199.109.153  10.139.206.222   TCP       66 https(443) → 6158 [SYN, ACK] Seq=0 Ack=1 Win=43200 Len=0 MSS=1400 SACK_PERM=1 W…
67 6.539858  10.139.206.222   185.199.109.153  TCP       54 efb-aci(6159) → https(443) [ACK] Seq=1 Ack=1 Win=131584 Len=0
68 6.539928  10.139.206.222   185.199.109.153  TCP       54 6158 → https(443) [ACK] Seq=1 Ack=1 Win=131584 Len=0
69 6.540711  10.139.206.222   185.199.109.153  TLSv1.2  571 Client Hello
70 6.541129  10.139.206.222   185.199.109.153  TLSv1.2  571 Client Hello
73 6.857316  185.199.109.153  10.139.206.222   TLSv1.2  204 Server Hello, Change Cipher Spec, Encrypted Handshake Message
74 6.857316  185.199.109.153  10.139.206.222   TCP       60 https(443) → 6158 [ACK] Seq=1 Ack=518 Win=43008 Len=0
75 6.857316  185.199.109.153  10.139.206.222   TLSv1.2  204 Server Hello, Change Cipher Spec, Encrypted Handshake Message
76 6.862521  10.139.206.222   185.199.109.153  TLSv1.2  105 Change Cipher Spec, Encrypted Handshake Message
77 6.864112  10.139.206.222   185.199.109.153  TLSv1.2  105 Change Cipher Spec, Encrypted Handshake Message
78 6.864354  10.139.206.222   185.199.109.153  TCP       54 efb-aci(6159) → https(443) [FIN, ACK] Seq=569 Ack=151 Win=131328 Len=0
79 6.864733  10.139.206.222   185.199.109.153  TLSv1.2  153 Application Data
80 6.865227  10.139.206.222   185.199.109.153  TLSv1.2  428 Application Data
81 7.164476  185.199.109.153  10.139.206.222   TLSv1.2   85 Encrypted Alert
82 7.164476  185.199.109.153  10.139.206.222   TCP       60 https(443) → 6158 [ACK] Seq=151 Ack=1042 Win=42496 Len=0
83 7.164476  185.199.109.153  10.139.206.222   TLSv1.2  120 Application Data
84 7.164476  185.199.109.153  10.139.206.222   TLSv1.2 1069 Application Data
85 7.164573  10.139.206.222   185.199.109.153  TCP       54 efb-aci(6159) → https(443) [RST, ACK] Seq=570 Ack=182 Win=0 Len=0
```

1. tcp/ip 三次握手



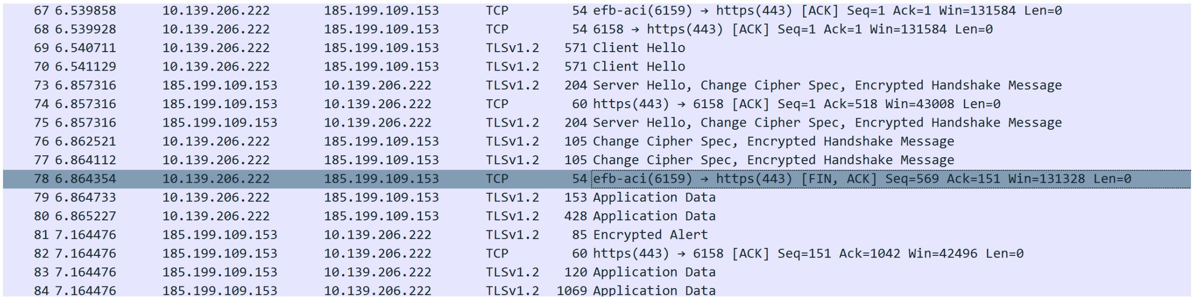
```
60 6.364338   10.139.206.222   185.199.109.153   TCP     66 6158 → https(443) [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
61 6.364738   10.139.206.222   185.199.109.153   TCP     66 efb-aci(6159) → https(443) [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PER…
