

实验四、应用层协议分析实验报告

组号：7-1		
姓名：白佳兴	学号：2204311549	班级：计算机2105
姓名：廖立彬	学号：2213611635	班级：计算机2105

一、实验目的

分析应用层协议（如FTP，HTTP）的工作过程，理解应用层与传输层及下层协议的关系。

二、实验内容

- （1）每组同学利用现有实验室网络及云服务器搭建内网、外网环境；
- （2）用Wireshark截获HTTP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。分析应用层协议跟TCP/DNS等协议的交互关系。
- （3）用Wireshark截获FTP的报文，分析FTP协议的连接；分析被动模式，普通模式的区别；分析NAT对FTP的影响。使用netcat工具模拟FTP的客户端。

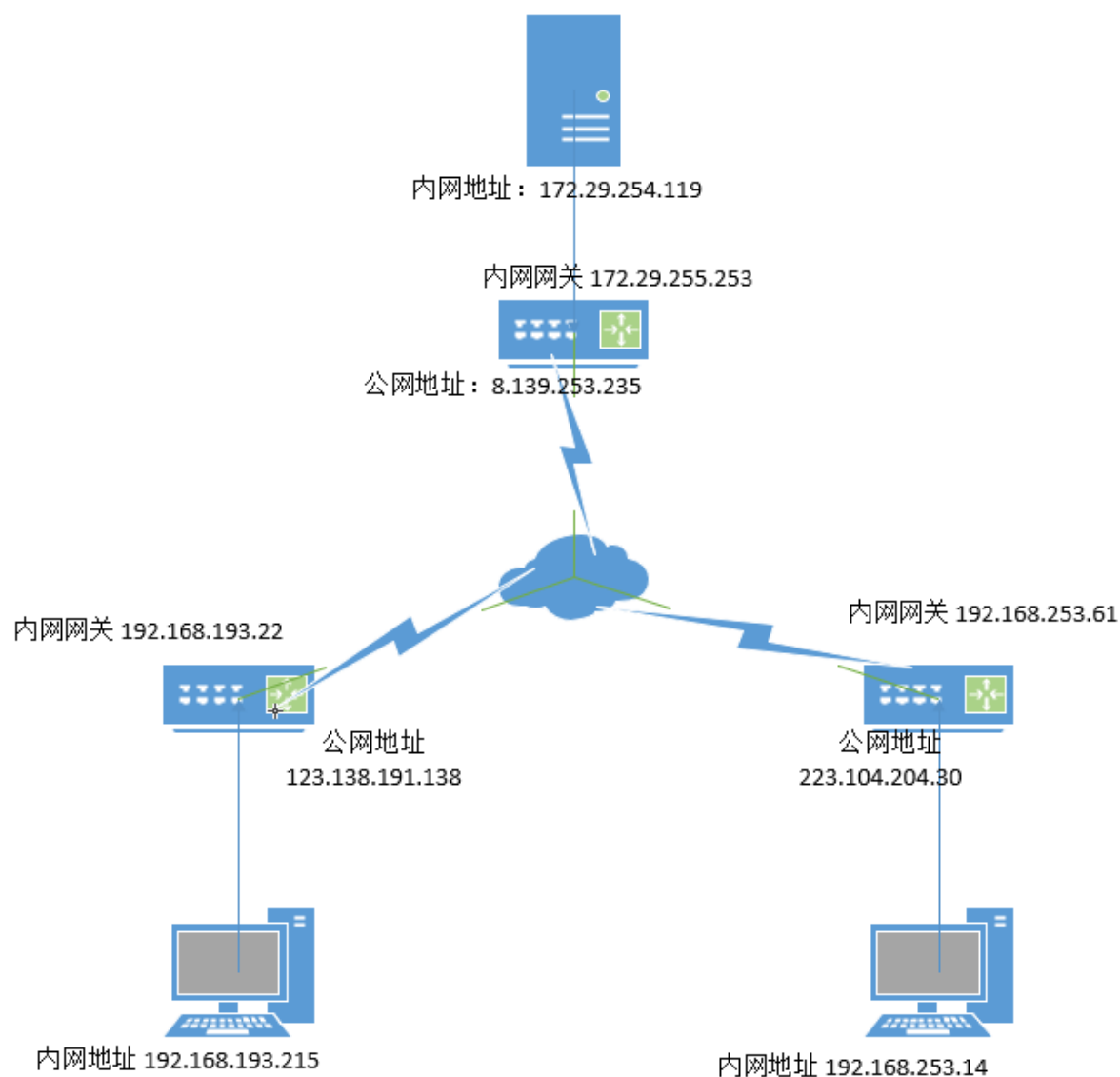
注：HTTP和FTP两个协议二选一。

三、实验环境与分组

每2名同学一组，以现有校园网络环境及云服务器搭建内网、外网网络。

四、实验组网

以各组现有网络实际情况为准，标注内网、公网地址。



五、 实验过程及结果分析

【过程记录应当详尽，截图并加以说明。以下过程和表格仅供参考。】

1、HTTP协议分析

（一）清空缓存后的ARP，DNS和HTTP协议分析

步骤1：在计算机终端上运行Wireshark截获所有的报文。

步骤2：清空ARP，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。

执行“ipconfig /flushdns”清除本地DNS缓存。

执行“arp -d”命令清空arp缓存。

注：如果arp命令无法运行，可使用以下命令代替：

```
netsh interface IP show neighbors
```

```
netsh interface IP delete arpcache
```

答：清除缓存

```
C:\WINDOWS\system32>ipconfig /flushdns

Windows IP 配置

已成功刷新 DNS 解析缓存。

C:\WINDOWS\system32>arp -d

C:\WINDOWS\system32>
```

步骤3: 在浏览器中访问3个网址, 比如www.xjtu.edu.cn, www.github.com, www.unb.br;

步骤4: 执行完之后, Wireshark停止报文截获, 分析截获的报文。

观察几个协议的配合使用, 注意访问的延迟情况。特别分析HTTP的请求和应答。注意一个网址的访问中:

1. 有几次DNS解析;

答:

访问 www.xjtu.edu.cn 时, 只需要一次DNS解析就可以得到相应IP

13	1.820843	10.172.147.69	221.11.1.67	DNS	75	Standard query 0x91e2 A www.xjtu.edu.cn
15	1.826778	221.11.1.67	10.172.147.69	DNS	91	Standard query response 0x91e2 A www.xjtu.edu.cn A 202.117.1.13

访问 www.github.com 时, 需要两次DNS解析来获取相应IP

1734	10.274402	10.172.147.69	221.11.1.67	DNS	74	Standard query 0xca9d A www.github.com
