# **实验3报告模板**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名： |  | 学号： |  | 班级： |  |
|  |  |  |  |  |  |

# **3 实验三 应用层协议分析**

## 3.1 实验目的

1. 分析应用层协议（如HTTP、HTTPS）的工作过程，理解应用层与传输层及下层协议的关系，分析缓存工作原理及好处。

2. 能够依据HTTP协议编写简单的HTTP客户端程序，从几个网站去读取主页，或者取文件。

## 3.2 实验内容

（1）利用现有校园网络及外网服务器搭建内网、外网环境；

（2）用Wireshark截获HTTP/HTTPS报文，分析报文结构及浏览器和服务器的交互过程；分析HTTP协议的缓存机制，分析HTTPS的加密传输机制。分析应用层协议跟TCP/DNS等协议的交互关系；

（3）依据HTTP协议，编写简单的HTTP客户端程序，从几个网站去读取主页，或者取文件。

## 3.3 实验步骤

### 3.3.1 实验环境

依托校园网和外网服务器搭建内网、外网环境，每名同学单独完成实验并撰写实验报告。

### 3.3.2 HTTP协议分析

清空缓存后的DNS和HTTP协议分析：

步骤1：在计算机终端上运行Wireshark开始截获报文。

步骤2：清空DNS和HTTP浏览器的缓存：

浏览器缓存的清除以Chrome浏览器为例，地址栏中输入chrome://settings/，找到高级选项中的“隐私设置和安全性”，清除浏览数据。

在cmd命令行执行“ipconfig /flushdns”清除本地DNS缓存（或者在PowerShell用Clear-DnsClientCache/Get-DnsClientCache命令清除/查看DNS缓存）。

步骤3：在浏览器中访问3个网址，分别是国内、国外和南半球网址。

国内网址：如http://www.xjtu.edu.cn（或http://www.people.com.cn/）。

**注意：必须指定http，目前很多网站默认是https，无法抓到http数据包；**

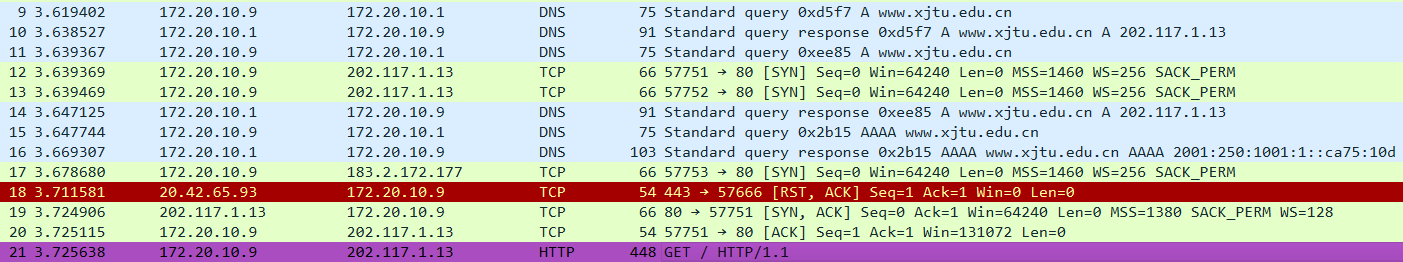
国外网址：如www.github.com, www.python.org；

南半球网址：如阿根廷的www.unla.edu.ar（或www.uns.edu.ar，en.unne.edu.ar，www.unne.edu.ar，www.unsj.edu.ar等）；或巴西的www.unb.br；或南非的www.unisa.ac.za（或www.up.ac.za）。