65 6.539704   185.199.109.153  10.139.206.222    TCP     66 https(443) → efb-aci(6159) [SYN, ACK] Seq=0 Ack=1 Win=43200 Len=0 MSS=1400 SACK…
66 6.539704   185.199.109.153  10.139.206.222    TCP     66 https(443) → 6158 [SYN, ACK] Seq=0 Ack=1 Win=43200 Len=0 MSS=1400 SACK_PERM=1 W…
67 6.539858   10.139.206.222   185.199.109.153   TCP     54 efb-aci(6159) → https(443) [ACK] Seq=1 Ack=1 Win=131584 Len=0
68 6.539928   10.139.206.222   185.199.109.153   TCP     54 6158 → https(443) [ACK] Seq=1 Ack=1 Win=131584 Len=0
```

60：客户端向 443 端口发送 SYN 信号

66：服务端回应连接

68：tcp/ip 三次握手完成

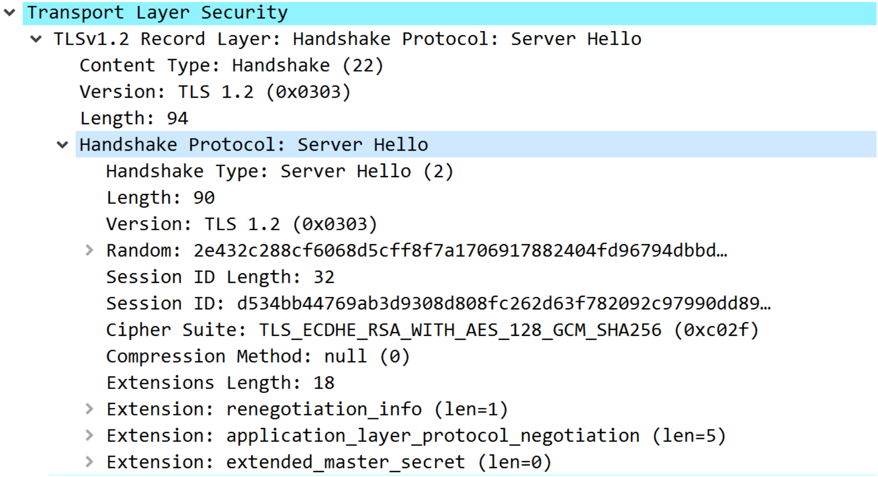
2. TLS 握手过程



```
67 6.539858   10.139.206.222   185.199.109.153   TCP        54 efb-aci(6159) → https(443) [ACK] Seq=1 Ack=1 Win=131584 Len=0
68 6.539928   10.139.206.222   185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1 Ack=1 Win=131584 Len=0
69 6.540711   10.139.206.222   185.199.109.153   TLSv1.2   571 Client Hello
70 6.541129   10.139.206.222   185.199.109.153   TLSv1.2   571 Client Hello
73 6.857316   185.199.109.153  10.139.206.222    TLSv1.2   204 Server Hello, Change Cipher Spec, Encrypted Handshake Message
74 6.857316   185.199.109.153  10.139.206.222    TCP        60 https(443) → 6158 [ACK] Seq=1 Ack=518 Win=43008 Len=0
75 6.857316   185.199.109.153  10.139.206.222    TLSv1.2   204 Server Hello, Change Cipher Spec, Encrypted Handshake Message
76 6.862521   10.139.206.222   185.199.109.153   TLSv1.2   105 Change Cipher Spec, Encrypted Handshake Message
77 6.864112   10.139.206.222   185.199.109.153   TLSv1.2   105 Change Cipher Spec, Encrypted Handshake Message
78 6.864354   10.139.206.222   185.199.109.153   TCP        54 efb-aci(6159) → https(443) [FIN, ACK] Seq=569 Ack=151 Win=131328 Len=0
79 6.864733   10.139.206.222   185.199.109.153   TLSv1.2   153 Application Data
80 6.865227   10.139.206.222   185.199.109.153   TLSv1.2   428 Application Data
81 7.164476   185.199.109.153  10.139.206.222    TLSv1.2    85 Encrypted Alert
82 7.164476   185.199.109.153  10.139.206.222    TCP        60 https(443) → 6158 [ACK] Seq=151 Ack=1042 Win=42496 Len=0
83 7.164476   185.199.109.153  10.139.206.222    TLSv1.2   120 Application Data