1736	10.281372	221.11.1.67	10.172.147.69	DNS	104	Standard query response 0xca9d A www.github.com CNAME github.com A 20.205.243.166
1784	10.679463	10.172.147.69	221.11.1.67	DNS	70	Standard query 0x488f A github.com
1785	10.700521	221.11.1.67	10.172.147.69	DNS	86	Standard query response 0x488f A github.com A 20.205.243.166

访问 www.unb.br 时, 需要一次DNS解析来获取相应IP

3061	18.059395	10.172.147.69	221.11.1.67	DNS	66	Standard query 0x8e8e A unb.br
3062	18.064564	221.11.1.67	10.172.147.69	DNS	82	Standard query response 0x8e8e A unb.br A 164.41.102.70

2. 用了几个连接 (建立连接时, 本地的端口不同, HTTP服务常用端口是80, HTTPS服务常用端口是443) ;

答:

通过查询SYN报文的数量, 即可以确定连接的个数, 这里使用 `tcp.flags.syn == 1 and tcp.flags.ack == 0` 来过滤SYN报文

202.117.1.13为 www.xjtu.edu.cn 的IP地址, 可以看到用了7个TCP连接

ip.addr == 202.117.1.13 and tcp.flags.syn == 1 and tcp.flags.ack == 0						
No.	Time	Source	Destination	Protocol	Length	Info
17	1.833154	10.172.147.69	202.117.1.13	TCP	66	58315 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
18	1.833533	10.172.147.69	202.117.1.13	TCP	66	58316 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
21	1.837078	10.172.147.69	202.117.1.13	TCP	66	58317 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
118	8.870867	10.172.147.69	202.117.1.13	TCP	66	58334 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
119	8.871241	10.172.147.69	202.117.1.13	TCP	66	58335 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
134	8.879886	10.172.147.69	202.117.1.13	TCP	66	58336 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
135	8.880080	10.172.147.69	202.117.1.13	TCP	66	58337 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