步骤4：网站加载完成之后，停止报文截获并保存数据，分析截获的报文，填写表5-1中“无缓存”列信息。

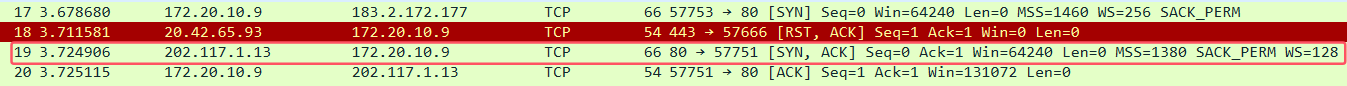
**（1）国内网址：http://www.xjtu.edu.cn**

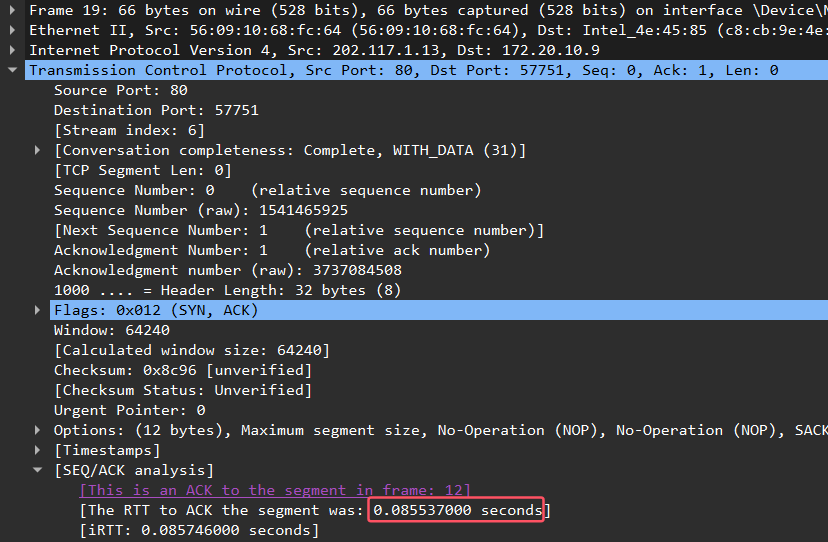
DNS查询解析次数：3次



应答时间：0.0855s

第一个应答报文：

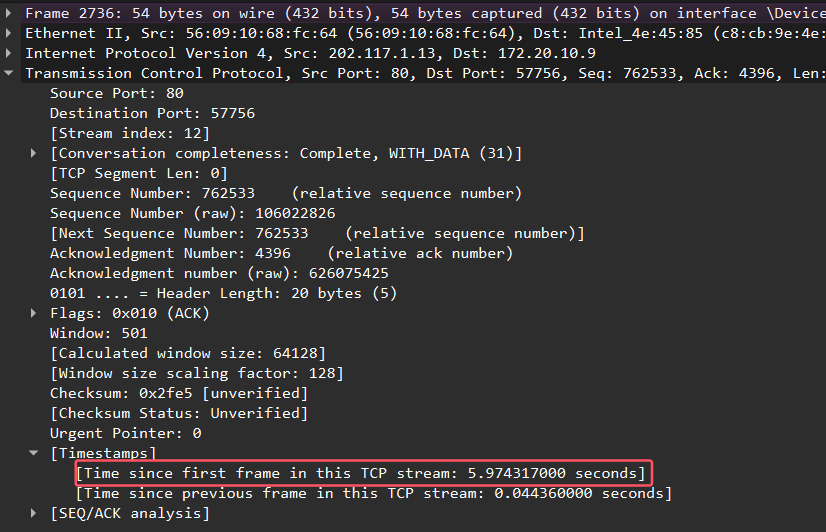




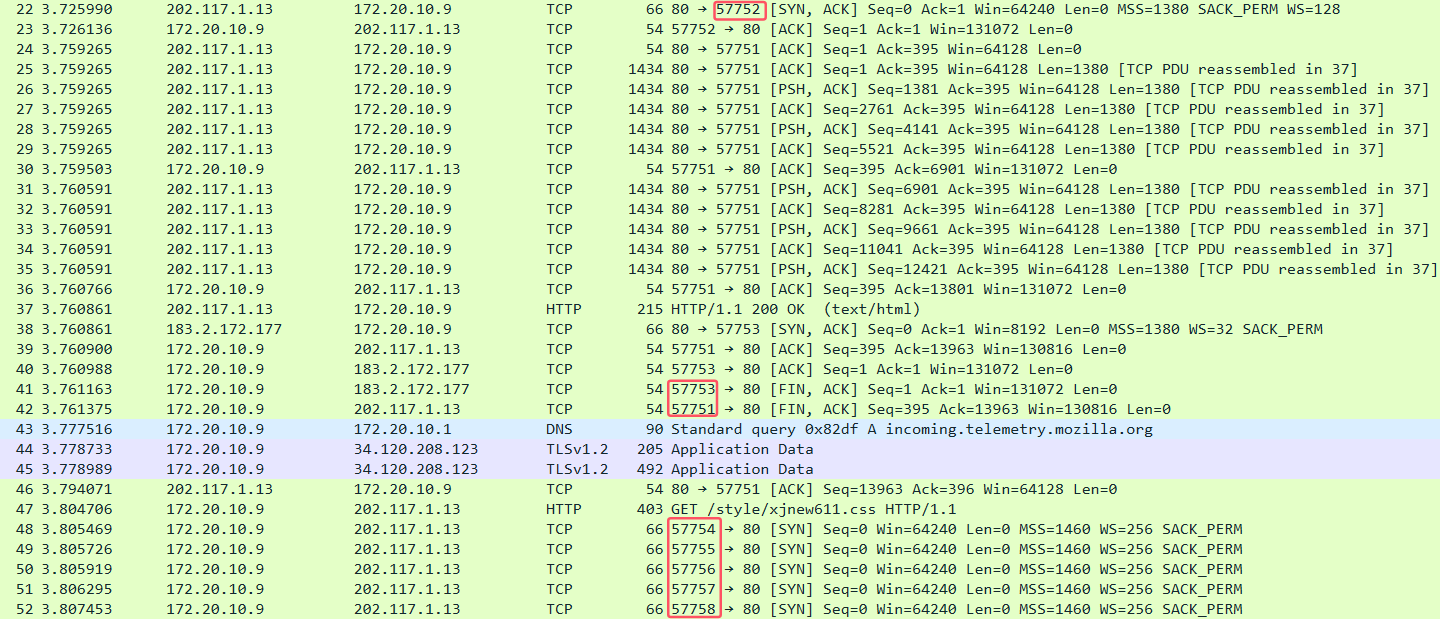
加载时间：5.9743s

最后一个报文：

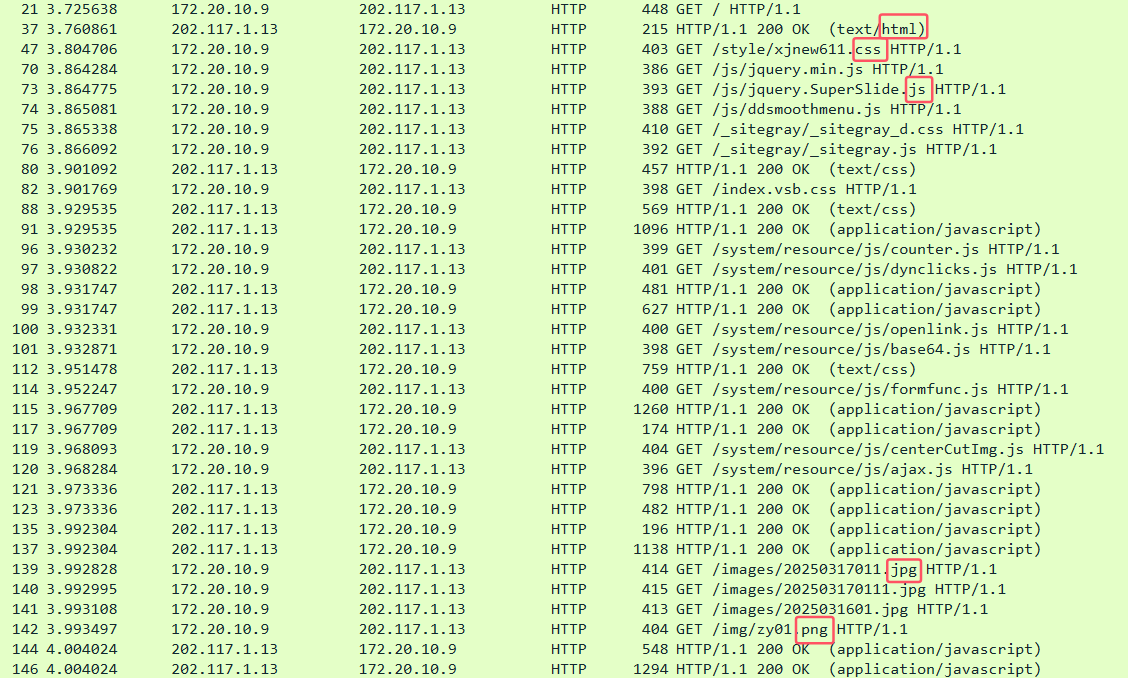




连接建立数：8个，本地端口号从57751至57758



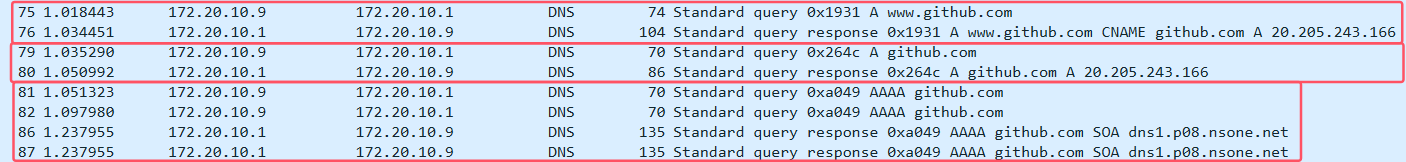
请求对象数：HTML、CSS、JS、JPG、PNG、GIF、JPEG、JFIF共8个



