实验四、应用层协议分析实验报告

组号：	**3-3**				
姓名：	李云广	学号：	2193712575	班级：	计算机93
姓名：	李怀邦	学号：	2193712530	班级：	计算机93

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。

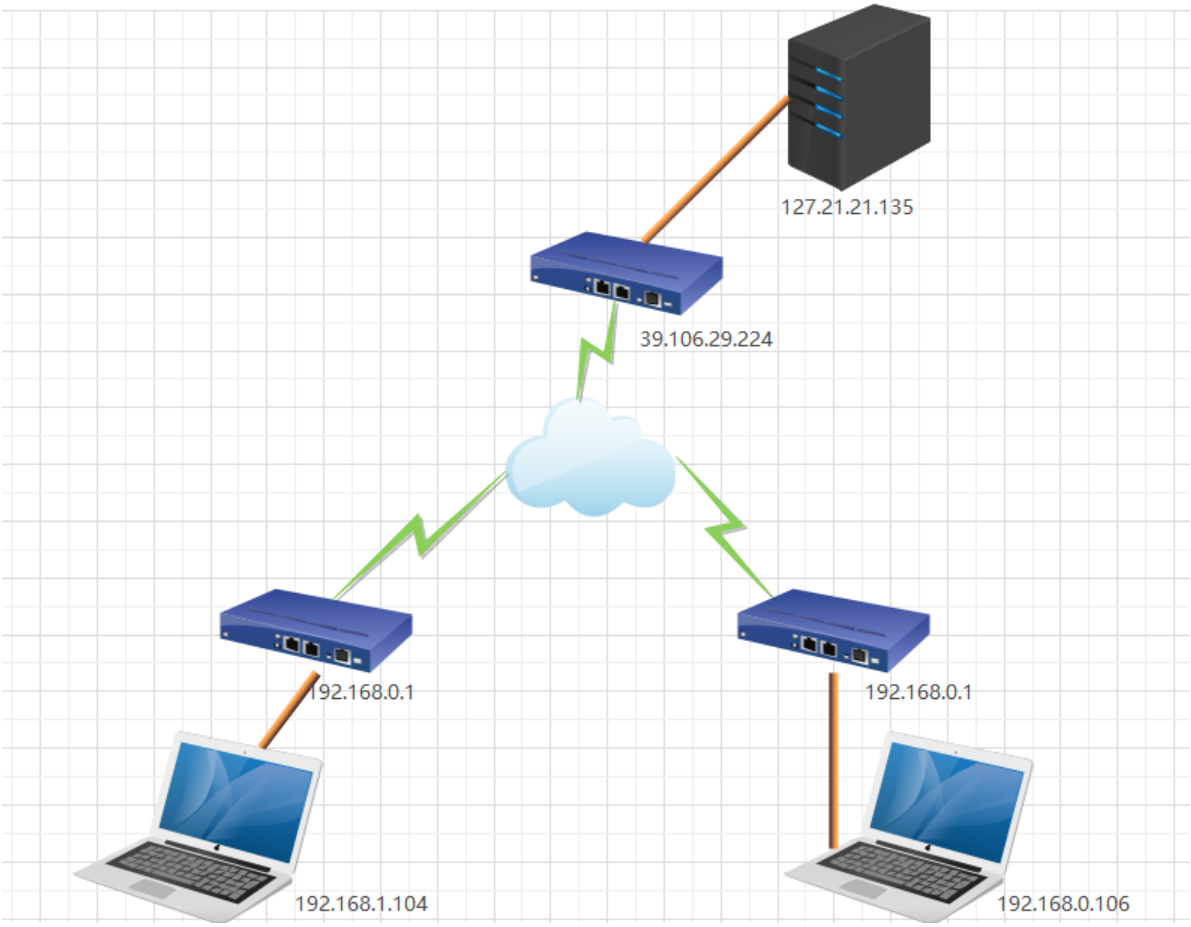
> 本实验选择的HTTP协议的分析

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。

步骤3：在浏览器中访问3个网址，比如www.xjtu.edu.cn, http://sz.xju.edu.cn/index.htm , http://www.sun.ac.za/english ；