84 7.164476   185.199.109.153  10.139.206.222    TLSv1.2  1069 Application Data
```

69：Client Hello 信号，客户端发送随机数字 + 自己可以支持的加密方法。

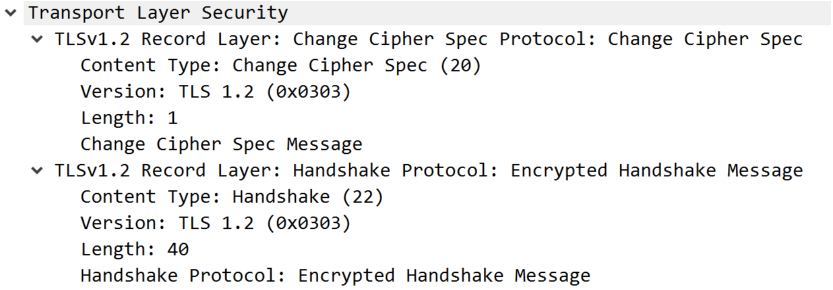
73：Server Hello 信号，服务器发送随机数字 + 选择双方都支持的加密方式，这里选择的是 `TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256`。



```
⌄ Transport Layer Security
  ⌄ TLSv1.2 Record Layer: Handshake Protocol: Server Hello
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 94
    ⌄ Handshake Protocol: Server Hello
        Handshake Type: Server Hello (2)
        Length: 90
        Version: TLS 1.2 (0x0303)
      › Random: 2e432c288cf6068d5cff8f7a1706917882404fd96794dbbd…
        Session ID Length: 32
        Session ID: d534bb44769ab3d9308d808fc262d63f782092c97990dd89…
        Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
        Compression Method: null (0)
        Extensions Length: 18
      › Extension: renegotiation_info (len=1)
      › Extension: application_layer_protocol_negotiation (len=5)
      › Extension: extended_master_secret (len=0)
```

74：ACK 应答

76：发送自己的公钥和从 CA 申请的证书



```
⌄ Transport Layer Security
  ⌄ TLSv1.2 Record Layer: Change Cipher Spec Protocol: Change Cipher Spec
      Content Type: Change Cipher Spec (20)
      Version: TLS 1.2 (0x0303)
      Length: 1
      Change Cipher Spec Message
  ⌄ TLSv1.2 Record Layer: Handshake Protocol: Encrypted Handshake Message
      Content Type: Handshake (22)
      Version: TLS 1.2 (0x0303)
      Length: 40
      Handshake Protocol: Encrypted Handshake Message
```

79，80：客户端发送的请求头（数据经过加密）

84：服务端用密文形式发送的 html 文档

3. 数据传输过程TCP+TLSv1.2

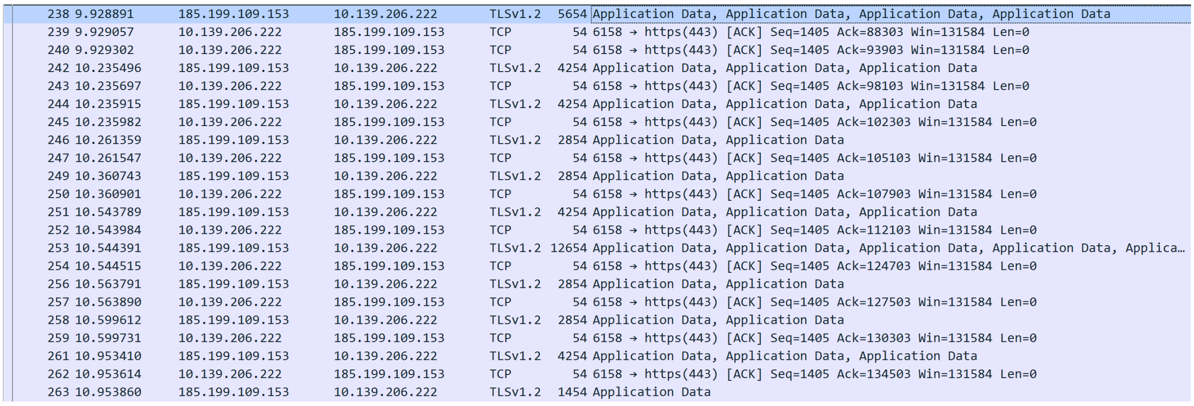
测试图片、音频、视频等文件的传输

服务器发来的数据都经过加密的，无法解析出具体内容。

在抓取的包中观察到如下几类交互过程。不同于小size的文本，这些文件在传输时一次TCP段长度（MSS）是不够的，需要多次传输，而这期间客户端只回应ACK。

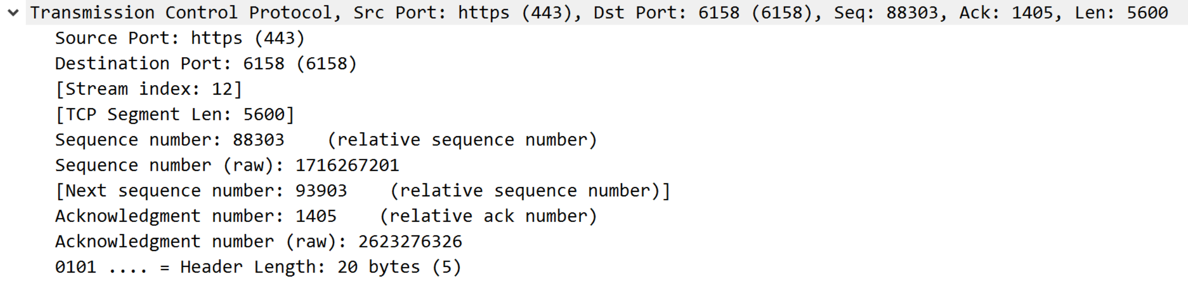
（1）正常传输过程：

注意到这里客户端发送的Seq始终为0，这是因为客户端每次只发送ACK报文，`TCP Segment Len: 0`，`Sequence number`与`Next Sequence number`相同，服务器每次反馈的ACK也便相同。这种情况在客户端发送具体请求后变化，例如当按下视频播放按钮，请求服务器发送后续的视频内容时，`TCP Segment Len`变为有效值。



```
238 9.928891    185.199.109.153   10.139.206.222    TLSv1.2  5654 Application Data, Application Data, Application Data, Application Data
239 9.929057    10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=88303 Win=131584 Len=0