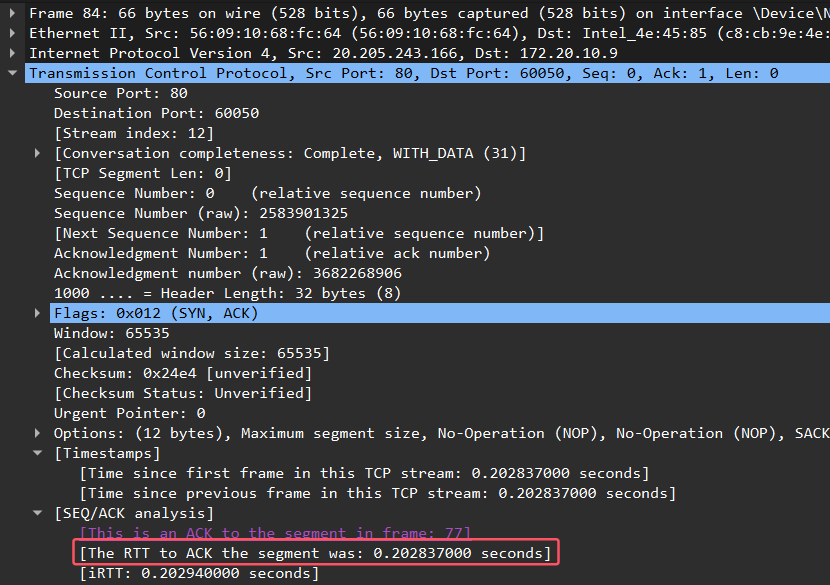


**（2）国外网址：[www.github.com](http://www.github.com)**

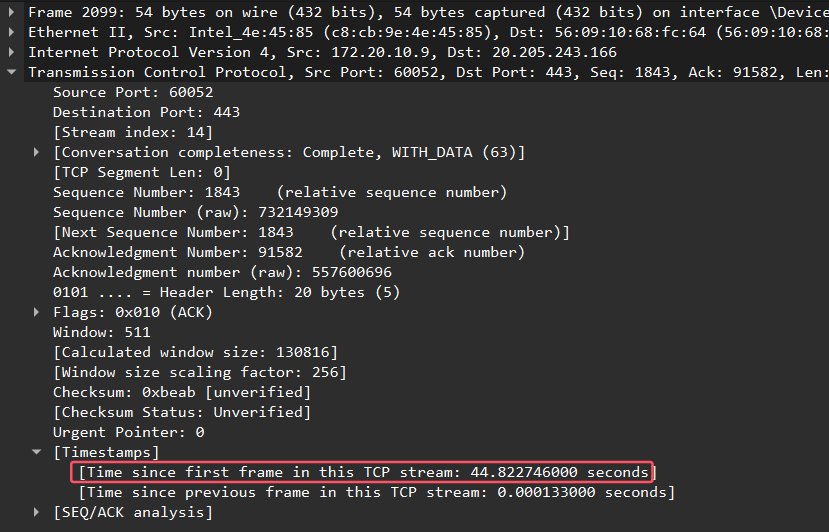
DNS查询和解析次数：3次



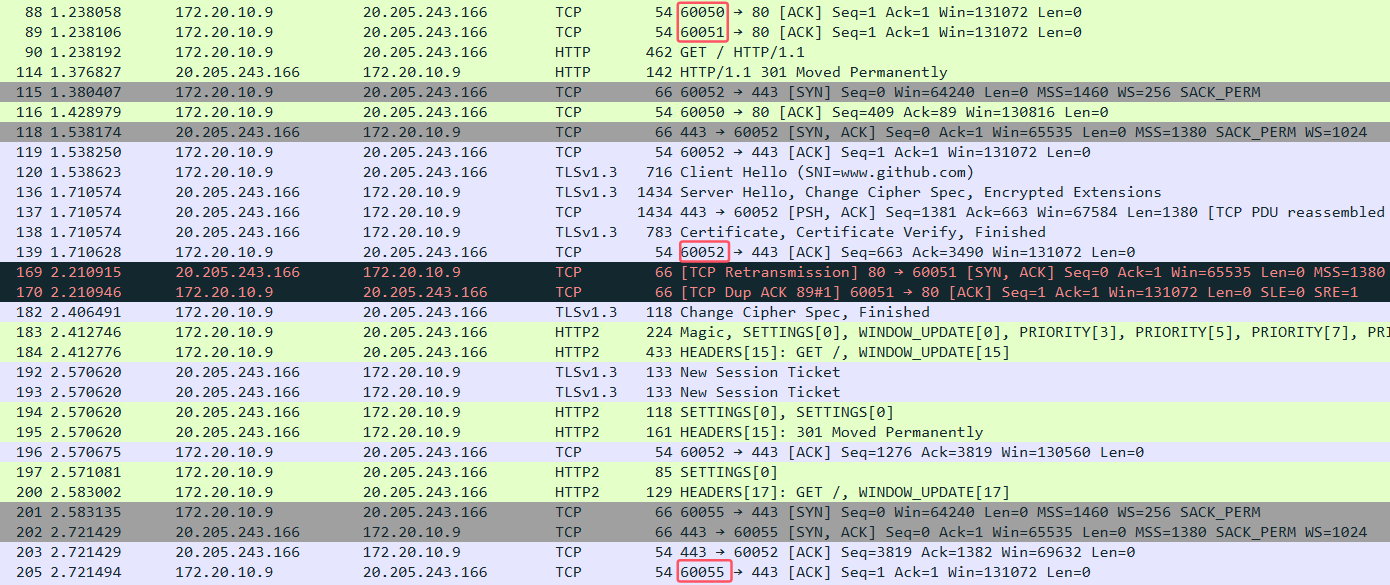
应答时间：0.2028s



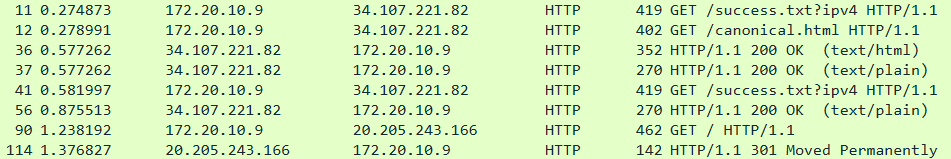
加载时间：44.8228s



建立连接数：4个，本地端口号为60050、60051、60052、60055

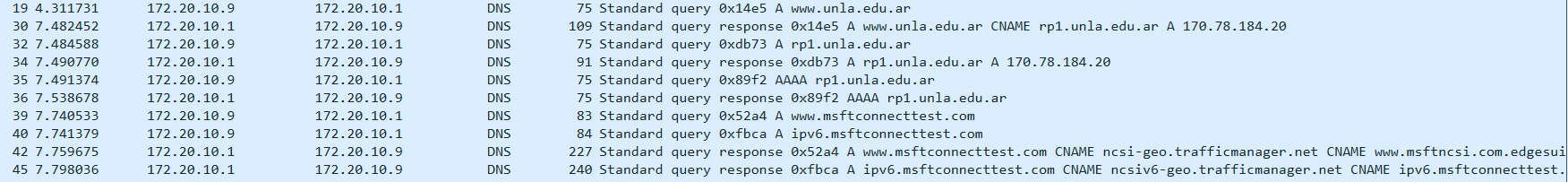


请求对象数：2个，HTML、TXT

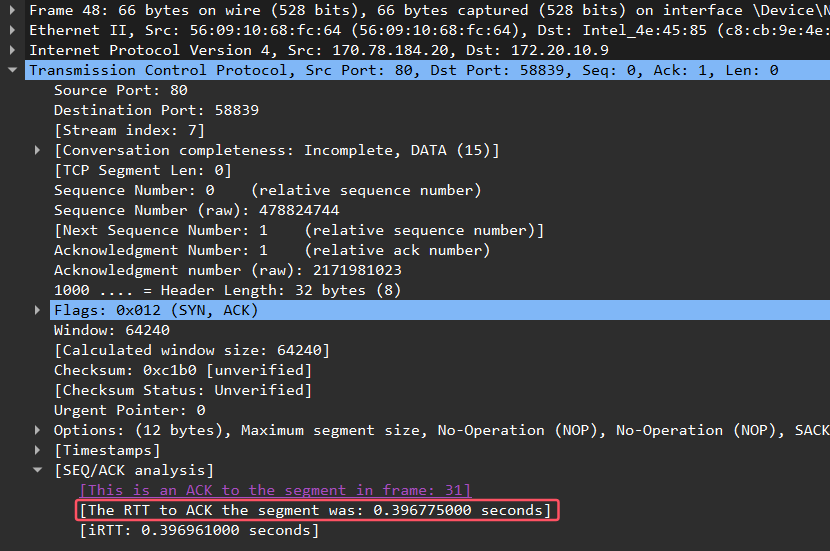


**（3）南半球网址：[www.unla.edu.ar](http://www.unla.edu.ar)**

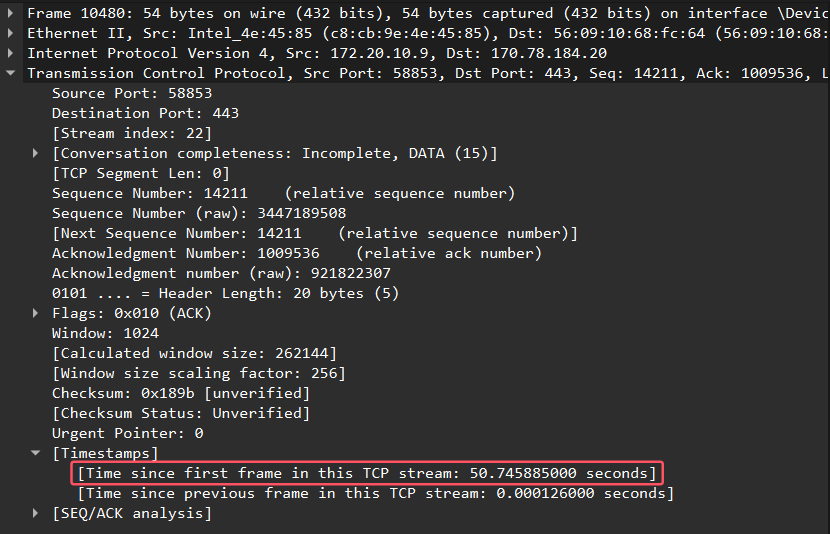
DNS查询和解析次数：5次



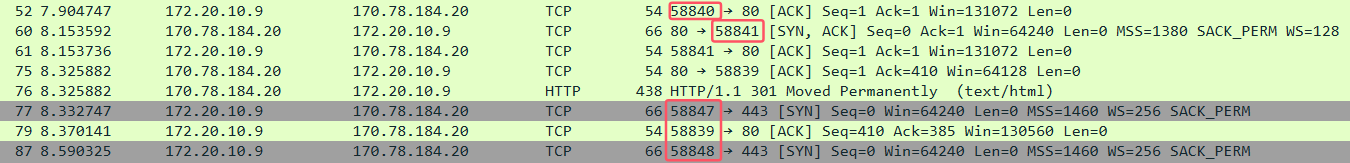
应答时间：0.3968s

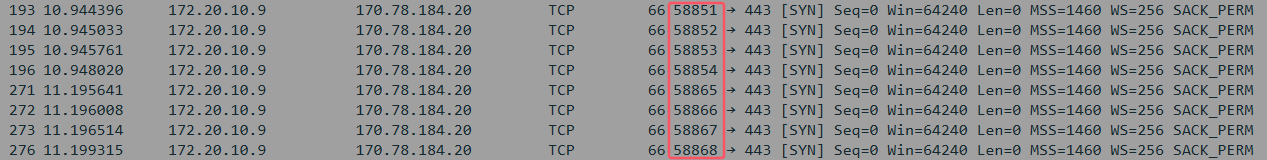


加载时间：50.7459s



建立连接数：13个，本地端口号为58839、58840、58841、58847、58848、58851、58852、58853、58854、58865、58866、58867、58868





请求对象数：5个，HTML、CSS、JS、JPG、PNG

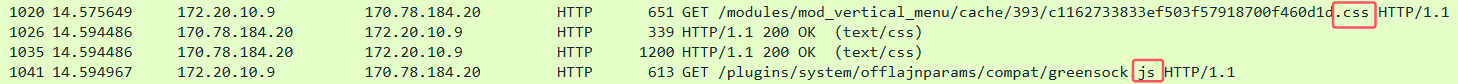




表5-1 不同网站访问信息统计表

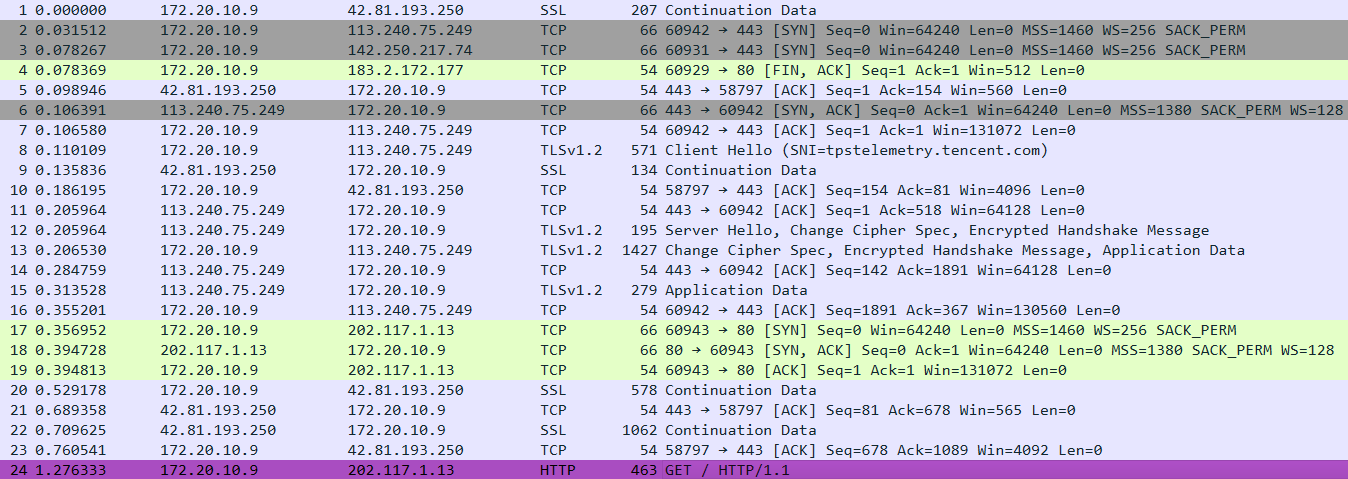
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 访问网站 | 国内 | | 国外 | | 南半球 | |
| 域名： | http://www.xjtu.edu.cn | | www.github.com | | www.unla.edu.ar | |
| IP地址： | 202.117.1.13 | | 20.205.243.166 | | 170.78.184.20 | |
| 是否带缓存 | 无缓存 | 带缓存 | 无缓存 | 带缓存 | 无缓存 | 带缓存 |
| DNS次数 | 3 | 0 | 3 | 0 | 5 | 1 |
| 应答时间(s) | 0.0855 | 0.0378 | 0.2028 | 0.2726 | 0.3968 | 0.3929 |
| 加载时间(s) | 5.9743 | 0.9423 | 44.8227 | 1.2038 | 50.7459 | 1.2173 |
| 建立连接数 | 8 | 7 | 4 | 2 | 13 | 9 |
| 请求对象数 | 8 | 5 | 2 | 0 | 5 | 0 |

带缓存的DNS和HTTP协议分析：

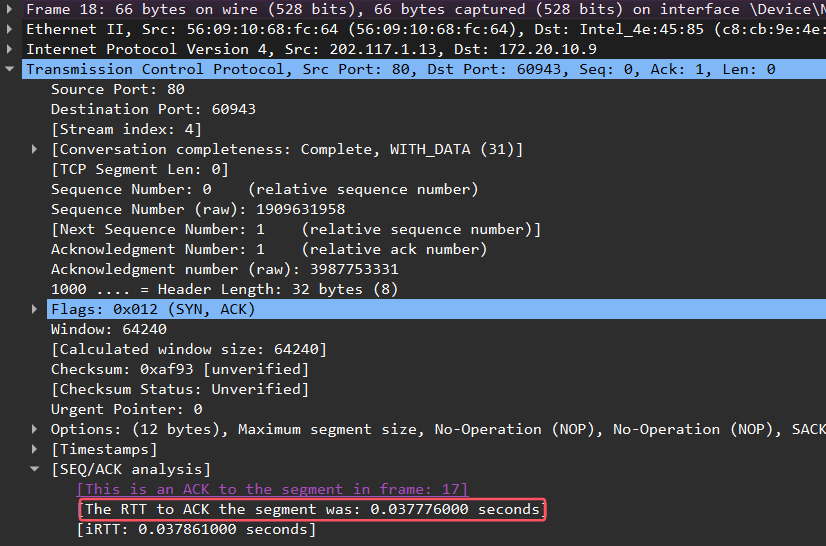
按上一节中的步骤1-4再次执行一遍，但不执行步骤2，填写表5-1中“带缓存”列信息。