20.205.243.166为 www.github.com 的IP地址, 用了3个TCP连接

ip.addr == 20.205.243.166 and tcp.flags.syn == 1 and tcp.flags.ack == 0						
No.	Time	Source	Destination	Protocol	Length	Info
1737	10.283986	10.172.147.69	20.205.243.166	TCP	66	58344 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
1738	10.290640	10.172.147.69	20.205.243.166	TCP	66	58345 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
1739	10.291047	10.172.147.69	20.205.243.166	TCP	66	58346 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM

164.41.102.70为 www.unb.br 的IP地址, 用了16个TCP连接

ip.addr == 164.41.102.70 and tcp.flags.syn == 1 and tcp.flags.ack == 0							
No.	Time	Source	Destination	Protocol	Length	Info	
2858	14.593588	10.172.147.69	164.41.102.70	TCP	66	58373 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
2859	14.593878	10.172.147.69	164.41.102.70	TCP	66	58374 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
2860	14.593994	10.172.147.69	164.41.102.70	TCP	66	58375 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
2912	14.851534	10.172.147.69	164.41.102.70	TCP	66	58378 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3063	18.065682	10.172.147.69	164.41.102.70	TCP	66	58382 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3068	18.331161	10.172.147.69	164.41.102.70	TCP	66	58383 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3112	20.437325	10.172.147.69	164.41.102.70	TCP	66	58385 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3113	20.437538	10.172.147.69	164.41.102.70	TCP	66	58386 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3115	20.439393	10.172.147.69	164.41.102.70	TCP	66	58387 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3116	20.440004	10.172.147.69	164.41.102.70	TCP	66	58388 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3731	42.620777	10.172.147.69	164.41.102.70	TCP	66	58441 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3732	42.620898	10.172.147.69	164.41.102.70	TCP	66	58442 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3733	42.620984	10.172.147.69	164.41.102.70	TCP	66	58443 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3734	42.621077	10.172.147.69	164.41.102.70	TCP	66	58444 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3735	42.621179	10.172.147.69	164.41.102.70	TCP	66	58445 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	
3736	42.633934	10.172.147.69	164.41.102.70	TCP	66	58446 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM	

3. 取了几个对象（GET的对象，如：HTML，CSS，JS，图片等）；

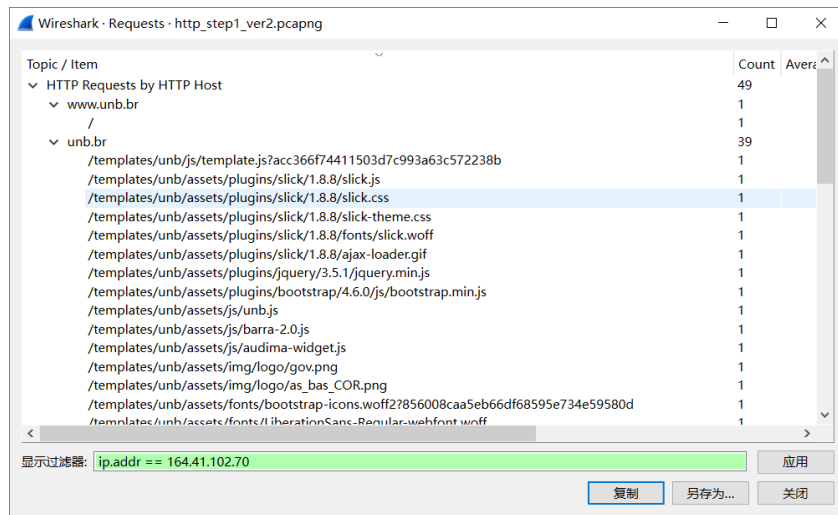
- `www.xjtu.edu.cn`：访问 `www.xjtu.edu.cn` 时一共有79个GET请求，即取了79个对象