步骤4：执行完之后，Wireshark停止报文截获，分析截获的报文。

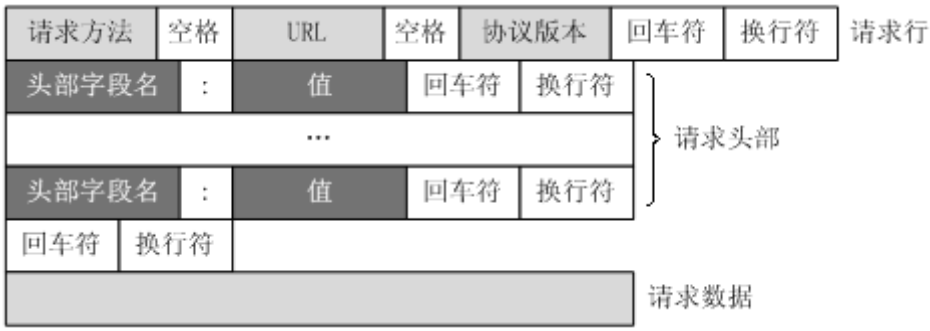
观察几个协议的配合使用，注意访问的延迟情况。特别分析HTTP的请求和应答。注意一个网址的访问中，用了几个连接，取了几个对象（HTML，CSS，JS，图片等），有几次DNS解析，有没有Cookie等。

1. 首先分析http的基本请求和应答报文，这里以xjtu官网为例：

```
> Frame 8449: 487 bytes on wire (3896 bits), 487 bytes captured (3896 bits) on interface \Device\NPF_{E87E10FB-CDA0-40E8-BEC6-BB261C1C9146}, id
> Ethernet II, Src: IntelCor_03:8c:98 (60:dd:8e:03:8c:98), Dst: Shenzhen_56:0e:d0 (c0:a5:dd:56:0e:d0)
> Internet Protocol Version 4, Src: 192.168.1.106, Dst: 159.138.3.217
> Transmission Control Protocol, Src Port: 51408, Dst Port: 80, Seq: 1, Ack: 1, Len: 433
v Hypertext Transfer Protocol
  v GET / HTTP/1.1\r\n
    v [Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]
        [GET / HTTP/1.1\r\n]
        [Severity level: Chat]
        [Group: Sequence]
      Request Method: GET
      Request URI: /
      Request Version: HTTP/1.1
    Host: www.xjtu.edn.cn\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/99.0.4844.74 Safari/537.36\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n
    Accept-Encoding: gzip, deflate\r\n
    Accept-Language: zh-CN,zh;q=0.9\r\n
    \r\n
    [Full request URI: http://www.xjtu.edn.cn/]
    [HTTP request 1/2]
    [Response in frame: 8561]
    [Next request in frame: 8702]
```

【HTTP请求报文分析1】

这里可以看到http请求报文的格式



报文结构大概是如上图所示的。

请求方法：GET

URL： 空

协议版本： HTTP/1.1 \r\n

Host: www.xjtu.edu.cn

Connection: Keep-alive

Accept: text/html 这里写明了要请求的数据

Accept-Encoding: gzip,dflate 这里写明了编码方式

Accept-Language: zh-CN,zh;q=0.9 写明了语言

注意到这里GET请求的URL内容为空，考虑到这是建立连接后第一次发送GET报文，所以说客户端可能并不知道要请求什么东西。

```
> Frame 13920: 960 bytes on wire (7680 bits), 960 bytes captured (7680 bits) on interface \Device\NPF_{E87E10FB-CDA0-40E8-BEC6-BB261C1C9146}, id 0
> Ethernet II, Src: Shenzhen_56:0e:d0 (c0:a5:dd:56:0e:d0), Dst: IntelCor_03:8c:98 (60:dd:8e:03:8c:98)
> Internet Protocol Version 4, Src: 202.117.1.13, Dst: 192.168.1.106
> Transmission Control Protocol, Src Port: 80, Dst Port: 51472, Seq: 12961, Ack: 474, Len: 906
> [10 Reassembled TCP Segments (13866 bytes): #13911(1440), #13912(1440), #13913(1440), #13914(1440), #13915(1440), #13916(1440), #13917(1440), #13918(1440), #13919(1440), #13920(906)]
v Hypertext Transfer Protocol
  > HTTP/1.1 200 OK\r\n
    Date: Sun, 20 Mar 2022 06:56:01 GMT\r\n
    Server: VWebServer\r\n
    X-Frame-Options: SAMEORIGIN\r\n
    Last-Modified: Sat, 19 Mar 2022 15:56:48 GMT\r\n
    ETag: "e0a3-5da944f89d000"\r\n
    Accept-Ranges: bytes\r\n
    Cache-Control: max-age=600\r\n
    Expires: Sun, 20 Mar 2022 07:06:01 GMT\r\n
    Vary: Accept-Encoding\r\n
    Content-Encoding: gzip\r\n
  > Content-Length: 13421\r\n
    Keep-Alive: timeout=5, max=200\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html\r\n
    Content-Language: zh-CN\r\n
    \r\n
    [HTTP response 1/19]
    [Time since request: 0.007584000 seconds]
    [Request in frame: 13906]
    [Next request in frame: 13935]
    [Next response in frame: 13958]
    [Request URI: http://www.xjtu.edu.cn/img/in_app_80.png]
    Content-encoded entity body (gzip): 13421 bytes -> 57507 bytes
    File Data: 57507 bytes
> Line-based text data: text/html (996 lines)
```