240 9.929302    10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=93903 Win=131584 Len=0
242 10.235496   185.199.109.153   10.139.206.222    TLSv1.2  4254 Application Data, Application Data, Application Data
243 10.235697   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=98103 Win=131584 Len=0
244 10.235915   185.199.109.153   10.139.206.222    TLSv1.2  4254 Application Data, Application Data, Application Data
245 10.235982   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=102303 Win=131584 Len=0
246 10.261359   185.199.109.153   10.139.206.222    TLSv1.2  2854 Application Data, Application Data
247 10.261547   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=105103 Win=131584 Len=0
249 10.360743   185.199.109.153   10.139.206.222    TLSv1.2  2854 Application Data, Application Data
250 10.360901   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=107903 Win=131584 Len=0
251 10.543789   185.199.109.153   10.139.206.222    TLSv1.2  4254 Application Data, Application Data, Application Data
252 10.543984   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=112103 Win=131584 Len=0
253 10.544391   185.199.109.153   10.139.206.222    TLSv1.2 12654 Application Data, Application Data, Application Data, Application Data, Applica…
254 10.544515   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=124703 Win=131584 Len=0
256 10.563791   185.199.109.153   10.139.206.222    TLSv1.2  2854 Application Data, Application Data
257 10.563890   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=127503 Win=131584 Len=0
258 10.599612   185.199.109.153   10.139.206.222    TLSv1.2  2854 Application Data, Application Data
259 10.599731   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=130303 Win=131584 Len=0
261 10.953410   185.199.109.153   10.139.206.222    TLSv1.2  4254 Application Data, Application Data, Application Data
262 10.953614   10.139.206.222    185.199.109.153   TCP        54 6158 → https(443) [ACK] Seq=1405 Ack=134503 Win=131584 Len=0
263 10.953860   185.199.109.153   10.139.206.222    TLSv1.2  1454 Application Data
```

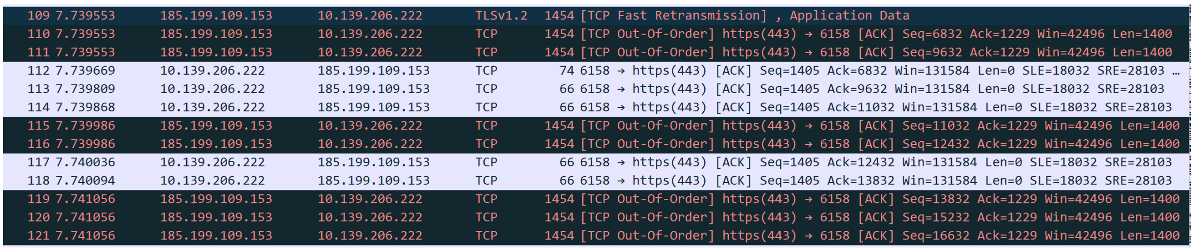
服务器发来的头部包含如下信息：



```