http.request.method == GET and ip.addr == 202.117.1.13							
No.	Time	Source	Destination	Protocol	Length	Info	
21	0.2148024	192.168.222.215	202.117.1.13	HTTP	493	GET / HTTP/1.1	
678	4.223028	192.168.222.215	202.117.1.13	HTTP	398	GET /style/xjnew611.css HTTP/1.1	
680	4.229111	192.168.222.215	202.117.1.13	HTTP	381	GET /js/jquery.min.js HTTP/1.1	
708	4.243447	192.168.222.215	202.117.1.13	HTTP	388	GET /js/jquery.SuperSlide.js HTTP/1.1	
1053	4.539276	192.168.222.215	202.117.1.13	HTTP	405	GET /_silegray/_silegray_d.css HTTP/1.1	
1076	4.550912	192.168.222.215	202.117.1.13	HTTP	393	GET /index.vsb.css HTTP/1.1	
1077	4.555168	192.168.222.215	202.117.1.13	HTTP	387	GET /_silegray/_silegray.js HTTP/1.1	
1155	4.764906	192.168.222.215	202.117.1.13	HTTP	394	GET /system/resource/js/counter.js HTTP/1.1	
1156	4.765268	192.168.222.215	202.117.1.13	HTTP	396	GET /system/resource/js/dynclinks.js HTTP/1.1	
1158	4.766015	192.168.222.215	202.117.1.13	HTTP	444	GET /img/logo_pic09.png HTTP/1.1	
1162	4.775209	192.168.222.215	202.117.1.13	HTTP	395	GET /system/resource/js/openlink.js HTTP/1.1	
1174	4.789909	192.168.222.215	202.117.1.13	HTTP	443	GET /images/search.png HTTP/1.1	
1175	4.793663	192.168.222.215	202.117.1.13	HTTP	393	GET /system/resource/js/base64.js HTTP/1.1	
1177	4.795819	192.168.222.215	202.117.1.13	HTTP	395	GET /system/resource/js/formfunc.js HTTP/1.1	
1185	4.816181	192.168.222.215	202.117.1.13	HTTP	383	GET /js/ddsmoothmenu.js HTTP/1.1	
1186	4.818584	192.168.222.215	202.117.1.13	HTTP	468	GET /images/14/12/13/1tf5znrw9c/20231118011.jpg HTTP/1.1	
1187	4.819282	192.168.222.215	202.117.1.13	HTTP	458	GET /images/24/1200X320-3.jpg HTTP/1.1	
1280	4.841681	192.168.222.215	202.117.1.13	HTTP	450	GET /images/24/1200X320-2.jpg HTTP/1.1	
1421	5.151228	192.168.222.215	202.117.1.13	HTTP	459	GET /img/beijing.jpg HTTP/1.1	
1626	5.341247	192.168.222.215	202.117.1.13	HTTP	460	GET /img/in_xn_28.png HTTP/1.1	
1770	6.400840	192.168.222.215	202.117.1.13	HTTP	488	GET /_local/1/4/88/57557CEB1C3A8747844263CDE_5C77AE4_70A95.png HTTP/1.1	
1771	6.108562	192.168.222.215	202.117.1.13	HTTP	467	GET /images/navigation.png HTTP/1.1	
1772	6.108774	192.168.222.215	202.117.1.13	HTTP	460	GET /img/in_xn_18.png HTTP/1.1	
1792	6.165183	192.168.222.215	202.117.1.13	HTTP	438	GET /img/zy01.png HTTP/1.1	
1793	6.165496	192.168.222.215	202.117.1.13	HTTP	438	GET /img/zy02.png HTTP/1.1	
1824	6.231477	192.168.222.215	202.117.1.13	HTTP	438	GET /img/zy03.png HTTP/1.1	
1825	6.231737	192.168.222.215	202.117.1.13	HTTP	438	GET /img/zy04.png HTTP/1.1	
1873	6.288035	192.168.222.215	202.117.1.13	HTTP	446	GET /images/icon_news.gif HTTP/1.1	
1875	6.288468	192.168.222.215	202.117.1.13	HTTP	450	GET /images/24/1200X320-1.jpg HTTP/1.1	
1926	6.344148	192.168.222.215	202.117.1.13	HTTP	448	GET /images/24/1200X320.jpg HTTP/1.1	
2346	6.903468	192.168.222.215	202.117.1.13	HTTP	451	GET /images/24/20240402002.jpg HTTP/1.1	
2349	6.904941	192.168.222.215	202.117.1.13	HTTP	451	GET /images/24/20240327001.png HTTP/1.1	
2409	6.909296	192.168.222.215	202.117.1.13	HTTP	451	GET /images/24/20240315001.png HTTP/1.1	
2446	7.020535	192.168.222.215	202.117.1.13	HTTP	468	GET /images/24/03/12/sexqzn2r4/20240312002.jpg HTTP/1.1	
2556	7.130445	192.168.222.215	202.117.1.13	HTTP	438	GET /img/zy05.png HTTP/1.1	
2622	7.182985	192.168.222.215	202.117.1.13	HTTP	488	GET /_local/E/7C/B3/8F84CEB7820C65E2AEABCD0480C_21AE88A6_1CFFC.jpg HTTP/1.1	
2753	7.349532	192.168.222.215	202.117.1.13	HTTP	495	GET /_local/A/75/4F/A3001F4E08068A3E9D569B7C6B_0483CE80_3D94A.one?w= HTTP/1.1	