这里可以看到第一次服务器给客户端返回的应答报文

【HTTP应答报文分析】

版本：HTTP/1.1

状态码：200 OK

Connection:keep-alive

Server:vwebserver

Last-Modified: Sat,19 MAR 2022 15:56:48 GMT

Content-length:13421

Content-Type:text/html

一、www.xjtu.edu.cn（无缓存）：

【连接分析】（6个TCP连接）：

3901	66.800425	192.168.1.106	202.117.1.13	TCP	66 51472 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK
3902	66.801868	192.168.1.106	202.117.1.13	TCP	66 51473 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK
3904	66.802332	192.168.1.106	202.117.1.13	TCP	54 51472 → 80 [ACK] Seq=1 Ack=1 Win=132352 Len=0
3906	66.802653	192.168.1.106	202.117.1.13	HTTP	527 GET / HTTP/1.1
3909	66.803871	192.168.1.106	202.117.1.13	TCP	54 51473 → 80 [ACK] Seq=1 Ack=1 Win=132352 Len=0
3921	66.810475	192.168.1.106	202.117.1.13	TCP	54 51472 → 80 [ACK] Seq=474 Ack=12961 Win=132352 Len=0
3922	66.810548	192.168.1.106	202.117.1.13	TCP	54 51472 → 80 [ACK] Seq=474 Ack=13867 Win=131328 Len=0
3935	66.853998	192.168.1.106	202.117.1.13	HTTP	432 GET /style/xjnew611.css HTTP/1.1
3936	66.854463	192.168.1.106	202.117.1.13	HTTP	415 GET /js/jquery.min.js HTTP/1.1
3937	66.854760	192.168.1.106	202.117.1.13	TCP	66 51475 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK
3938	66.855184	192.168.1.106	202.117.1.13	TCP	66 51476 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK
3939	66.855381	192.168.1.106	202.117.1.13	TCP	66 51477 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK
3940	66.855524	192.168.1.106	202.117.1.13	TCP	66 51478 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK
3948	66.857569	192.168.1.106	202.117.1.13	TCP	54 51475 → 80 [ACK] Seq=1 Ack=1 Win=132352 Len=0
3949	66.857622	192.168.1.106	202.117.1.13	TCP	54 51476 → 80 [ACK] Seq=1 Ack=1 Win=132352 Len=0
3950	66.857637	192.168.1.106	202.117.1.13	TCP	54 51477 → 80 [ACK] Seq=1 Ack=1 Win=132352 Len=0
3951	66.857703	192.168.1.106	202.117.1.13	TCP	54 51478 → 80 [ACK] Seq=1 Ack=1 Win=132352 Len=0
3952	66.857725	192.168.1.106	202.117.1.13	HTTP	439 GET /_sitegray/_sitegray_d.css HTTP/1.1

这里可以看到一共有6个端口（51472、51473、51475、51476、51477、51478）与xjtu主机进行了TCP连接，这六个端口均发送[SYN]报文请求连接，并且服务器也均返回了应答可以连接。

【对象分析】

192.168.1.106	202.117.1.13	HTTP	381	GET /_sitegray/_sitegray.js HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	399	GET /_sitegray/_sitegray_d.css HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	484	GET /favicon.ico HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	438	GET /images/1234444.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	462	GET /images/14/12/11/1tf5znre9c/20220305012.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	462	GET /images/14/12/11/1tf5znre9c/20220314011.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	462	GET /images/14/12/11/1tf5znre9c/20220315011.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	462	GET /images/14/12/11/1tf5znre9c/20220315012.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	462	GET /images/14/12/11/1tf5znre9c/20220319011.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	455	GET /images/17/09/04/1mi2amdixo/wmjd.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	436	GET /images/170-1.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	436	GET /images/170-2.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	459	GET /images/18/01/23/1ay37aq43t/icon_r_1.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	459	GET /images/18/01/23/1ay37aq43t/icon_r_2.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	459	GET /images/18/01/23/1ay37aq43t/icon_r_3.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	459	GET /images/18/01/23/1ay37aq43t/icon_r_4.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	445	GET /images/20190925chuxin.jpg HTTP/1.1

此次**xjtu**网站的**http**请求中共有：**html**文件：**1**个 **css**文件： **12**个 **javascript**文件：**33**个 **jpg**文件：**61**个