⌄ Transmission Control Protocol, Src Port: https (443), Dst Port: 6158 (6158), Seq: 88303, Ack: 1405, Len: 5600
     Source Port: https (443)
     Destination Port: 6158 (6158)
     [Stream index: 12]
     [TCP Segment Len: 5600]
     Sequence number: 88303    (relative sequence number)
     Sequence number (raw): 1716267201
     [Next sequence number: 93903    (relative sequence number)]
     Acknowledgment number: 1405    (relative ack number)
     Acknowledgment number (raw): 2623276326
     0101 .... = Header Length: 20 bytes (5)
```

（2）失序处理：

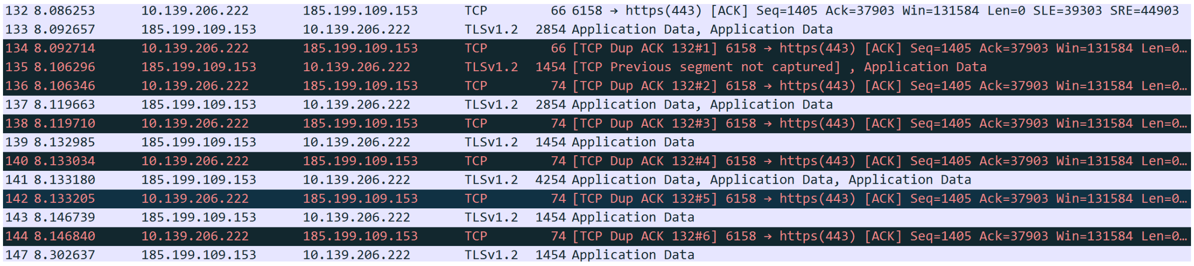
重复发送ACK



```
109 7.739553   185.199.109.153   10.139.206.222    TLSv1.2  1454 [TCP Fast Retransmission] , Application Data
110 7.739553   185.199.109.153   10.139.206.222    TCP      1454 [TCP Out-Of-Order] https(443) → 6158 [ACK] Seq=6832 Ack=1229 Win=42496 Len=1400
111 7.739553   185.199.109.153   10.139.206.222    TCP      1454 [TCP Out-Of-Order] https(443) → 6158 [ACK] Seq=9632 Ack=1229 Win=42496 Len=1400
112 7.739669   10.139.206.222    185.199.109.153   TCP        74 6158 → https(443) [ACK] Seq=1405 Ack=6832 Win=131584 Len=0 SLE=18032 SRE=28103 …
113 7.739809   10.139.206.222    185.199.109.153   TCP        66 6158 → https(443) [ACK] Seq=1405 Ack=9632 Win=131584 Len=0 SLE=18032 SRE=28103
114 7.739868   10.139.206.222    185.199.109.153   TCP        66 6158 → https(443) [ACK] Seq=1405 Ack=11032 Win=131584 Len=0 SLE=18032 SRE=28103
115 7.739986   185.199.109.153   10.139.206.222    TCP      1454 [TCP Out-Of-Order] https(443) → 6158 [ACK] Seq=11032 Ack=1229 Win=42496 Len=1400
116 7.739986   185.199.109.153   10.139.206.222    TCP      1454 [TCP Out-Of-Order] https(443) → 6158 [ACK] Seq=12432 Ack=1229 Win=42496 Len=1400
117 7.740036   10.139.206.222    185.199.109.153   TCP        66 6158 → https(443) [ACK] Seq=1405 Ack=12432 Win=131584 Len=0 SLE=18032 SRE=28103
118 7.740094   10.139.206.222    185.199.109.153   TCP        66 6158 → https(443) [ACK] Seq=1405 Ack=13832 Win=131584 Len=0 SLE=18032 SRE=28103
119 7.741056   185.199.109.153   10.139.206.222    TCP      1454 [TCP Out-Of-Order] https(443) → 6158 [ACK] Seq=13832 Ack=1229 Win=42496 Len=1400