- `www.github.com`：使用条件 `http.request.method == GET` and `ip.addr == 20.205.243.166` 过滤时，并未发现有向 `www.github.com` 发送的GET请求

http.request.method == GET and ip.addr == 20.205.243.166						
No.	Time	Source	Destination	Protocol	Length	Info

- `www.unb.br`：共有49个GET请求

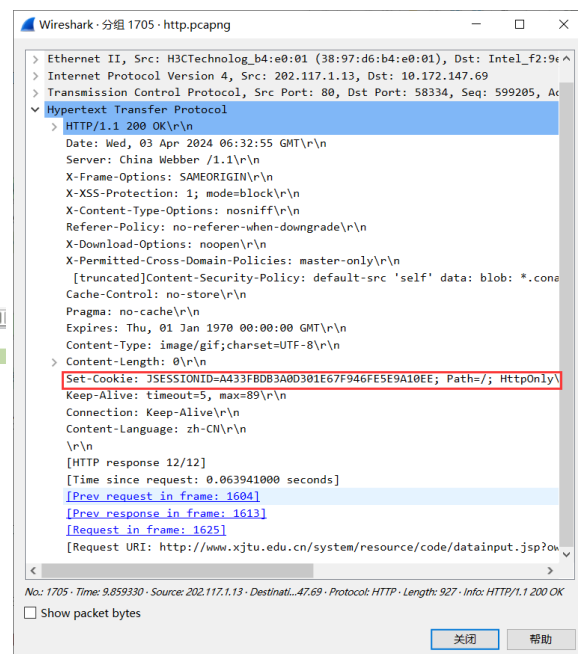


4. 有没有Cookie及其工作过程（Set-Cookie、Cookie、304 Not Modified相关数据包）等；

答：

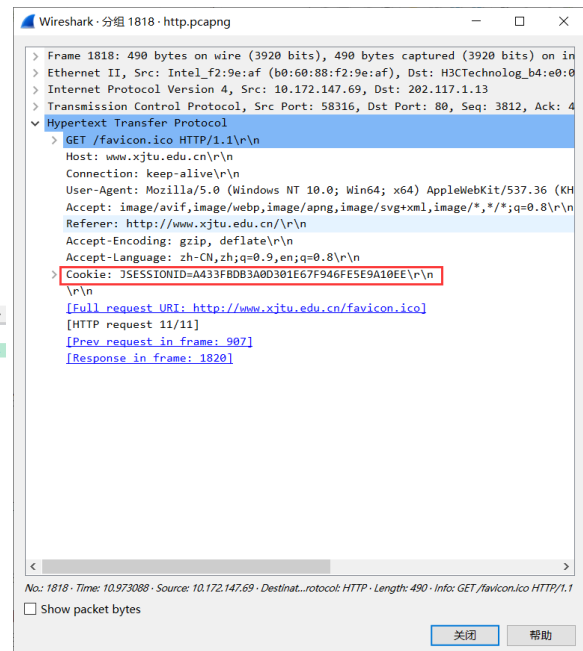
Set-Cookie

No.	Time	Source	Destination	Protocol	Length	Info
1705	9.859330	202.117.1.13	10.172.147.69	HTTP	927	HTTP/1.1 200 OK



Cookie

No.	Time	Source	Destination	Protocol	Length	Info
1818	10.973088	10.172.147.69	202.117.1.13	HTTP	490	GET /favicon.ico HTTP/1.1