【DNS解析】(54次）

dns分析由访问的本网站（www.xjtu.edu.cn)的解析202.117.1.13和附加（连接）网站构成，其中连接的网站及其ip地址为

ef.sjtu.edu.cn 122.228.238.15

chuxin.xjtu.edu.cn 122.228.238.15

dangshi.xjtu.edu.cn 117.23.61.159

dwzzb.xjtu.edu.cn 116.55.250.151

192.168.1.106	61.134.1.4	DNS	78	Standard query 0x943a A www.ef.xjtu.edu.cn
61.134.1.4	192.168.1.106	DNS	90	Standard query response 0xb0a9 A en.xjtu.edu.cn A 202.117.1.13
61.134.1.4	192.168.1.106	DNS	154	Standard query response 0x943a A www.ef.xjtu.edu.cn CNAME 81986794494f8bc
192.168.1.106	61.134.1.4	DNS	78	Standard query 0x438a A alumni.xjtu.edu.cn
192.168.1.106	61.134.1.4	DNS	75	Standard query 0x1401 A baike.baidu.com
192.168.1.106	61.134.1.4	DNS	76	Standard query 0xed0b A cfsp.xjtu.edu.cn
61.134.1.4	192.168.1.106	DNS	150	Standard query response 0x1401 A baike.baidu.com CNAME bk.baidu.com CNAME
61.134.1.4	192.168.1.106	DNS	154	Standard query response 0x438a A alumni.xjtu.edu.cn CNAME c2a445a772ec4af
61.134.1.4	192.168.1.106	DNS	152	Standard query response 0xed0b A cfsp.xjtu.edu.cn CNAME 84b0cf2a8b2c0aac.
192.168.1.106	61.134.1.4	DNS	78	Standard query 0x12ab A chuxin.xjtu.edu.cn
192.168.1.106	61.134.1.4	DNS	79	Standard query 0x9301 A dangshi.xjtu.edu.cn
192.168.1.106	61.134.1.4	DNS	77	Standard query 0xd9f0 A dwzzb.xjtu.edu.cn
192.168.1.106	61.134.1.4	DNS	91	Standard query 0xe83b A content-autofill.googleapis.com
61.134.1.4	192.168.1.106	DNS	154	Standard query response 0x12ab A chuxin.xjtu.edu.cn CNAME 59d2665d55c4263
61.134.1.4	192.168.1.106	DNS	155	Standard query response 0x9301 A dangshi.xjtu.edu.cn CNAME 51efdd9ecf1d59
61.134.1.4	192.168.1.106	DNS	153	Standard query response 0xd9f0 A dwzzb.xjtu.edu.cn CNAME dbcd67a5140926a9
192.168.1.106	61.134.1.4	DNS	75	Standard query 0xc51d A ejp.xjtu.edu.cn

【cookie分析】

192.168.1.106	202.117.1.13	HTTP	506	GET /img/tabar-1.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	504	GET /img/tab-2.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	504	GET /img/tab-3.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	509	GET /img/tabar-1-on.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	509	GET /img/tabar-2-on.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	509	GET /img/tabar-3-on.jpg HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	507	GET /img/in_xn_tb.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	507	GET /img/in_xn_cw.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	507	GET /img/in_xn_ky.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	509	GET /img/yidongxjtu.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	507	GET /img/icon_a10.gif HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	507	GET /img/icon_a11.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	508	GET /img/in_app_80.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	514	GET /images/navigationbg.png HTTP/1.1
192.168.1.106	202.117.1.13	HTTP	484	GET /favicon.ico HTTP/1.1