120 7.741056   185.199.109.153   10.139.206.222    TCP      1454 [TCP Out-Of-Order] https(443) → 6158 [ACK] Seq=15232 Ack=1229 Win=42496 Len=1400
121 7.741056   185.199.109.153   10.139.206.222    TCP      1454 [TCP Out-Of-Order] https(443) → 6158 [ACK] Seq=16632 Ack=1229 Win=42496 Len=1400
```

（3）冗余包：

遇到冗余包会重复发送ACK[冗余包的next seq]

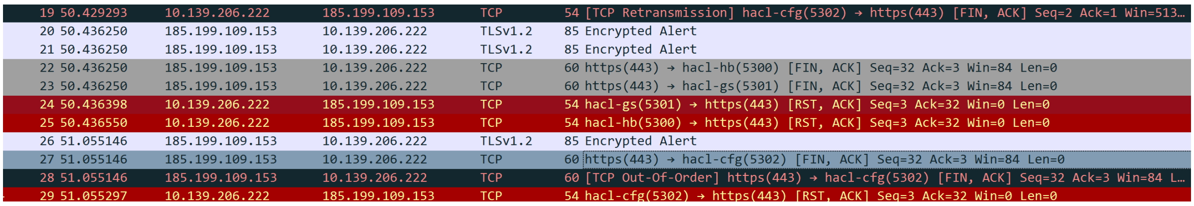


```
132 8.086253   10.139.206.222    185.199.109.153   TCP        66 6158 → https(443) [ACK] Seq=1405 Ack=37903 Win=131584 Len=0 SLE=39303 SRE=44903
133 8.092657   185.199.109.153   10.139.206.222    TLSv1.2  2854 Application Data, Application Data
134 8.092714   10.139.206.222    185.199.109.153   TCP        66 [TCP Dup ACK 132#1] 6158 → https(443) [ACK] Seq=1405 Ack=37903 Win=131584 Len=0…
135 8.106296   185.199.109.153   10.139.206.222    TLSv1.2  1454 [TCP Previous segment not captured] , Application Data
136 8.106346   10.139.206.222    185.199.109.153   TCP        74 [TCP Dup ACK 132#2] 6158 → https(443) [ACK] Seq=1405 Ack=37903 Win=131584 Len=0…
137 8.119663   185.199.109.153   10.139.206.222    TLSv1.2  2854 Application Data, Application Data
138 8.119710   10.139.206.222    185.199.109.153   TCP        74 [TCP Dup ACK 132#3] 6158 → https(443) [ACK] Seq=1405 Ack=37903 Win=131584 Len=0…
139 8.132985   185.199.109.153   10.139.206.222    TLSv1.2  1454 Application Data
140 8.133034   10.139.206.222    185.199.109.153   TCP        74 [TCP Dup ACK 132#4] 6158 → https(443) [ACK] Seq=1405 Ack=37903 Win=131584 Len=0…
141 8.133180   185.199.109.153   10.139.206.222    TLSv1.2  4254 Application Data, Application Data, Application Data
142 8.133205   10.139.206.222    185.199.109.153   TCP        74 [TCP Dup ACK 132#5] 6158 → https(443) [ACK] Seq=1405 Ack=37903 Win=131584 Len=0…
143 8.146739   185.199.109.153   10.139.206.222    TLSv1.2  1454 Application Data
144 8.146840   10.139.206.222    185.199.109.153   TCP        74 [TCP Dup ACK 132#6] 6158 → https(443) [ACK] Seq=1405 Ack=37903 Win=131584 Len=0…
147 8.302637   185.199.109.153   10.139.206.222    TLSv1.2  1454 Application Data
```

4. TCP四次挥手

由于TCP连接是全双工的，因此每个方向都必须单独进行关闭。

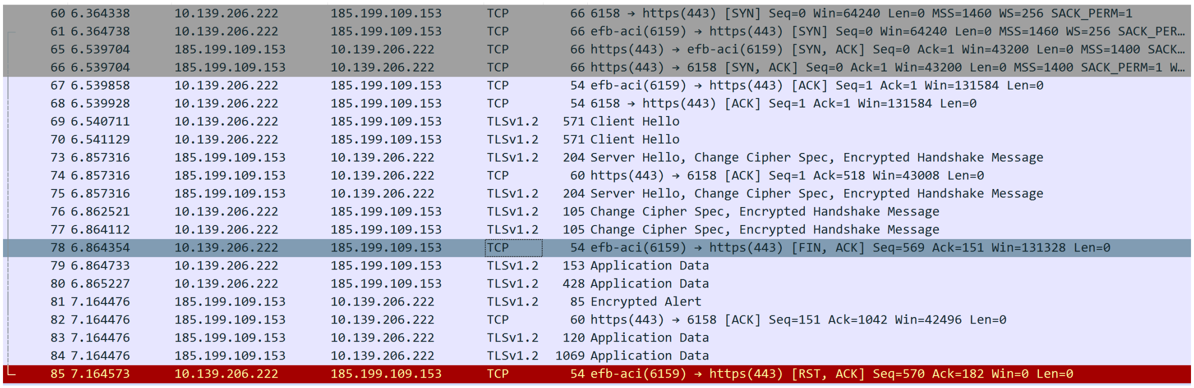
19：TCP客户端发送一个FIN，用来关闭客户到服务器的数据传送
22：服务器收到这个FIN，它发回一个ACK，确认序号为收到的序号加1