**（1）国内网址：http://www.xjtu.edu.cn**

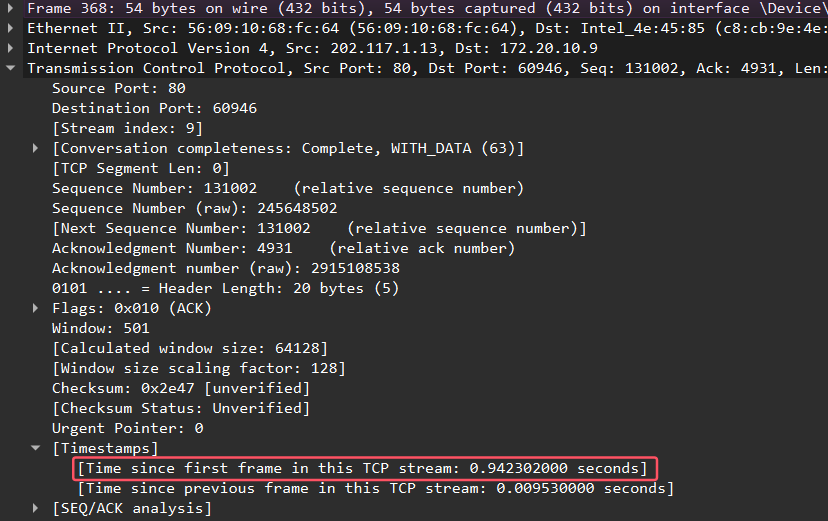
DNS查询解析次数：0



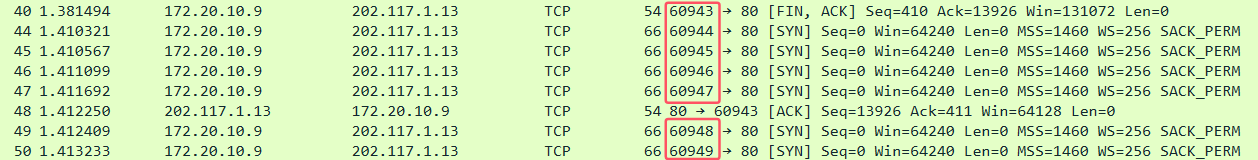
应答时间：0.0378s



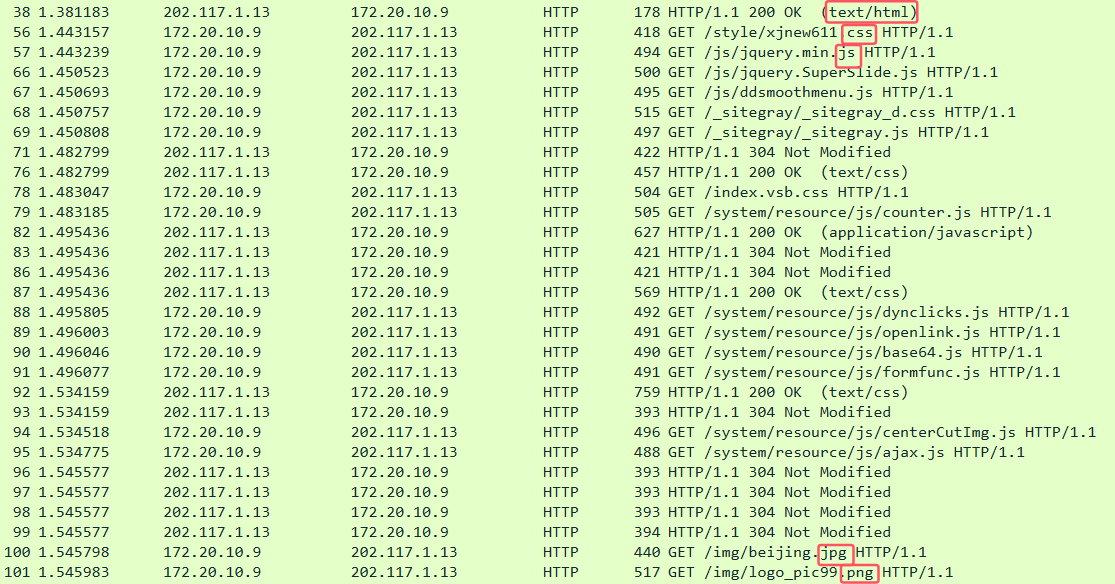
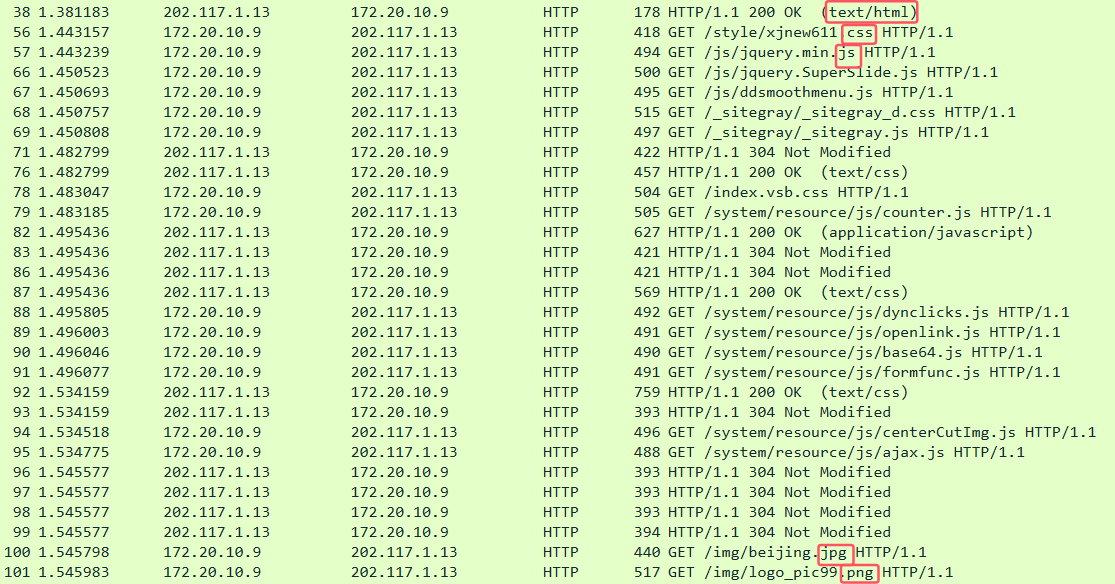
加载时间：0.9423s



