**实验四、应用层协议分析实验报告**

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

1. 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。

1. 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。

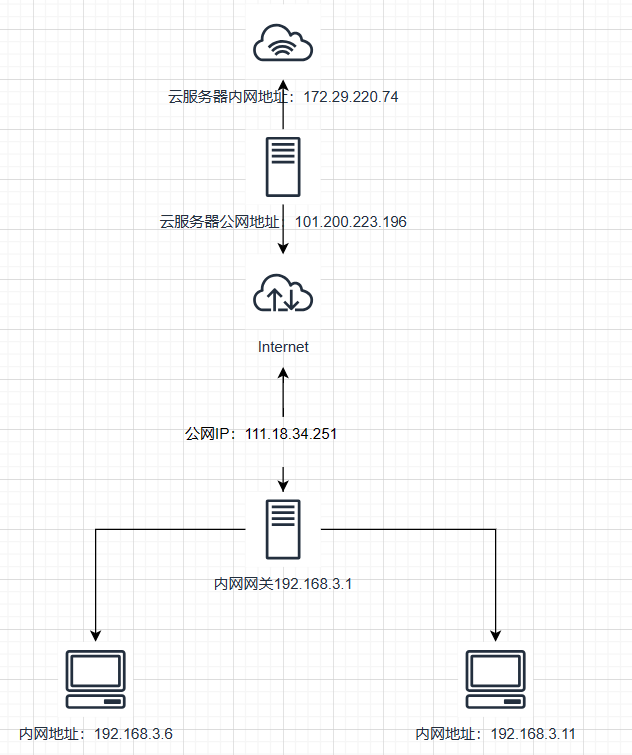
**注：HTTP和FTP两个协议二选一。**

1. 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。

1. 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。



1. 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

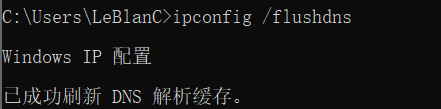
步骤1：在计算机终端上运行Wireshark截获所有的报文。

步骤2：清空ARP，DNS和HTTP浏览器的缓存：

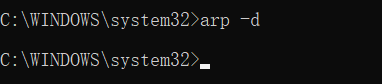
浏览器缓存的清除以Chrome浏览器为例，地址栏中输入chrome://settings/，找到高级选项中的“隐私设置和安全性”，清除浏览数据。

**此步直接使用了新下载的浏览器，跳过。**

执行“ipconfig /flushdns”清除本地DNS缓存。



执行“arp –d”命令清空arp缓存。



**执行成功。**

步骤3：在浏览器中访问3个网址，比如[www.xjtu.edu.cn](http://www.xjtu.edu.cn), [www.github.com](http://www.github.com), [www.unb.br](http://www.unb.br)；

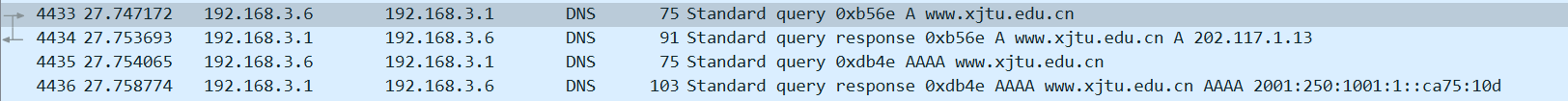


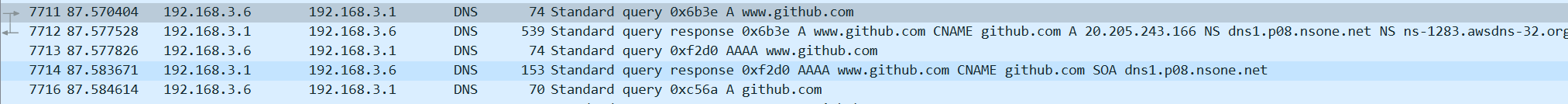
步骤4：执行完之后，Wireshark停止报文截获，分析截获的报文。

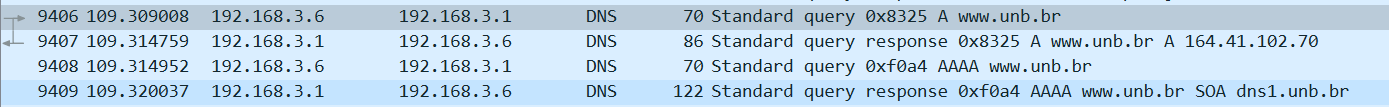
观察几个协议的配合使用，注意访问的延迟情况。特别分析HTTP的请求和应答。注意一个网址的访问中：