22：服务器关闭客户端的连接，发送一个FIN给客户端
24：客户端发回ACK报文确认，并将确认序号设置为收到序号加1



```
   19 50.429293   10.139.206.222    185.199.109.153   TCP       54 [TCP Retransmission] hacl-cfg(5302) → https(443) [FIN, ACK] Seq=2 Ack=1 Win=513…
   20 50.436250   185.199.109.153   10.139.206.222    TLSv1.2   85 Encrypted Alert
   21 50.436250   185.199.109.153   10.139.206.222    TLSv1.2   85 Encrypted Alert
   22 50.436250   185.199.109.153   10.139.206.222    TCP       60 https(443) → hacl-hb(5300) [FIN, ACK] Seq=32 Ack=3 Win=84 Len=0
   23 50.436250   185.199.109.153   10.139.206.222    TCP       60 https(443) → hacl-gs(5301) [FIN, ACK] Seq=32 Ack=3 Win=84 Len=0
   24 50.436398   10.139.206.222    185.199.109.153   TCP       54 hacl-gs(5301) → https(443) [RST, ACK] Seq=3 Ack=32 Win=0 Len=0
   25 50.436550   10.139.206.222    185.199.109.153   TCP       54 hacl-hb(5300) → https(443) [RST, ACK] Seq=3 Ack=32 Win=0 Len=0
   26 51.055146   185.199.109.153   10.139.206.222    TLSv1.2   85 Encrypted Alert
   27 51.055146   185.199.109.153   10.139.206.222    TCP       60 https(443) → hacl-cfg(5302) [FIN, ACK] Seq=32 Ack=3 Win=84 Len=0
   28 51.055146   185.199.109.153   10.139.206.222    TCP       60 [TCP Out-Of-Order] https(443) → hacl-cfg(5302) [FIN, ACK] Seq=32 Ack=3 Win=84 L…
   29 51.055297   10.139.206.222    185.199.109.153   TCP       54 hacl-cfg(5302) → https(443) [RST, ACK] Seq=3 Ack=32 Win=0 Len=0
```

5. 其他现象——chrome的双端口连接

在TCP握手时，可以观察到本地同时发出了两个不同端口的连接请求——`6158`与`6159`端口，其中6158是之后持续使用的端口，而6159端口在不久之后就由客户端断开连接（见78，81，85）。经搜索，这是许多浏览器的一种加速访问策略：一次建立两个连接，然后使用其中一个继续通信，避免有一个连接失败的情况。



```
     60 6.364338    10.139.206.222    185.199.109.153   TCP       66 6158 → https(443) [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM=1
     61 6.364738    10.139.206.222    185.199.109.153   TCP       66 efb-aci(6159) → https(443) [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PER…
     65 6.539704    185.199.109.153   10.139.206.222    TCP       66 https(443) → efb-aci(6159) [SYN, ACK] Seq=0 Ack=1 Win=43200 Len=0 MSS=1400 SACK…
     66 6.539704    185.199.109.153   10.139.206.222    TCP       66 https(443) → 6158 [SYN, ACK] Seq=0 Ack=1 Win=43200 Len=0 MSS=1400 SACK_PERM=1 W…