建立连接数：7个，本地端口号从60943至60949

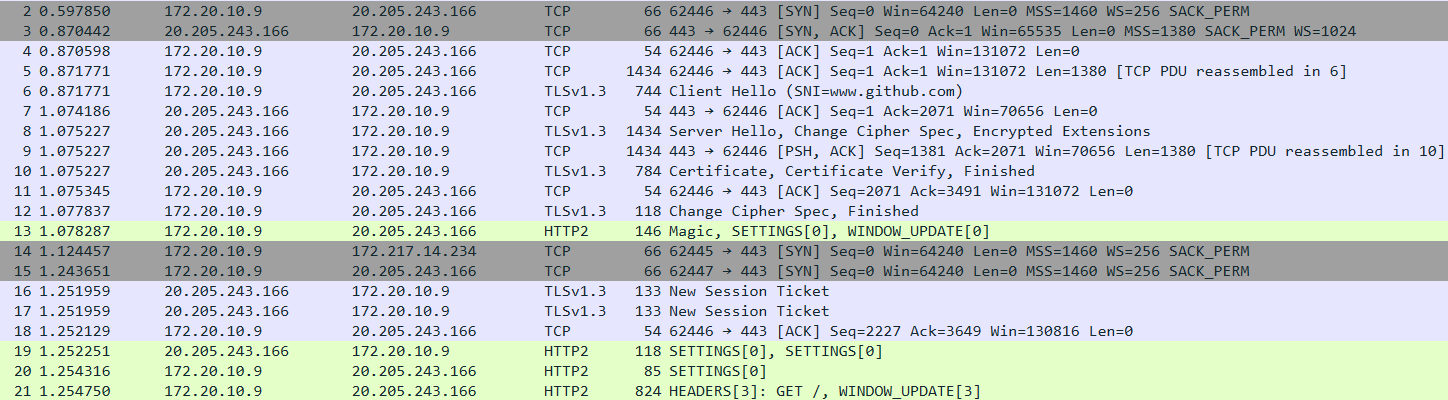


请求对象数：5个，HTML、CSS、JS、JPG、PNG

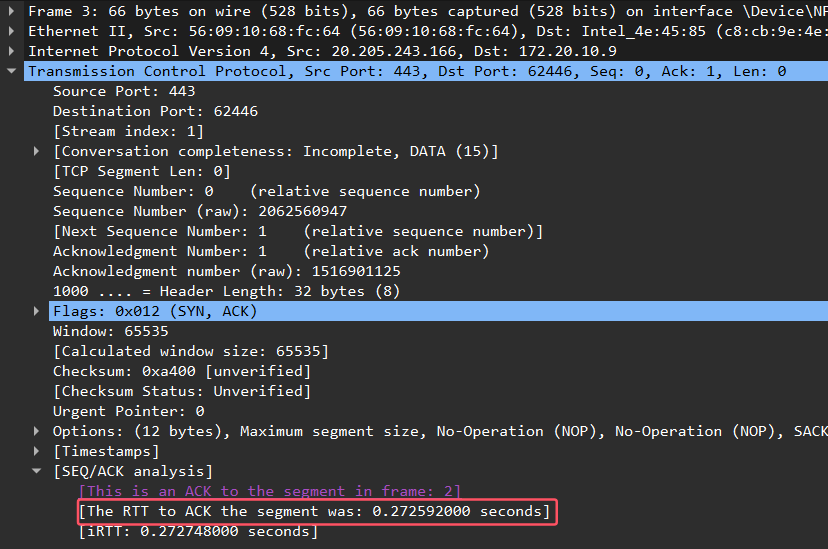


**（2）国外网址：[www.github.com](http://www.github.com)**

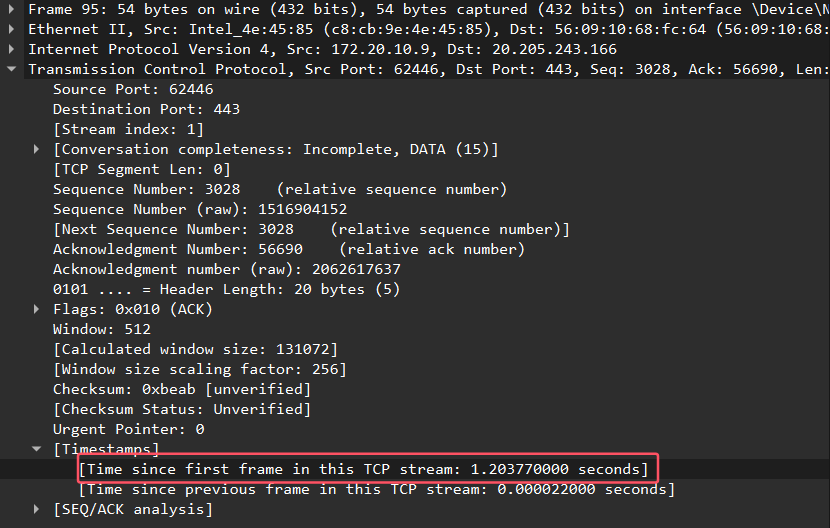
DNS查询和解析次数：0次



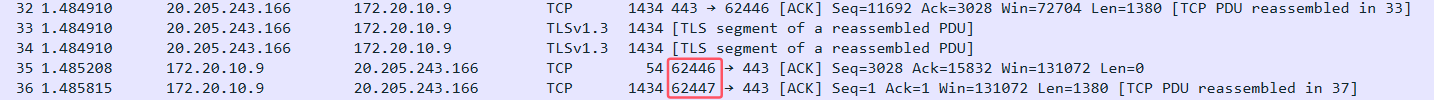
应答时间：0.2726s



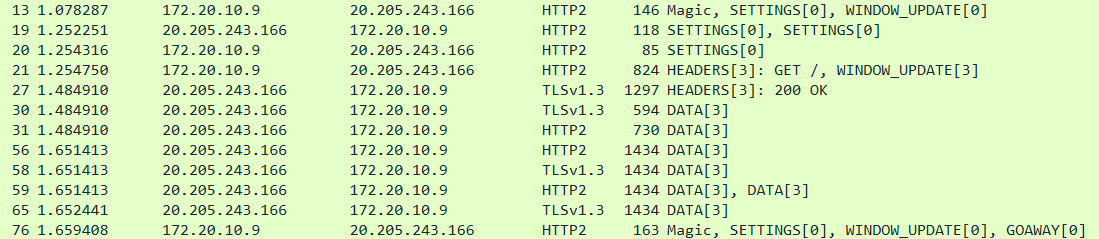
加载时间：1.2038s



建立连接数：2个，本地端口号为62446和62447

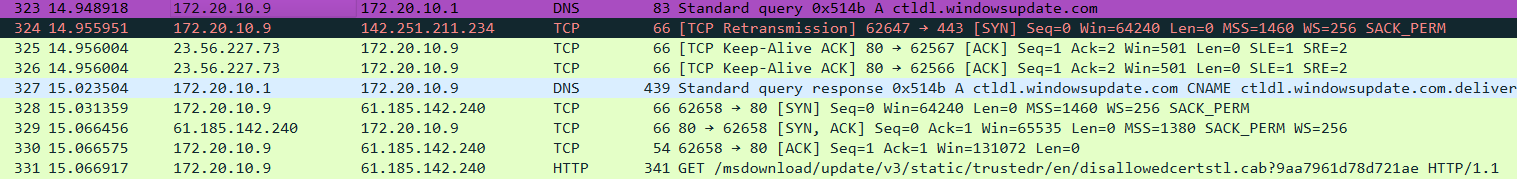


请求对象数：0

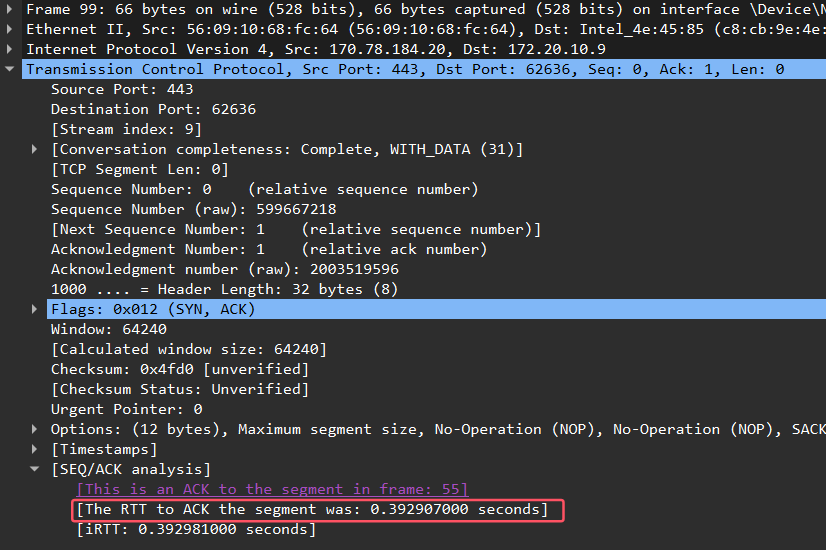


**（3）南半球网址：[www.unla.edu.ar](http://www.unla.edu.ar)**

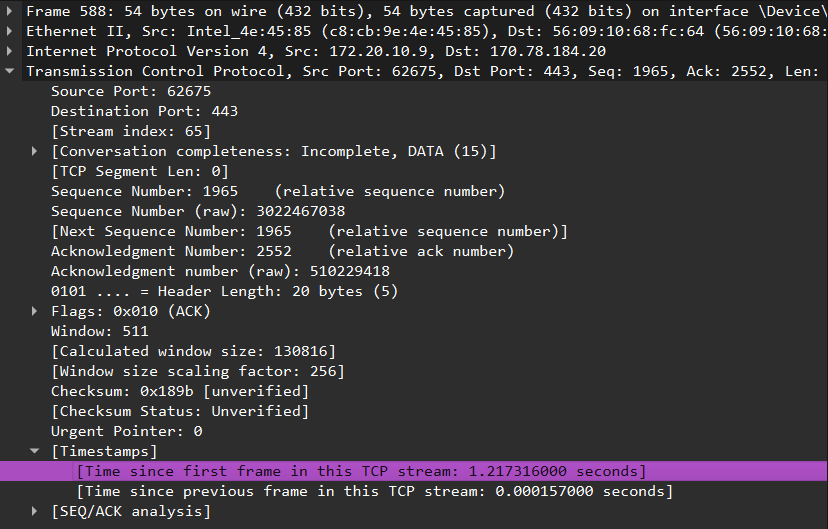
DNS查询和解析次数：1次



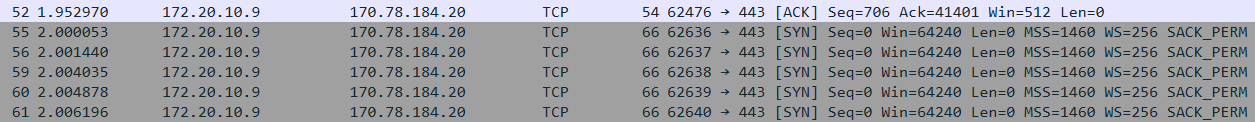
应答时间：0.3929s

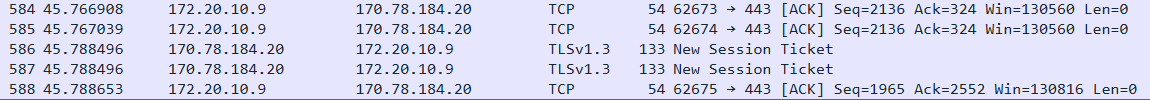


加载时间：1.2173s



建立连接数：9个，本地端口号62476、62636至62640、62673至62675





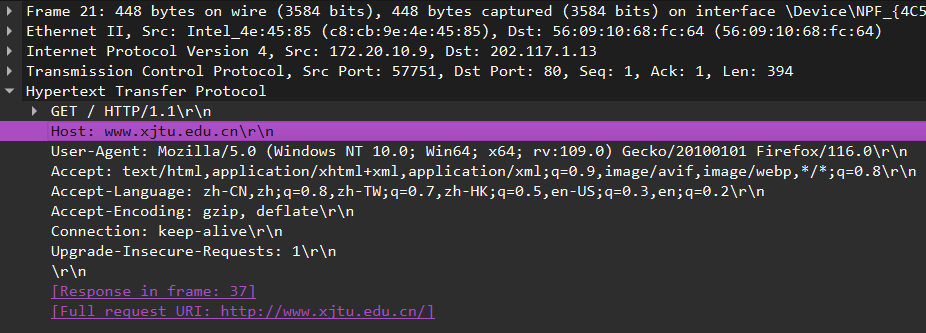
请求对象数：0



**HTTP分析：**

选中一条HTTP报文，点击右键，选择“追踪流->HTTP Stream”。关闭统计窗口，在主界面观察数据流，结合数据包分析HTTP的请求和应答，以及与TCP协议的配合使用。

HTTP请求报文：



第一行：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：请求将要发送到的服务器主机名为www.xjtu.edu.cn，端口号默认为80。

User-Agent：客户端运行的浏览器类型。

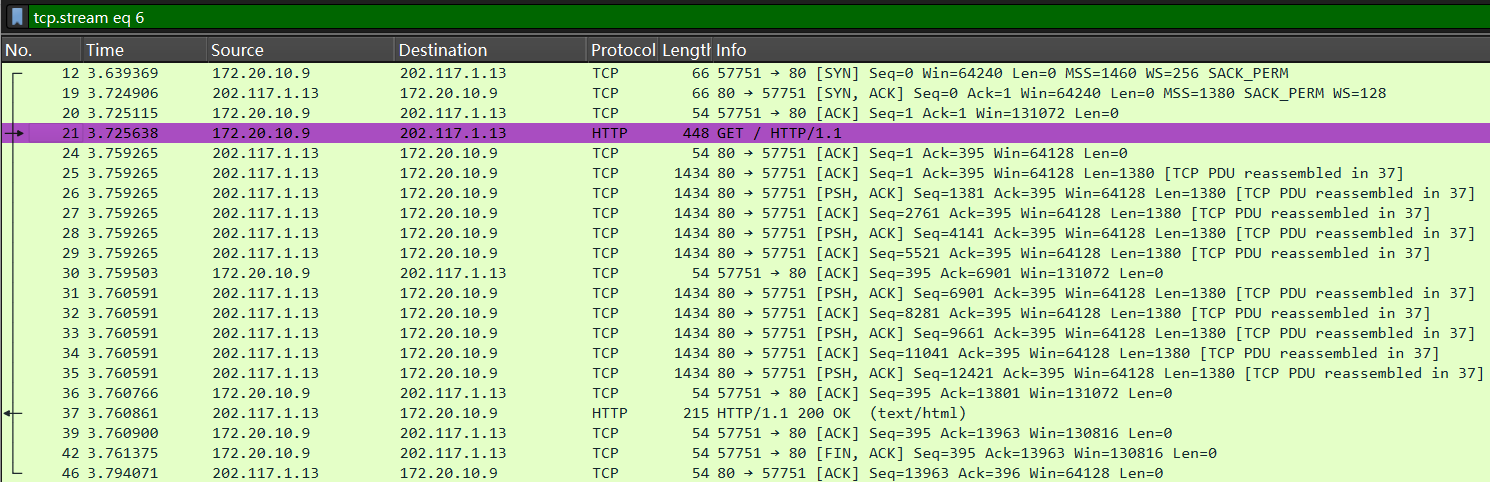
Accept：指定客户端能够接收的内容类型，text/html、xml、webp、image等。