使用`http.cookie`过滤命令可以过滤出存在cookie的报文，发现仅有部分图片的请求存在cookie但其他http请求报文均没有cookie。

【延迟分析】

在访问网站的时候就明显感觉到南非的网站访问的非常缓慢，考虑到这应该就是地域所造成的影响。下面可以使用wireshark具体来分析一下：

	Time	Source	Destination	Protocol	Length	The RTT to ACK the :	Info
302	25.585267	192.168.1.106	202.117.1.13	TCP	54	0.000598000	54795 → 80
382	25.645008	192.168.1.106	202.117.1.13	TCP	54	0.000605000	54796 → 80
515	25.669361	192.168.1.106	202.117.1.13	HTTP	454	0.000774000	GET /img/ir
384	25.906051	192.168.1.106	202.117.1.13	TCP	54	0.000661000	54798 → 80
788	26.017439	192.168.1.106	202.117.1.13	TCP	54	0.000730000	54798 → 80
502	25.772401	192.168.1.106	202.117.1.13	HTTP	454	0.000978000	GET /img/ir
553	25.474546	192.168.1.106	202.117.1.13	TCP	54	0.000786000	54795 → 80
385	25.906178	192.168.1.106	202.117.1.13	TCP	54	0.000788000	54800 → 80
959	25.716109	192.168.1.106	202.117.1.13	HTTP	454	0.001021000	GET /img/ir
494	25.761516	192.168.1.106	202.117.1.13	HTTP	454	0.001855000	GET /img/ir
355	26.114302	202.117.1.13	192.168.1.106	TCP	1494	0.001559000	80 → 54796
541	25.566507	192.168.1.106	202.117.1.13	HTTP	438	0.001733000	GET /img/lo
529	31.348085	202.117.1.13	192.168.1.106	TCP	54	0.001812000	80 → 54796
528	31.348085	202.117.1.13	192.168.1.106	TCP	54	0.001857000	80 → 54798
584	25.572548	192.168.1.106	202.117.1.13	HTTP	462	0.002004000	GET /images
527	31.348085	202.117.1.13	192.168.1.106	TCP	54	0.001902000	80 → 54800
579	25.770395	192.168.1.106	202.117.1.13	HTTP	454	0.002028000	GET /img/ir
526	31.348085	202.117.1.13	192.168.1.106	TCP	54	0.001950000	80 → 54801
525	31.348085	202.117.1.13	192.168.1.106	TCP	54	0.002000000	80 → 54799
209	25.871953	192.168.1.106	202.117.1.13	HTTP	507	0.002208000	GET /img/ir
574	25.530764	202.117.1.13	192.168.1.106	TCP	66	0.002053000	80 → 54801
524	31.348085	202.117.1.13	192.168.1.106	TCP	54	0.002133000	80 → 54795
536	25.564774	202.117.1.13	192.168.1.106	HTTP	794	0.002156000	HTTP/1.1 20
537	25.461312	202.117.1.13	192.168.1.106	TCP	66	0.002170000	80 → 54796
969	25.827794	202.117.1.13	192.168.1.106	TCP	1494	0.002212000	80 → 54801
539	25.565861	202.117.1.13	192.168.1.106	HTTP	897	0.002225000	HTTP/1.1 20

1. 可以看到xjtu官网请求和应答的RTT大约为0.0007s左右

	Time	Source	Destination	Protocol	Length	The RTT to ACK the ...	Info
16905	57.734652	111.115.76.75	192.168.1.106	TCP	1434	0.053560000	80 → 54840 [ACK] S
16683	57.558598	111.115.76.75	192.168.1.106	TCP	1434	0.053593000	80 → 54843 [ACK] S
16872	57.649115	111.115.76.75	192.168.1.106	TCP	1434	0.053684000	80 → 54842 [ACK] S
12821	56.666261	111.115.76.75	192.168.1.106	HTTP	1134	0.053778000	HTTP/1.1 200 OK (
13376	56.880731	111.115.76.75	192.168.1.106	TCP	1434	0.053782000	80 → 54840 [ACK] S
16852	57.620842	111.115.76.75	192.168.1.106	TCP	1434	0.053950000	80 → 54840 [ACK] S
16387	57.503548	111.115.76.75	192.168.1.106	TCP	1434	0.053959000	80 → 54839 [ACK] S
12131	56.121926	111.115.76.75	192.168.1.106	HTTP	1256	0.053976000	HTTP/1.1 200 OK (
12272	56.372899	111.115.76.75	192.168.1.106	TCP	1434	0.054005000	80 → 54840 [ACK] S
12097	56.063912	111.115.76.75	192.168.1.106	TCP	66	0.054013000	80 → 54844 [SYN, A
12477	56.508168	111.115.76.75	192.168.1.106	TCP	1434	0.054104000	80 → 54844 [ACK] S
12273	56.374204	111.115.76.75	192.168.1.106	TCP	1434	0.054173000	80 → 54842 [ACK] S
16851	57.620842	111.115.76.75	192.168.1.106	TCP	1434	0.054235000	80 → 54839 [ACK] S
14792	57.272538	111.115.76.75	192.168.1.106	TCP	1434	0.054255000	80 → 54839 [ACK] S
12387	56.449487	111.115.76.75	192.168.1.106	TCP	1434	0.054293000	80 → 54839 [ACK] S
12586	56.566738	111.115.76.75	192.168.1.106	TCP	1434	0.054605000	80 → 54844 [ACK] S
16845	57.619056	111.115.76.75	192.168.1.106	TCP	1434	0.054994000	80 → 54841 [ACK] S
12388	56.450579	111.115.76.75	192.168.1.106	TCP	1434	0.055037000	80 → 54844 [ACK] S
12957	56.723013	111.115.76.75	192.168.1.106	TCP	1434	0.055070000	80 → 54843 [ACK] S
13497	56.937018	111.115.76.75	192.168.1.106	TCP	1434	0.055293000	80 → 54842 [ACK] S
13470	56.902512	111.115.76.75	192.168.1.106	TCP	1434	0.055300000	80 → 54844 [ACK] S
14472	57.214551	111.115.76.75	192.168.1.106	TCP	1434	0.055547000	80 → 54839 [ACK] S
15954	57.445241	111.115.76.75	192.168.1.106	TCP	1434	0.055626000	80 → 54839 [ACK] S
12331	56.434086	111.115.76.75	192.168.1.106	TCP	1434	0.055840000	80 → 54842 [ACK] S
16701	57.561839	111.115.76.75	192.168.1.106	HTTP	461	0.056017000	HTTP/1.1 200 OK