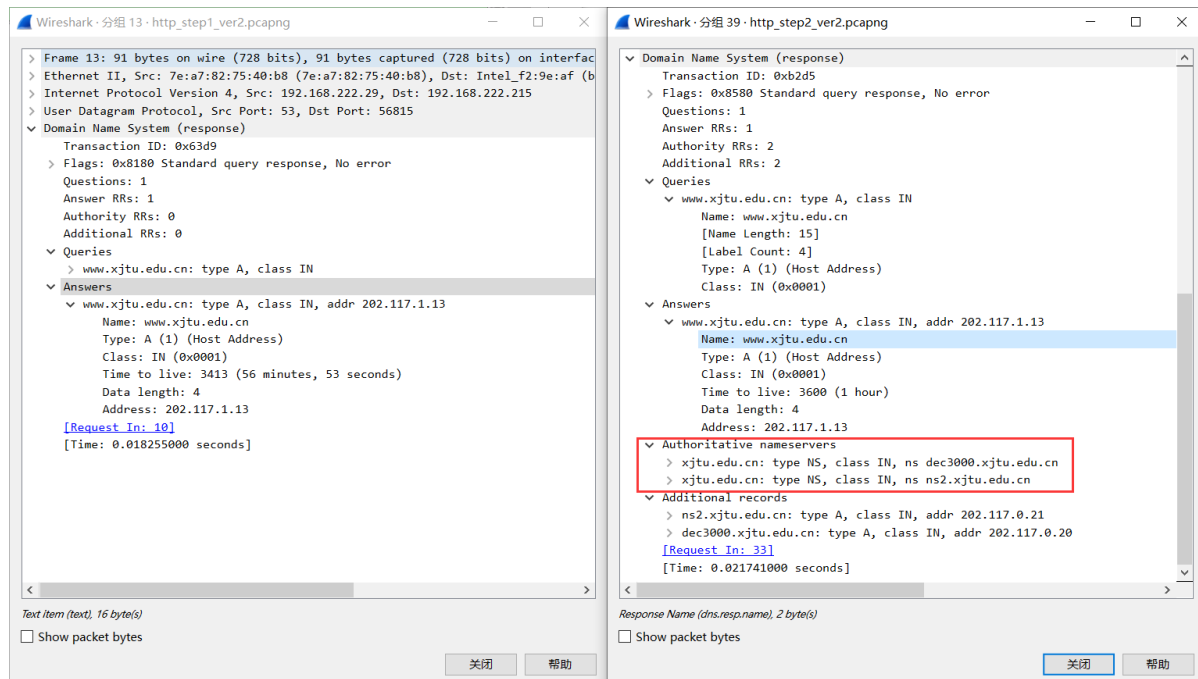


HTTPS的加密内容不好分析，可观察TLS加密传输的建立过程，传输端口等。

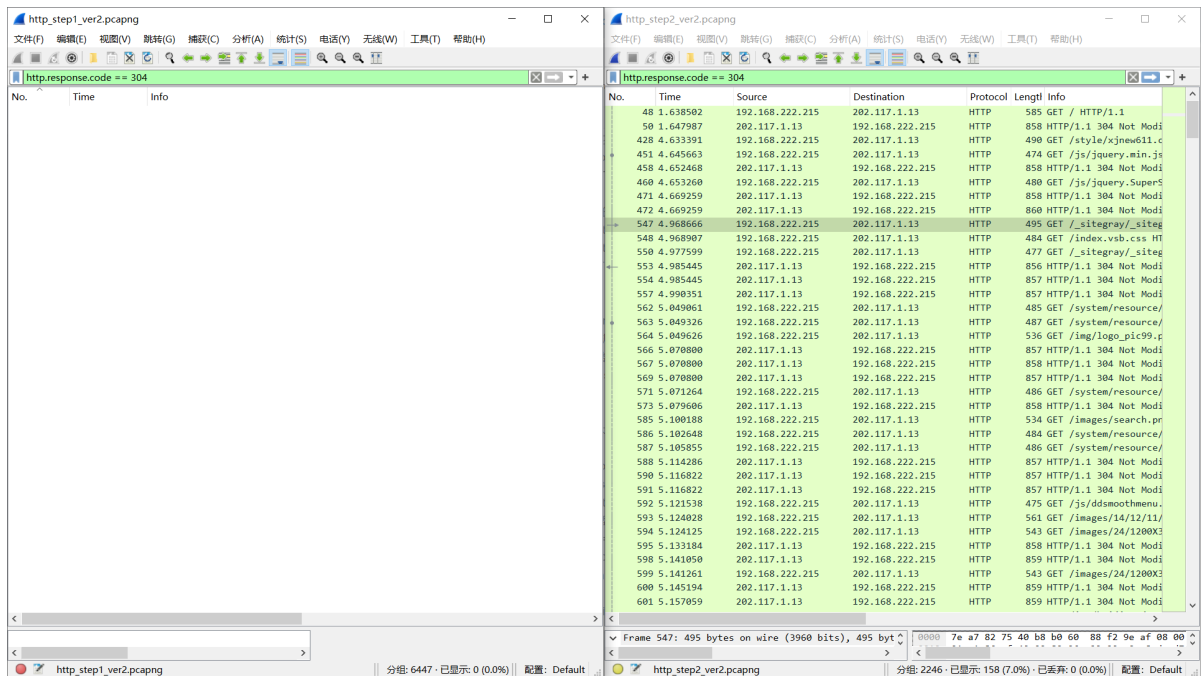
(二) 带缓存的ARP, DNS和HTTP协议分析

照着1.7.1中的步骤1-4再次执行一遍，但不执行步骤2。观察缓存的使用和带来的好处。

- 在DNS解析的过程中，缓存记录了相应权威域名服务器的地址，对相同域名进行解析时无需多次迭代查询，直接向权威域名服务器查询即可



- 通过http GET请求获取资源时，缓存避免了对已有对象的重复获取，节省了计算机资源



(三) 使用ncat工具访问HTTP服务

参考1.7.1中的步骤1-4和分析结果，在命令窗口执行ncat -C xxx.xxx.xxx.xxx 80，ncat连接上HTTP服务器后，根据协议输入合适的请求。其中xxx.xxx.xxx.xxx 为服务器地址。

答：

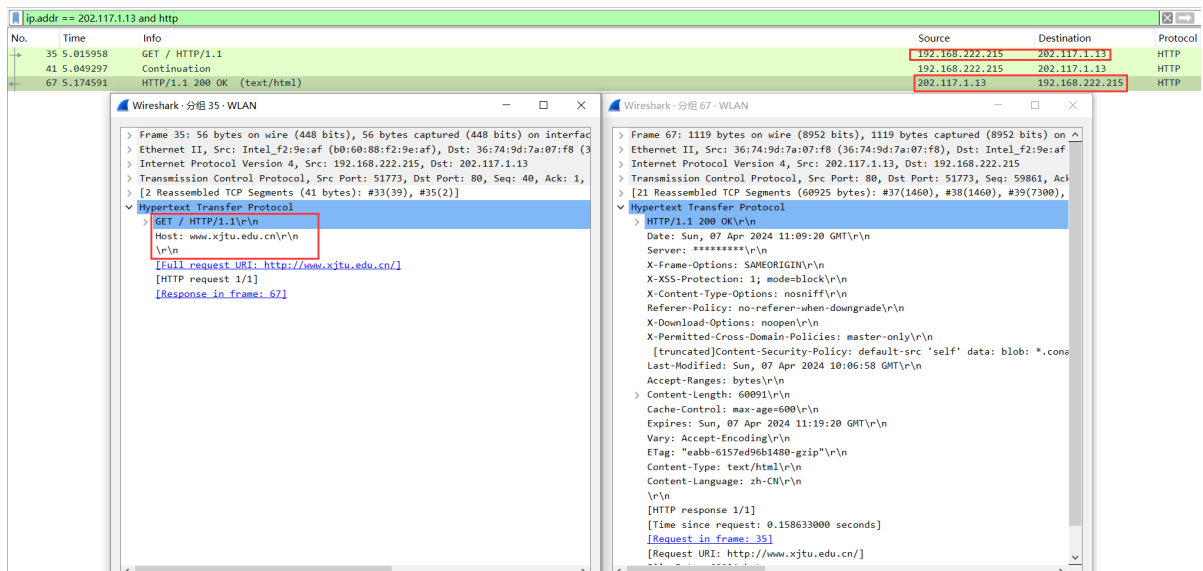
命令输入

```
PS G:\nmap> .\ncat.exe -C 202.117.1.13 80
GET / HTTP/1.1
Host: www.xjtu.edu.cn

HTTP/1.1 200 OK
Date: Sun, 07 Apr 2024 11:09:20 GMT
Server: *****
X-Frame-Options: SAMEORIGIN
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
Referer-Policy: no-referer-when-downgrade
X-Download-Options: noopen
X-Permitted-Cross-Domain-Policies: master-only
Content-Security-Policy: default-src 'self' data: blob: *.conac.cn *.xjtu.edu.cn *.gov.cn *.jiathis.com *.baidu.com *.bshare.cn *.eol.cn *.qq.com *.kaipuyun.cn *.bdimg.com *.wx.qq.com *.people.com.cn *.weibo.com *.m1905.cn 'unsafe-inline' 'unsafe-eval'; frame-ancestors 'self';
Last-Modified: Sun, 07 Apr 2024 10:06:58 GMT
Accept-Ranges: bytes
Content-Length: 60091
Cache-Control: max-age=600
Expires: Sun, 07 Apr 2024 11:19:20 GMT
Vary: Accept-Encoding
ETag: "eabb-6157ed96b1480-gzip"
Content-Type: text/html
Content-Language: zh-CN

<?DOCTYPE HTML>
<HTML><HEAD><TITLE>缓垮番浜ら€氣ゝ瀛?/TITLE>
```