* 有几次DNS解析；

[www.xjtu.edu.cn](http://www.xjtu.edu.cn)



[www.github.com](http://www.github.com)

[www.unb.br](http://www.unb.br)

DNS解析的一共有四次。

一次主机向网关发起IPv4询问。

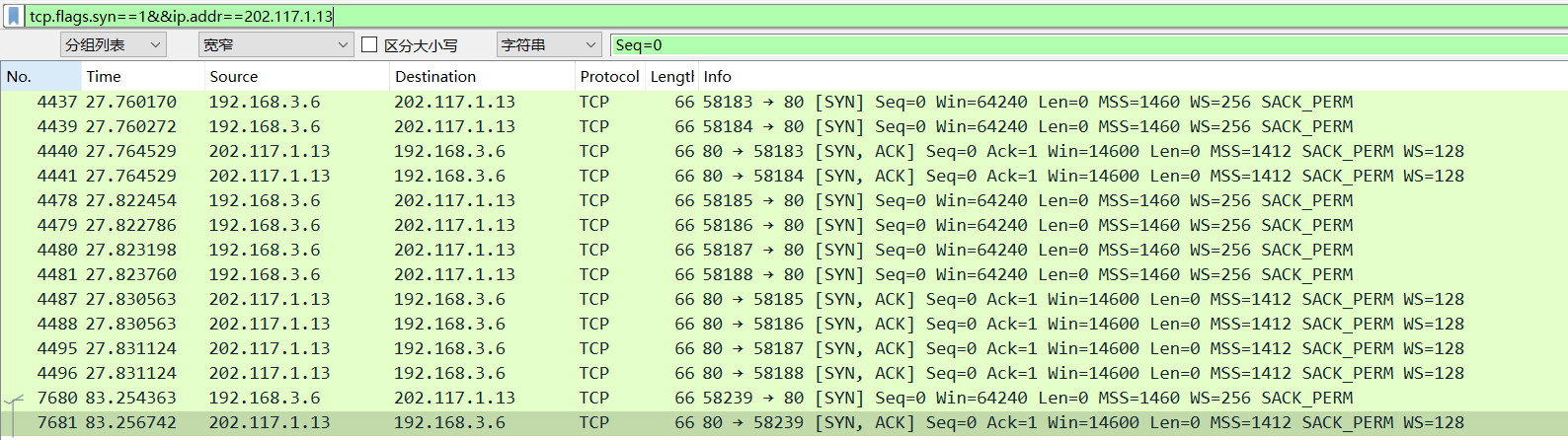
一次网关回复主机IPv4地址。

一次主机向网关发起IPv6询问。

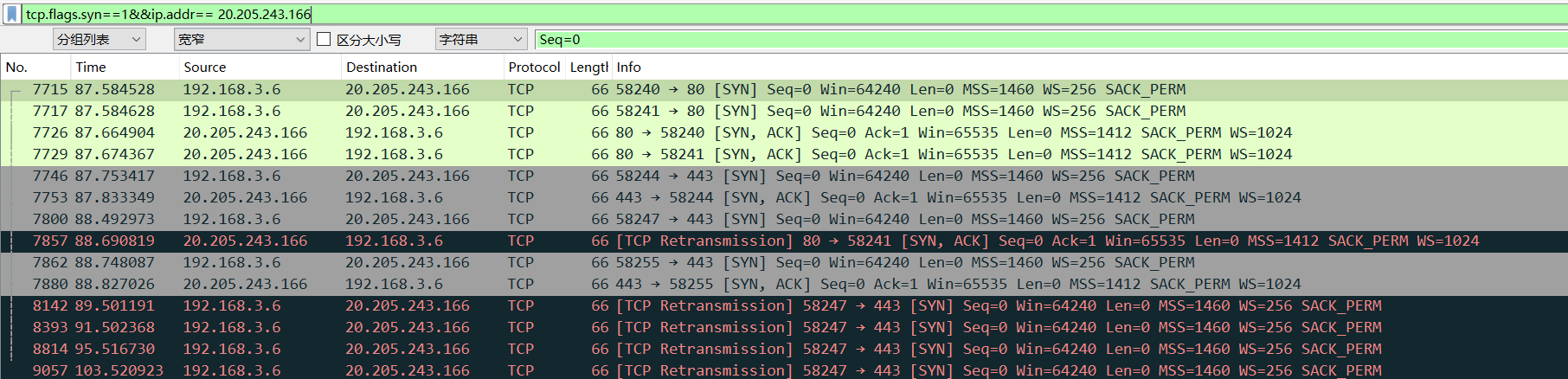
一次网关向主机回复IPv6地址。

之后会定期检查或刷新DNS缓存。这种刷新可以确保在DNS记录发生变化时（例如，网站更换了服务器或IP地址），本地缓存能够得到更新，从而保持与最新DNS记录的同步。

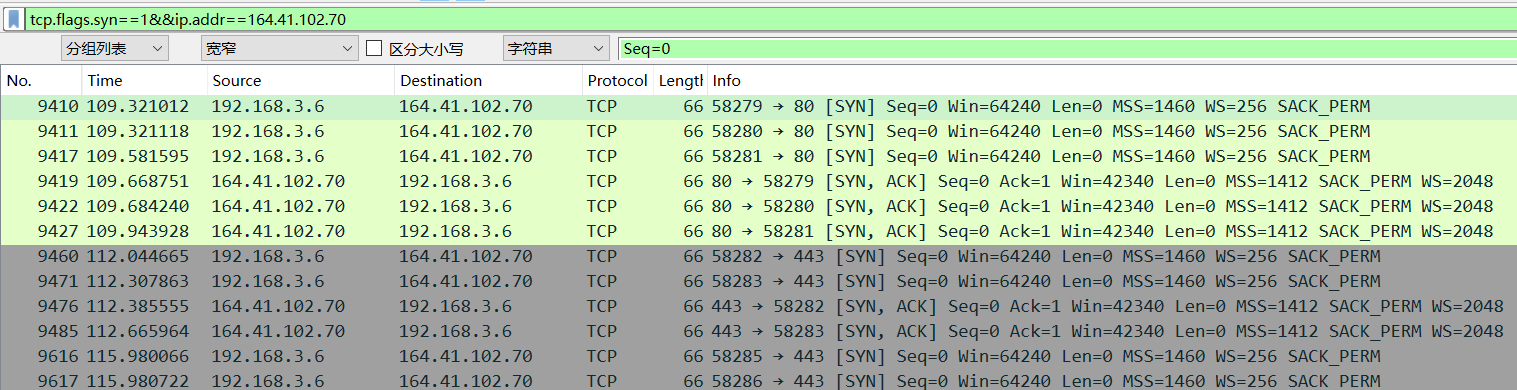
* 用了几个连接（建立连接时，本地的端口不同，HTTP服务常用端口是80，HTTPS服务常用端口是443）；

对于[www.xjtu.edu.cn(202.117.1.13)](http://www.xjtu.edu.cn(202.117.1.13))

有52183，52184，52185，52186，52187，52188，52239 7个端口进行了TCP连接。

对于[www.github.com(20.205.243.166)](http://www.github.com(20.205.243.166))

有58240，58241，58244，58247，58255 5个端口进行了TCP连接，但是58241和58237端口的连接发生了异常。

对于[www.unb.br(164.41.102.70)](http://www.unb.br(164.41.102.70))

端口号从58279——58311，在此不再赘述。

* 取了几个对象（GET的对象，如：HTML，CSS，JS，图片等）；

以[www.xjtu.edu.cn](http://www.xjtu.edu.cn)为例有HTML,CSS,JS,PNG/JPEG JEIF image,GIF89a