Accept-Language：表明哪些语言客户端是能够理解，zh-CN表示简体中文。

Accept-Encoding：内容编码类型为gzip或deflate。

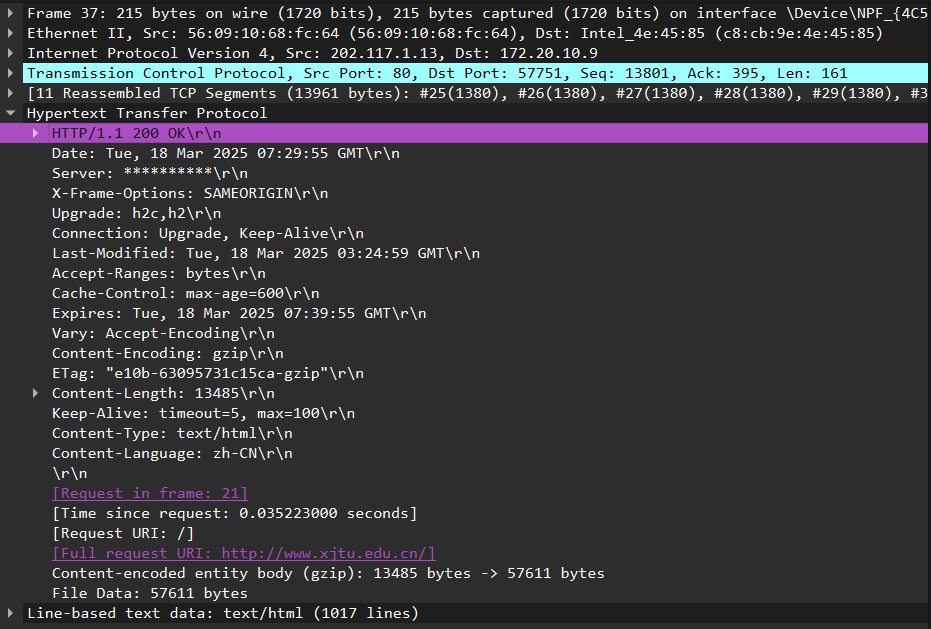
Connection：连接类型为keep-alive，表示网络连接是持久的。

Upgrade-Insecure-Requests：客户端对加密和认证响应的偏好。



当服务器收到客户端发来的HTTP请求报文后，通过TCP报文向客户端发送请求数据。服务器在传送完客户端请求的文件后，会向客户端发送一个HTTP应答报文，通知客户端数据已经传输完成。

HTTP应答报文：



第一行：HTTP协议版本号为1.1，状态码200，状态语句OK表示请求已成功。200响应默认是可缓存的。

Connection：连接类型为keep-alive，表示网络连接是持久的。

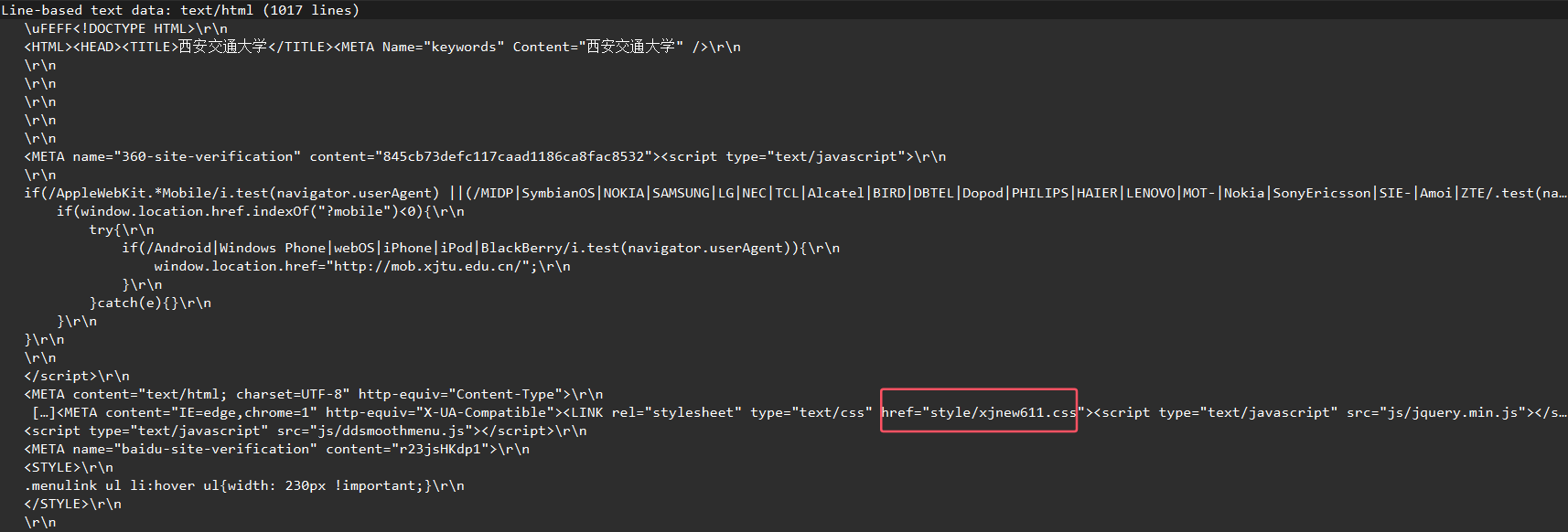
Content-Encoding：用于向接收方指示对象的类型，这里表示内容编码格式为gzip。

Content-Length：用于指明正文中返回对象的长度，13485bytes

Keep-Alive：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。Timeout=5空闲连接必须保持打开的最短时间为5秒，max=100再关闭它之前可以再次连接上发送的最大请求数为100。

Content-Type：用于向接收方指示对象的类型，text/html。

应答报文的主体部分：

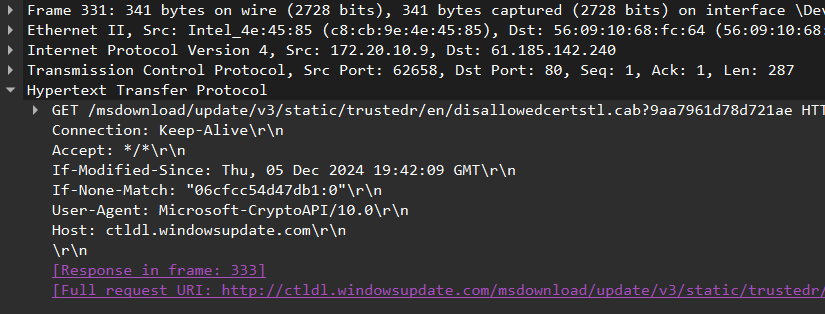


服务器返回了gzip编码的交大主页的HTML文件

找到HTTP应答包中状态码为304的数据包和对应的请求包，分析缓存工作原理和带来的好处。



HTTP请求报文：



客户端请求disallowedcertstl.cab文件，并附带一个查询参数?9aa7961d78d721ae。

请求头：

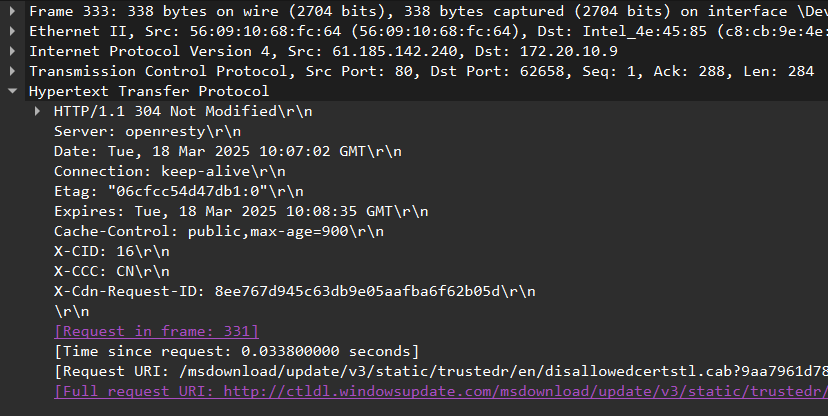
If-Modified-Since: Thu, 05 Dec 2024 19:42:09 GMT表示客户端缓存中该资源的最后修改时间。

If-None-Match: "06cfcc54d47db1:0"用于客户端缓存中该资源的ETag标识。

User-Agent: Microsoft-CryptoAPI/10.0客户端是Windows系统的加密API组件。

Host: ctldl.windowsupdate.com表示请求的目标主机。

HTTP应答报文：



服务器返回状态码304 Not Modified，表示客户端缓存的文件仍然有效，无需重新下载。

响应头：

Etag: "06cfcc54d47db1:0"用于服务器确认资源的ETag标识与客户端缓存一致。

Expires: Tue, 18 Mar 2025 10:08:35 GMT表示资源在缓存中的过期时间。

Cache-Control: public, max-age=900表示缓存控制策略，表示资源可以被公共缓存，且缓存有效期为900秒（15分钟）。

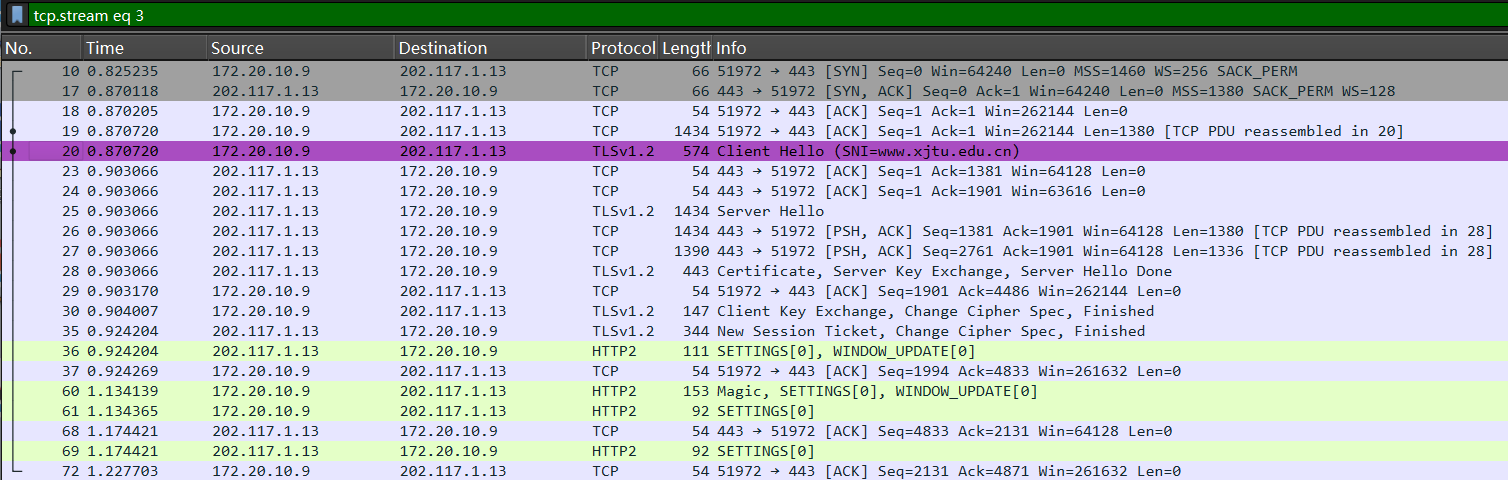
客户端缓存机制：客户端在首次请求资源时，会将资源保存到本地缓存中，并记录Last-Modified、ETag，当客户端再次请求同一资源时，会发送If-Modified-Since、If-None-Match等信息，来询问服务器资源是否已修改。