抓包结果，可以看到发出了一个内容为 Get / HTTP/1.1\r\nHost: www.xjtu.edu.cn\r\n的GET请求包，相应的也收到了对应的响应包



2. FTP协议分析

(一) FTP协议的分析

步骤1：在远程的云服务器上开启ftp服务，并在云服务器控制台把21端口开放。在云服务器上运行报文截获工具（如Linux的tcpdump，Windows的Wireshark）截获FTP报文。

步骤2：在计算机终端上运行Wireshark截获报文，使用IE浏览器来访问该FTP服务器。比如在地址栏输入 <ftp://xxx.xxx.xxx.xxx/> 其中xxx.xxx.xxx.xxx 是该云服务器的IP地址。

步骤3：进入FTP服务器中的某个目录中，下载一个文件。结束后，停止报文截获。

分析该FTP的过程。注意对照服务器和客户端的一起分析，注意NAT的影响。观察是否使用了被动模式，如果是主动模式并且工具允许，使用被动模式再做一次下载，并分析。

如果FTP不能正常工作，请仔细抓包并分析原因。NAT穿越问题可能是原因之一。

(二) 使用ncat工具来访问FTP服务

步骤1：提前把用到的FTP命令准备好（写在notepad++中）。

步骤2：在云服务器上运行报文截获工具准备截获FTP报文。在命令窗口执行ncat -C xxx.xxx.xxx.xxx 21，ncat连接上FTP服务器后，根据协议输入合适的命令（**注意：控制连接和数据连接需要在两个命令窗口分别建立**）。其中xxx.xxx.xxx.xxx 为云服务器地址。

步骤3：解析完毕后停止报文截获。把过程整理记录到报告中。

在ncat模拟FTP客户端的过程中，请查看服务器的文件列表，并下载一个不大的文本文件。过程中，必要时用netstat命令观察双方的（新开）端口监听情况。

六、 互动讨论主题

1、HTTP协议的缓存，DNS的缓存；缓存对网络访问速度的影响。

http缓存机制：

ETag 字段，资源标识符，当资源内容不变时其 **ETag** 值也不会变，与 **If-None-Match** 字段配合使用

If-None-Match 字段，与响应头部中的 **ETag** 字段配合使用；当资源过期时，将该字段的值设置为 **ETag** 的值，服务器收到请求后将该值与服务器中所有资源的 **ETag** 值进行对比，如果资源没变化状态码返回304(Not Modified)，如果有变化状态码返回200、并返回变化后的资源

Last-Modified 字段，包含文件上一次修改时间，与 **If-Modified-Since** 配合使用

`If-Modified-Since` 字段，与响应头部中的 `Last-Modified` 字段配合使用；当资源过期时，将该字段的值设置为 `Last-Modified` 的值，服务器收到请求后对比资源的最近修改时间，如果最近修改时间大于 `Last-Modified` 的值则状态码返回200、并返回最新资源，否则状态码返回304

DNS缓存机制：

将DNS解析记录保存在本地客户机和服务器上，并设置TTL(生存时间 time to live)，在该记录过期之前，对相同域名的重复解析可以直接使用本地缓存内容

影响：

缓存机制的存在大大提高了网络访问速度

2、NAT对FTP传输的影响，比较HTTP与FTP的特点；

七、进阶自设计

1、用nmap的ncat来模拟https客户端，访问1-2个网站。

2、在云服务器上搭建Apache2（或其他WEB服务器），并测试修改HTML或图片文件，看客户端能否及时访问到更新的内容。注意抓包分析。