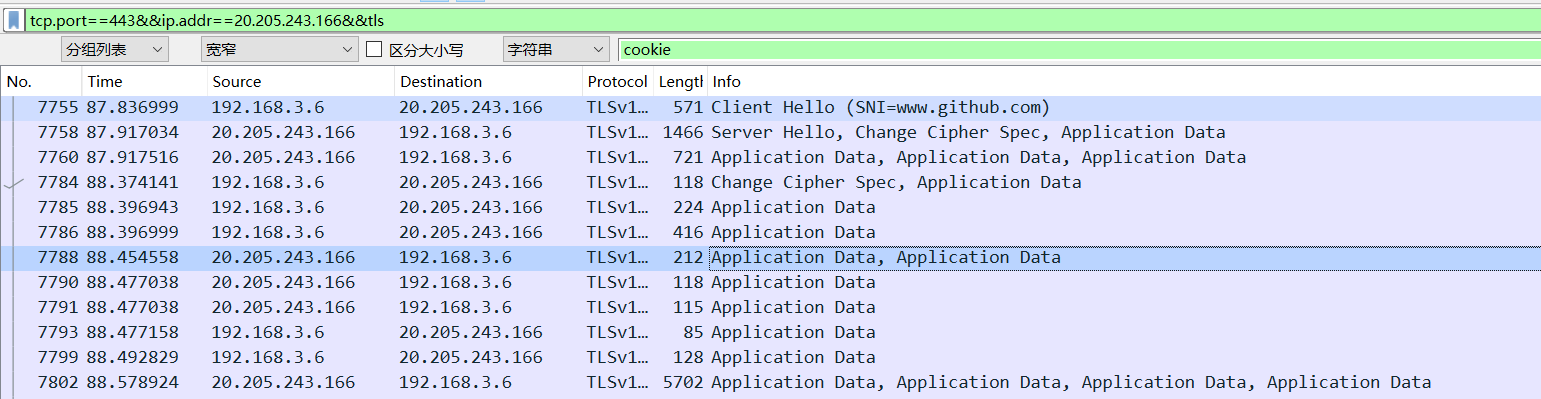


* 有没有Cookie及其工作过程（Set-Cookie、Cookie、304 Not Modified相关数据包）等；

没有捕捉到Cookie等包，推测是因为使用的是新下载的浏览器，不存在任何cookie。

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。

通过端口号tcp.port==443筛选HTTPS的TLS传输过程：



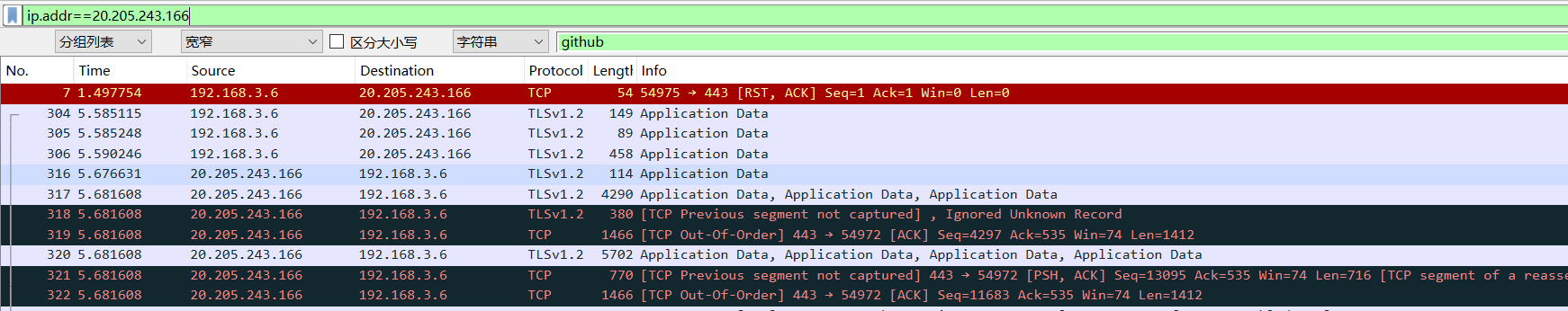
可以看到首先有客户端向放服务器发起Client Hello请求，然后服务回应Server Hello，并且捎带Application Data，开始通信