2. 可以看到新疆大学的官网RTT大约为0.054s左右

	Time	Source	Destination	Protocol	Length	The RTT to ACK the ...	Info
21514	76.625441	146.232.21.213	192.168.1.106	TCP	54	0.274695000	80 →
21544	76.638083	192.168.1.106	146.232.21.213	TCP	54	0.274984000	5487
19454	72.947652	146.232.21.213	192.168.1.106	TCP	54	0.275349000	80 →
19360	72.677913	146.232.21.213	192.168.1.106	TCP	66	0.276043000	80 →
21720	77.157927	146.232.21.213	192.168.1.106	TCP	54	0.276323000	80 →
24504	82.013791	146.232.21.213	192.168.1.106	TCP	1494	0.277048000	80 →
26384	86.335606	192.168.1.106	146.232.21.213	TCP	54	0.277253000	5487
21405	76.341499	146.232.21.213	192.168.1.106	TCP	54	0.277256000	80 →
21996	77.753316	192.168.1.106	146.232.21.213	TCP	74	0.277415000	5487
25240	83.153296	146.232.21.213	192.168.1.106	TCP	54	0.277621000	80 →
21999	77.754280	192.168.1.106	146.232.21.213	TCP	66	0.278379000	5487
20504	75.209606	146.232.21.213	192.168.1.106	TCP	1494	0.278405000	80 →
21883	77.472368	192.168.1.106	146.232.21.213	TCP	54	0.278599000	5487
20115	74.654707	146.232.21.213	192.168.1.106	TCP	54	0.278668000	80 →
21515	76.632078	146.232.21.213	192.168.1.106	TCP	54	0.278829000	80 →
20131	74.658573	192.168.1.106	146.232.21.213	TCP	82	0.278862000	5487
18973	70.401084	146.232.21.213	192.168.1.106	TCP	66	0.279090000	80 →
22955	79.153378	192.168.1.106	146.232.21.213	TCP	66	0.279259000	5487
19727	73.531731	146.232.21.213	192.168.1.106	TCP	54	0.279434000	80 →
20141	74.659534	192.168.1.106	146.232.21.213	TCP	74	0.279510000	5487
20326	74.941714	146.232.21.213	192.168.1.106	TCP	54	0.279850000	80 →
23400	79.707572	146.232.21.213	192.168.1.106	TCP	54	0.279908000	80 →
22738	78.955575	146.232.21.213	192.168.1.106	TCP	1494	0.280602000	80 →
21222	76.063237	192.168.1.106	146.232.21.213	TCP	54	0.280906000	5487
25221	83.117074	192.168.1.106	146.232.21.213	TCP	54	0.281994000	5487

3. 可以看南非的这个网站RTT足足有0.28s左右

【协议之间关系分析】

首先，客户机通过arp广播寻求网关地址，网关告诉主机网关地址之后，才能在以太网头部写上网关的MAC地址，从而可以实现局域网内报文传递到网关。知道网关的地址之后，将网关的MAC地址存进arp缓存表中，然后就可以进行下面的操作。

然后利用dns协议解析域名，这里是询问DNS服务器（61.134.1.4）知道IP地址之后，才能在IP包的头部写上目的的IP地址，从而经过路由转发可以到达目的节点。

IntelCor_03:8c:98	Shenzhen_56:0e:d0	ARP	42	Who has 192.168.1.1? Tell 192.168.1.106
Shenzhen_56:0e:d0	IntelCor_03:8c:98	ARP	42	192.168.1.1 is at c0:a5:dd:56:0e:d0
4a:e7:8d:0a:15:73	Broadcast	ARP	42	Who has 192.168.1.18? Tell 192.168.1.100
Shenzhen_56:0e:d0	Broadcast	ARP	42	Who has 192.168.1.102? Tell 192.168.1.1
Shenzhen_56:0e:d0	Broadcast	ARP	42	Who has 192.168.1.105? Tell 192.168.1.1
192.168.1.106	61.134.1.4	DNS	75	Standard query 0x513a A www.xjtu.edu.cn
61.134.1.4	192.168.1.106	DNS	91	Standard query response 0x513a A www.xjtu.edu.cn A 202.117.1.13