     67 6.539858    10.139.206.222    185.199.109.153   TCP       54 efb-aci(6159) → https(443) [ACK] Seq=1 Ack=1 Win=131584 Len=0
     68 6.539928    10.139.206.222    185.199.109.153   TCP       54 6158 → https(443) [ACK] Seq=1 Ack=1 Win=131584 Len=0
     69 6.540711    10.139.206.222    185.199.109.153   TLSv1.2  571 Client Hello
     70 6.541129    10.139.206.222    185.199.109.153   TLSv1.2  571 Client Hello
     73 6.857316    185.199.109.153   10.139.206.222    TLSv1.2  204 Server Hello, Change Cipher Spec, Encrypted Handshake Message
     74 6.857316    185.199.109.153   10.139.206.222    TCP       60 https(443) → 6158 [ACK] Seq=1 Ack=518 Win=43008 Len=0
     75 6.857316    185.199.109.153   10.139.206.222    TLSv1.2  204 Server Hello, Change Cipher Spec, Encrypted Handshake Message
     76 6.862521    10.139.206.222    185.199.109.153   TLSv1.2  105 Change Cipher Spec, Encrypted Handshake Message
     77 6.864112    10.139.206.222    185.199.109.153   TLSv1.2  105 Change Cipher Spec, Encrypted Handshake Message
     78 6.864354    10.139.206.222    185.199.109.153   TCP       54 efb-aci(6159) → https(443) [FIN, ACK] Seq=569 Ack=151 Win=131328 Len=0
     79 6.864733    10.139.206.222    185.199.109.153   TLSv1.2  153 Application Data
     80 6.865227    10.139.206.222    185.199.109.153   TLSv1.2  428 Application Data
     81 7.164476    185.199.109.153   10.139.206.222    TLSv1.2   85 Encrypted Alert
     82 7.164476    185.199.109.153   10.139.206.222    TCP       60 https(443) → 6158 [ACK] Seq=151 Ack=1042 Win=42496 Len=0
     83 7.164476    185.199.109.153   10.139.206.222    TLSv1.2  120 Application Data
     84 7.164476    185.199.109.153   10.139.206.222    TLSv1.2 1069 Application Data
     85 7.164573    10.139.206.222    185.199.109.153   TCP       54 efb-aci(6159) → https(443) [RST, ACK] Seq=570 Ack=182 Win=0 Len=0
```

三、总结

在本次实验中，实现了包括文本、图片、音频和视频的测试html页面，搭载到web服务器上，并通过Wireshark抓取客户端与服务器段的交互过程，观察到TCP握手/挥手过程、TLS握手/传输过程以及对一些异常情况的处理。

通过本次实验，对TCP/IP协议的过程有了更深层次的理解，为后续的实验做好了准备。

参考文献

[1] https://blog.csdn.net/lblblblblzdx/article/details/88684788 **TLSv1.2 协议了解**

[2] https://www.cnblogs.com/monkey0307/p/9675123.html **SSL/TSL 握手过程详解**