（二）带缓存的ARP，DNS和HTTP协议分析

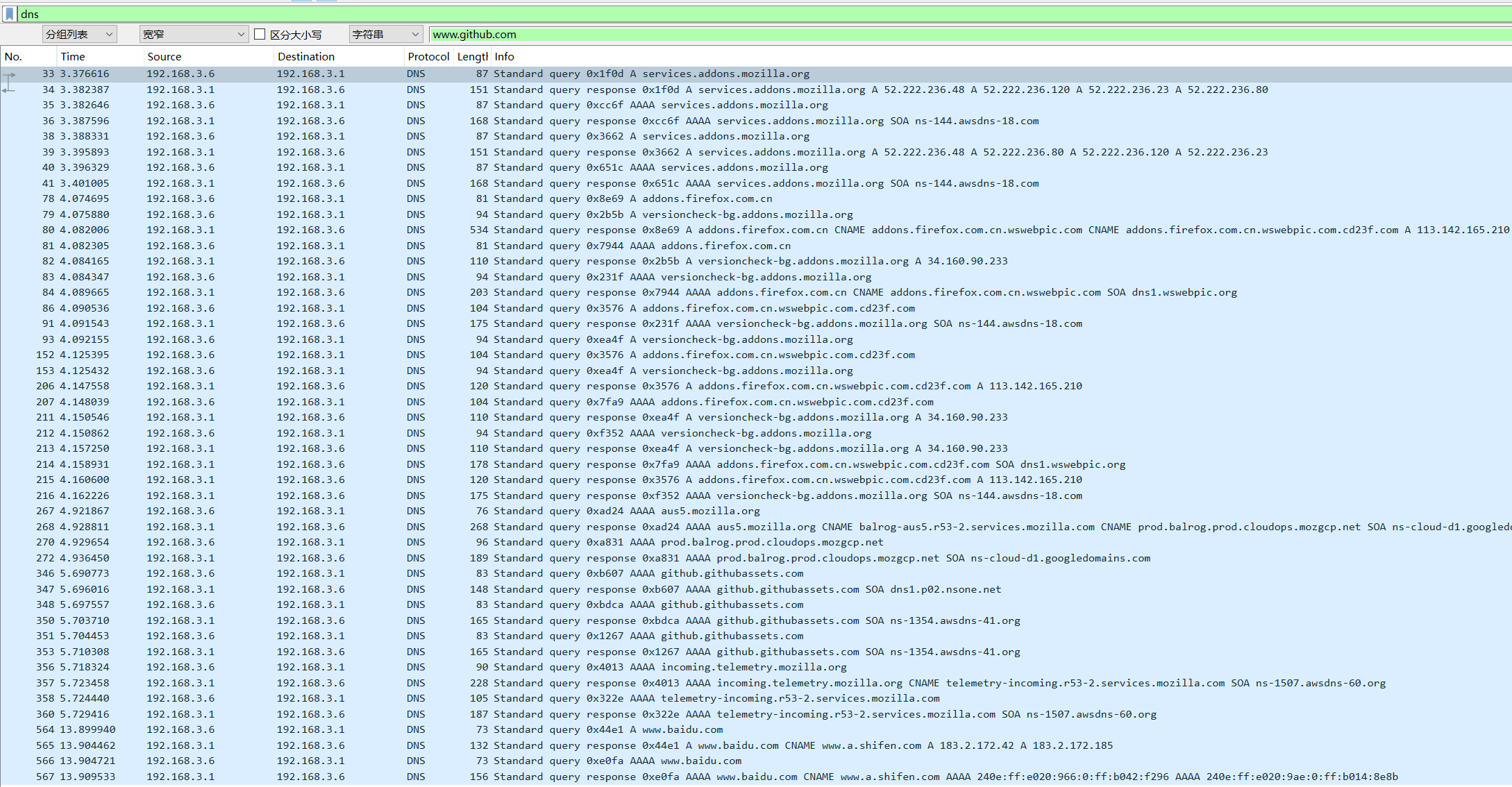
照着1.7.1中的步骤1-4再次执行一遍，但不执行步骤2。观察缓存的使用和带来的好处。

不执行步骤2，以[www.github.com](http://www.github.com)(20.205.243.166)为例：

以ip.addr==20.205.243.166为筛选条件，可以看到确实与[www.github.com](http://www.github.com)进行了通信。



使用dns筛选：



没有找到任何与[www.github.com有关的dns](http://www.github.com有关的dns)请求，说明直接使用了本地的dns缓存而没有进行dns域名解析，减少了工作量。

（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。

**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。

1. 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。

2、NAT对FTP传输的影响，比较HTTP与FTP的特点；

1. 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。