在获得目的服务器ip地址后，发送http请求报文，http协议是封装在最里层的，通过http协议的规定，可以请求资源，响应请求等，从而使得客户端获得服务器的资源。

8.1.106	61.134.1.4	DNS	75	Standard query 0x513a A www.xjtu.edu.cn
.1.4	192.168.1.106	DNS	91	Standard query response 0x513a A www.xjtu.edu.cn A 202.117.1.13
en_56:0e:d0	Broadcast	ARP	42	Who has 192.168.1.102? Tell 192.168.1.1
8d:0a:15:73	Broadcast	ARP	42	Who has 192.168.1.103? Tell 192.168.1.100
8d:0a:15:73	Broadcast	ARP	42	Who has 169.254.239.204? Tell 192.168.1.100
en_56:0e:d0	Broadcast	ARP	42	Who has 192.168.1.102? Tell 192.168.1.1
8d:0a:15:73	Broadcast	ARP	42	Who has 192.168.1.7? Tell 192.168.1.100
8.1.106	202.117.1.13	HTTP	487	GET / HTTP/1.1
7.1.13	192.168.1.106	HTTP	960	HTTP/1.1 200 OK (text/html)

（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。

【DNS协议分析】

192.168.1.106	61.134.1.4	DNS	74	Standard query 0x08cf A www.google.com
61.134.1.4	192.168.1.106	DNS	90	Standard query response 0x08cf A www.google.com A 103.252.11
192.168.1.106	61.134.1.4	DNS	77	Standard query 0xcc74 A shuc-js.ksord.com
61.134.1.4	192.168.1.106	DNS	93	Standard query response 0xcc74 A shuc-js.ksord.com A 110.43.
192.168.1.106	61.134.1.4	DNS	70	Standard query 0x0483 A google.com
61.134.1.4	192.168.1.106	DNS	86	Standard query response 0x0483 A google.com A 142.251.43.14
192.168.1.106	61.134.1.4	DNS	77	Standard query 0x6ab2 A xjsz.xjedu.gov.cn
192.168.1.106	61.134.1.4	DNS	74	Standard query 0xa4e6 A www.iyaxin.com
61.134.1.4	192.168.1.106	DNS	146	Standard query response 0x6ab2 No such name A xjsz.xjedu.gov
61.134.1.4	192.168.1.106	DNS	135	Standard query response 0xa4e6 No such name A www.iyaxin.com
192.168.1.106	61.134.1.4	DNS	91	Standard query 0x0c53 A content-autofill.googleapis.com
61.134.1.4	192.168.1.106	DNS	107	Standard query response 0x0c53 A content-autofill.googleapis
192.168.1.106	61.134.1.4	DNS	80	Standard query 0x645f A beacons.gcp.gvt2.com
61.134.1.4	192.168.1.106	DNS	126	Standard query response 0x645f A beacons.gcp.gvt2.com CNAME

由于有本地DNS缓存，所以浏览器直接通过查找本地的DNS缓存就可以获得需要的IP地址，因此基本没有抓到有效的DNS请求和应答包。从这里可以看出DNS的缓存可以提高访问速度，因为不需要向DNS服务器询问IP地址了。

【ARP协议分析】

以访问xjtu官网为例，此时因为有缓存，所以并不需要arp协议进行网关的地址解析（图中的ARP协议是别的主机发送的），也不需要dns协议进行地址解析（图片中非相关网址dns解析），直接发送了http报文。

1286 6.451767	4a:e7:8d:0a:15:73	Broadcast	ARP	42	Who has 192.168.1.8? Tell 192.168.1.100
1291 6.553888	4a:e7:8d:0a:15:73	Broadcast	ARP	42	Who has 192.168.1.4? Tell 192.168.1.100
1292 6.558422	192.168.1.106	61.134.1.4	DNS	74	Standard query 0x08cf A www.google.com
1293 6.583851	61.134.1.4	192.168.1.106	DNS	90	Standard query response 0x08cf A www.google.com A 103.252.115.169
1422 7.170000	4a:e7:8d:0a:15:73	Broadcast	ARP	42	Who has 192.168.1.6? Tell 192.168.1.100
1584 7.782845	4a:e7:8d:0a:15:73	Broadcast	ARP	42	Who has 192.168.1.4? Tell 192.168.1.100
1793 8.499578	4a:e7:8d:0a:15:73	Broadcast	ARP	42	Who has 192.168.1.7? Tell 192.168.1.100
1947 8.818708	192.168.1.106	202.117.1.13	HTTP	601	GET /system/resource/code/datainput.jsp?owner=1151962237&e=1&w=1920
1949 8.822501	202.117.1.13	192.168.1.106	HTTP	388	HTTP/1.1 200 OK