服务器验证机制：服务器收到请求后，会检查资源的Last-Modified时间和ETag标识。如果资源未修改，服务器返回304 Not Modified，通知客户端使用缓存。如果资源已修改，服务器返回200 OK和新资源内容。

因此，有缓存时，服务器无需重复传输相同资源，减少带宽消耗；客户端直接从本地缓存加载资源，避免了网络传输延迟，加快加载速度；服务器无需处理重复的资源请求，降低服务器负载；页面加载速度更快。

**HTTPS分析：**

访问HTTPS网站时，因数据是加密传输无法观察内容，可观察分析TLS加密传输的建立过程、传输端口等（选中一条TLS数据报文，点击右键，选择“追踪流->TLS Stream”）。



**TLS加密传输的建立过程**

1. 客户端发起握手

客户端向服务器发起TLS握手请求Client Hello，并发送支持的TLS版本、加密套件列表、随机数等信息。

SNI表示客户端想要访问的服务器域名是www.xjtu.edu.cn。SNI用于在同一个IP地址上托管多个域名时，帮助服务器选择正确的证书。

1. 服务器响应

服务器响应客户端的Client Hello，发送Server Hello消息，服务器选择TLSv1.2作为通信协议，并确定了加密套件，并且生成一个随机数，用于后续的密钥交换。

1. 服务器发送证书和密钥交换信息

Certificate证书中包含了服务器的公钥和域名信息。

Server Key Exchange是服务器发送的密钥交换信息，用于后续的密钥协商。

Server Hello Done表示服务器已经完成了握手的第一阶段，等待客户端响应。

1. 客户端密钥交换和加密切换

Client Key Exchange表示客户端生成一个预主密钥，并使用服务器的公钥加密后发送给服务器。

Change Cipher Spec表示客户端将切换到加密通信模式。

Finished消息是客户端发送的第一个加密消息，用于验证握手过程的完整性。

1. 服务器完成握手

New Session Ticket用于支持会话恢复，以便在未来的连接中减少握手时间。

Change Cipher Spec表示服务器也将切换到加密通信模式。

Finished消息是服务器发送的第一个加密消息，用于验证握手过程的完整性。

**传输端口：**

本地端口号为51972，TLS/HTTPS的默认端口为443。

### 3.3.3 Ncat访问HTTP

Nmap(Network Mapper)是一款功能强大的开源网络扫描工具，其包含的工具Ncat可以用于网络连接、调试和数据传输。Nmap软件解压后，在cmd命令行进入目录，就可以执行命令，也可以将软件目录添加至系统环境变量。

前面学习了HTTP协议，也用Wireshark分析了典型的HTTP请求和应答。现在使用ncat来**模拟**浏览器来跟服务器进行HTTP交互。在命令窗口执行“ncat -C 域名/ip地址 端口（80）”连接HTTP服务器（如www.xjtu.edu.cn），输入HTTP请求（可参考之前捕获的数据包中的请求），请求头至少有Host和User-Agent，其中**User-Agent后跟上自己的英文名字**。查看服务器的应答。

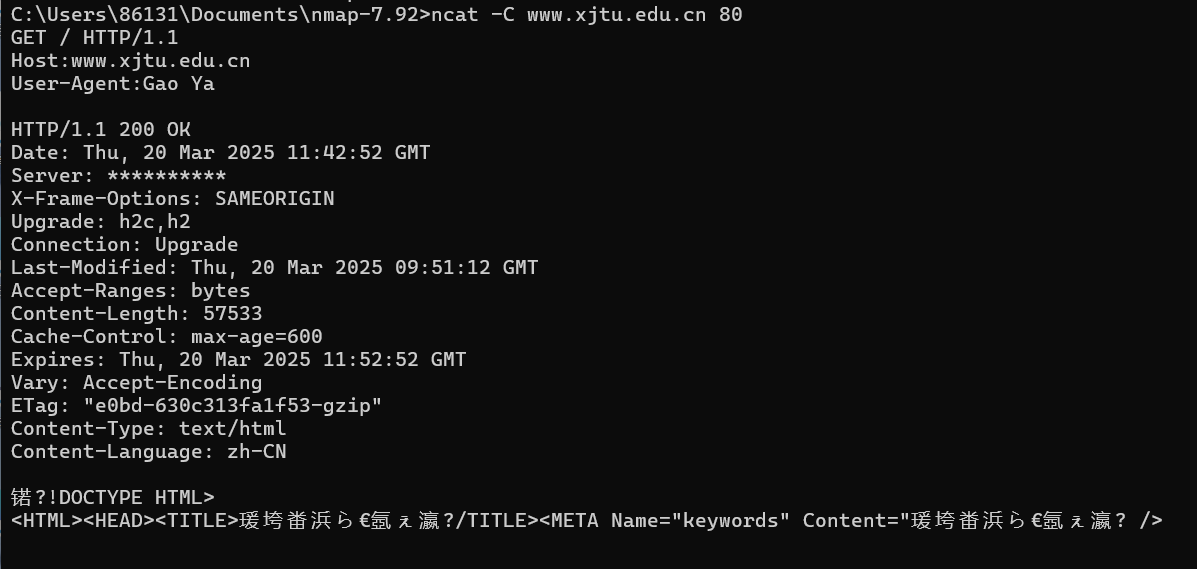
例：

ncat -C **xxx.xxx.xxx.xxx** 80 **（xxx.xxx.xxx.xxx是具体的域名或IP地址）**

GET / HTTP/1.1

Host: **xxx.xxx.xxx.xxx（xxx.xxx.xxx.xxx是具体的域名。）**

User-Agent: Zhang San （**跟上自己的英文名字。**然后敲两次回车，应答状态码200表示请求成功）



服务器成功应答

### 3.3.4 编写HTTP客户端程序

（1）编写一个简单的HTTP客户端程序。建议使用Python的socket模块，但不能使用Python的高级HTTP模块（如requests、urllib.request、http.client等）。

（2）运行TCP客户端程序进行连接测试（如连接到www.xjtu.edu.cn或www.hit.edu.cn服务器80端口，获取并打印主页）。

输入HTTP请求时，除了请求行以外，请求头至少有Host和User-Agent，其中**User-Agent后跟上自己的英文名字**。

程序代码：

import socket

def http\_client():

    host = "www.xjtu.edu.cn"

    port = 80

    request\_line = "GET / HTTP/1.1\r\n"

    headers = (

        f"Host: {host}\r\n"

        f"User-Agent: Gao Ya\r\n"

        "Connection: close\r\n"

        "\r\n"

    )

    request = request\_line + headers

    with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as client\_socket:

        client\_socket.connect((host, port))

        client\_socket.sendall(request.encode())

        response = b""

        while True:

            data = client\_socket.recv(4096)

            if not data:

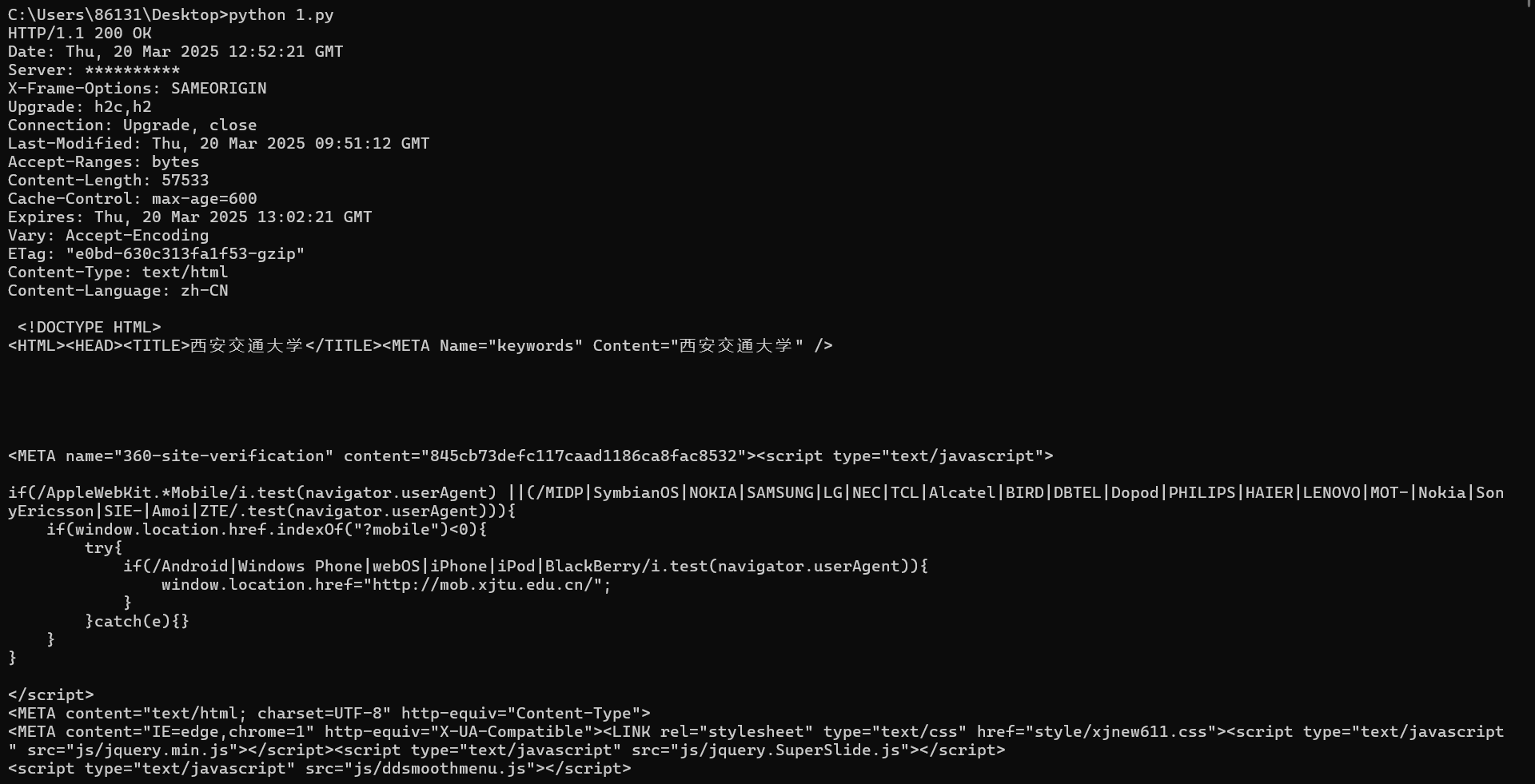
                break

            response += data

    print(response.decode(errors='ignore'))

if \_\_name\_\_ == "\_\_main\_\_":

    http\_client()





## 3.4 互动讨论主题

（1）HTTP协议的缓存，DNS的缓存；

**HTTP协议缓存机制：**

HTTP缓存分为两大类：强制缓存和协商缓存。两类缓存规则不同，强制缓存在缓存数据未失效的情况下，不需要与服务器交互；而协商缓存需要与服务器进行比较判断是否可以使用缓存。两类缓存规则可以同时存在，强制缓存的优先级高于协商缓存。

强制缓存（无条件信任）：采用Expires和Cache-Control双重控制策略

Expires：服务器告诉浏览器的缓存过期时间（值为GMT时间）。如果浏览器端的当前时间未到达过期时间，则直接使用缓存数据。

Cache-Control：服务器告诉浏览器一个相对时间（如Cache-Control=10秒，表示在10秒内直接使用浏览器缓存）。

强制缓存仅根据时间判断缓存是否过期。如果文件未改动但缓存已过期，再次获取文件会浪费服务器资源。

协商缓存（条件验证）：通过两组报文实现精准校验

时间戳校验：Last-Modified和If-Modified-Since，浏览器和服务器协商，服务器每次返回文件的同时，告诉浏览器文件在服务器上最近的修改时间。

内容校验：ETag和If-None-Match，为了解决文件修改时间不精确带来的问题，服务器和浏览器再次协商，这次不返回时间，返回文件的唯一标识ETag。只有当文件内容改变时，ETag才改变。

**DNS缓存：**

DNS缓存机制存在四级缓存架构：浏览器缓存、操作系统缓存、路由器缓存、递归DNS服务器缓存。

当输入域名时优先检查浏览器缓存，如果未命中则查询操作系统，再未命中转向路由器。若本地全未命中，向递归DNS服务器发起请求。递归服务器从根DNS开始逐级查询，最终获得IP地址并逐级回传缓存。

1. 缓存对网络访问速度的影响。

缓存机制可以在重复访问时直接从内存读取缓存，避免完整HTTP请求流程，大量减少了客户端请求与服务器响应时间，提升了网络访问速度。

但强制缓存期间用户无法获取最新版本，并且有可能低频访问的资源长期占用缓存空间。

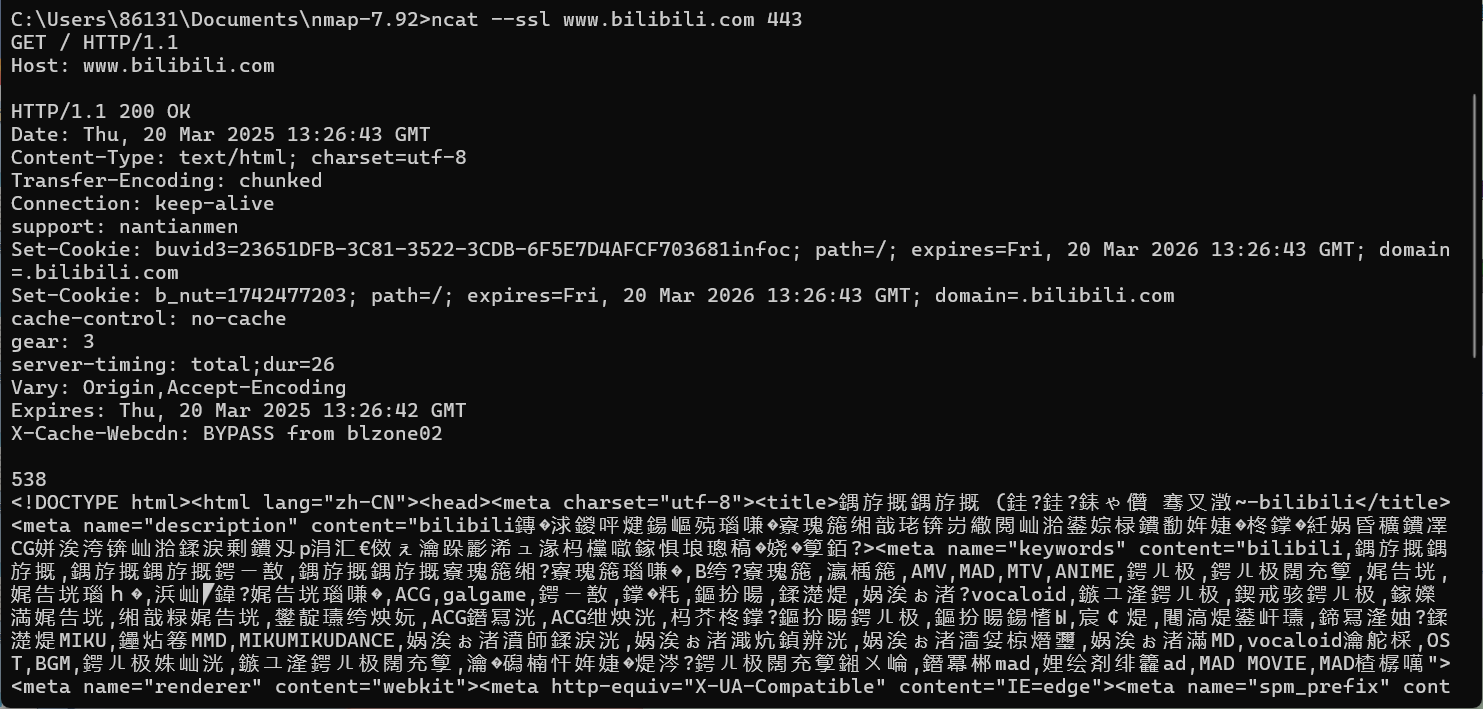
## 3.5 进阶自设计

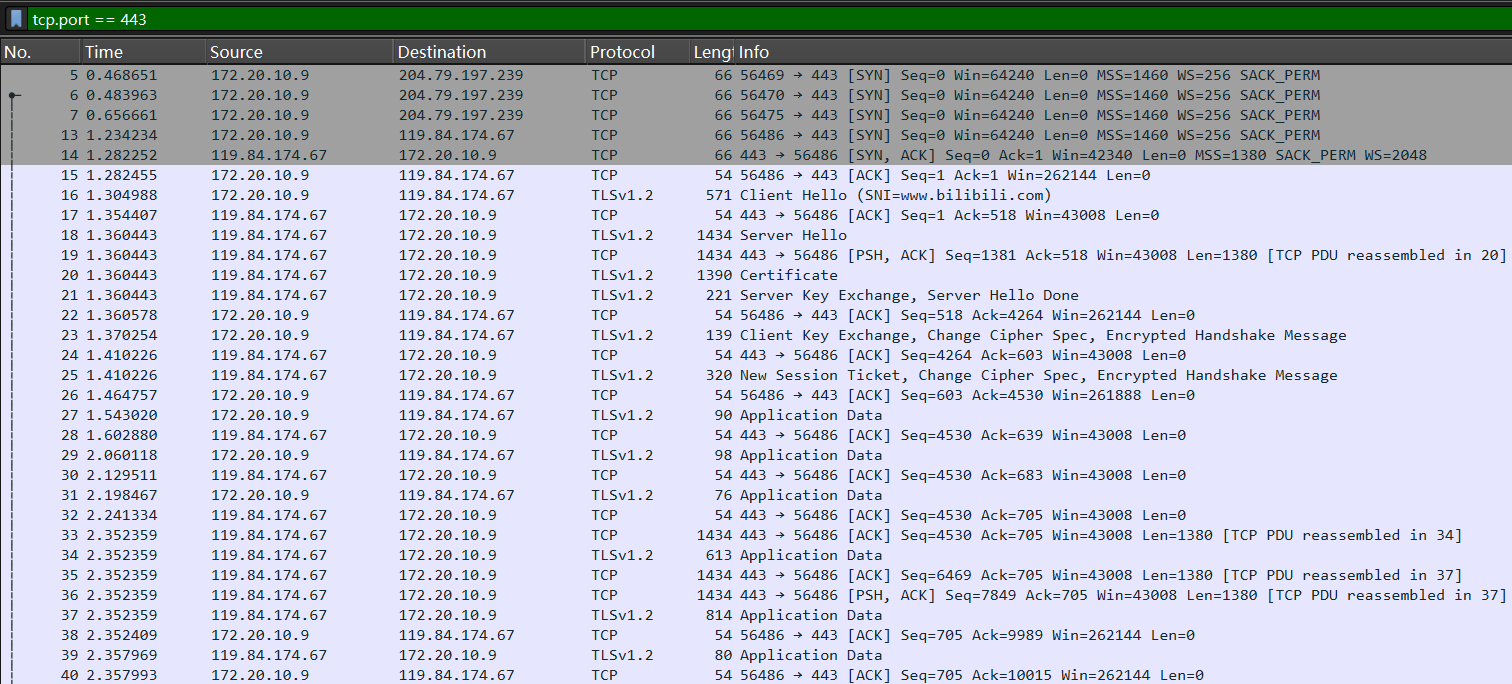
用nmap的ncat来模拟https客户端，访问1-2个网站，抓包分析访问过程。

ncat --ssl www.bilibili.com 443

GET / HTTP/1.1

Host: www.bilibili.com





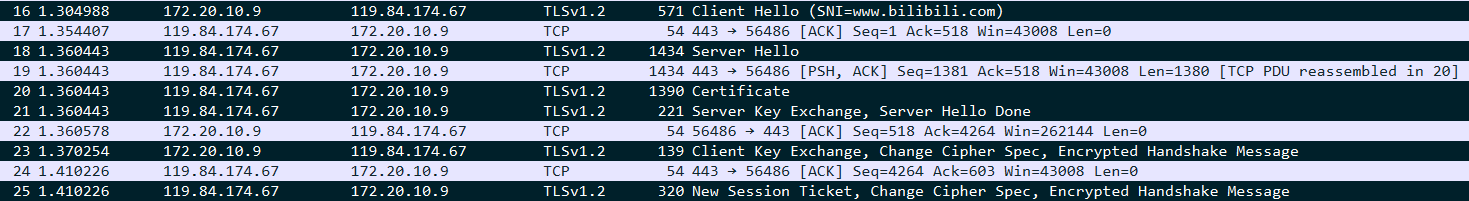
访问过程：

1. TCP三次握手



客户端向www.example.com服务器发起TCP连接

1. TLS握手



抓包中可以看到完整的TLS握手过程，包括Client Hello、Server Hello、证书交换、密钥交换、预主密钥等。

1. HTTP 请求和响应

HTTP报文在传输过程中是加密的，wireshark只能看到加密的数据流，不能直接解析出明文HTTP报文。