【http请求分析】

首先以xjtu官网为例：

Time	Source	Destination	Protocol	Length	Info
1947 8.818708	192.168.1.106	202.117.1.13	HTTP	601	GET /system/resource/code/datainput.jsp?owner=1151962237&e=1&w=1920&h=
1949 8.822501	202.117.1.13	192.168.1.106	HTTP	388	HTTP/1.1 200 OK

可以看到http请求应答非常少，这就可以看出缓存的效果。

有缓存和没有缓存最主要的区别就在于http这里，然后筛选出http的请求报文如下：

495	2.732392	192.168.1.106	117.34.47.242	HTTP	361		GET /json/xl_chrome_ext_config.json HTT
537	2.891718	192.168.1.106	180.163.203.14	HTTP	590		GET /?xlbtid=1&aid=1022&id=920&peerid=6
540	2.892849	192.168.1.106	180.163.203.14	HTTP	599		GET /?xlbtid=1&aid=1022&id=916&peerid=6
662	3.633734	192.168.1.106	239.255.255.250	SSDP	215		M-SEARCH * HTTP/1.1
855	4.634909	192.168.1.106	239.255.255.250	SSDP	215		M-SEARCH * HTTP/1.1
968	5.635285	192.168.1.106	239.255.255.250	SSDP	215		M-SEARCH * HTTP/1.1
1200	6.476421	192.168.1.106	42.81.183.241	HTTP	282		POST /cgi-bin/httpconn HTTP/1.1
1915	10.465585	192.168.1.106	202.117.1.13	HTTP	588		GET /system/resource/code/datainput.jsp
4090	23.285508	192.168.1.106	42.81.183.241	HTTP	282		POST /cgi-bin/httpconn HTTP/1.1
5564	26.268015	192.168.1.106	111.115.76.75	HTTP	553		GET /system/resource/code/datainput.jsp
15007	38.108948	192.168.1.106	146.232.21.213	HTTP	631		GET /english HTTP/1.1
15147	38.533642	192.168.1.106	146.232.21.213	HTTP	710		GET /Style%20Library/SUN/Fonts/Raleway-
15241	38.708909	192.168.1.106	146.232.21.213	HTTP	711		GET /Style%20Library/SUN/Fonts/Raleway-
15296	38.811957	192.168.1.106	146.232.66.121	HTTP	854		POST /piwik.php?action_name=Stellenbosc
15335	38.830654	192.168.1.106	146.232.21.213	HTTP	712	0.000653000	GET /Style%20Library/SUN/Fonts/Raleway-
15409	38.994803	192.168.1.106	146.232.21.213	HTTP	709	0.000547000	GET /Style%20Library/SUN/Fonts/Raleway-

可以看到访问三个网站的HTTP请求仅仅有10个包，因此缓存大大提高了访问网站的速度。

【http缓存分析】

```
    TCP payload (582 bytes)
▾ Hypertext Transfer Protocol
  ▾ HTTP/1.1 304 NOT MODIFIED\r\n
    › [Expert Info (Chat/Sequence): HTTP/1.1 304 NOT MODIFIED\r\n]
      Response Version: HTTP/1.1
      Status Code: 304
      [Status Code Description: Not Modified]
      Response Phrase: NOT MODIFIED
    Cache-Control: private,max-age=0\r\n
    Expires: Sat, 05 Mar 2022 06:58:50 GMT\r\n
    Accept-Ranges: bytes\r\n
    Server: Microsoft-IIS/10.0\r\n
    X-SharePointHealthScore: 0\r\n
    Public-Extension: http://schemas.microsoft.com/repl-2\r\n
    SPRequestGuid: bf842ba0-6503-509f-4cff-d433793e0162\r\n
    request-id: bf842ba0-6503-509f-4cff-d433793e0162\r\n
    X-FRAME-OPTIONS: SAMEORIGIN\r\n
    SPRequestDuration: 9\r\n
```

如图，返回的304Not Modified就是使用了http缓存技术，并且是使用的协商缓存，也就是客户端主动询问服务器可以不可以使用缓存，服务器返回一个Not Modified，就是可以直接使用缓存。

```
▾ Content-Length: 682\r\n
    [Content length: 682]
  Connection: keep-alive\r\n
  Date: Sun, 20 Mar 2022 06:07:37 GMT\r\n
  Last-Modified: Wed, 02 Mar 2022 02:54:31 GMT\r\n
  ETag: "621edc67-2aa"\r\n
  Expires: Sun, 20 Mar 2022 07:07:37 GMT\r\n
  Cache-Control: max-age=3600\r\n
  Access-Control-Allow-Origin: *\r\n
  Accept-Ranges: bytes\r\n
  Ali-Swift-Global-Savetime: 1647756457\r\n
  Via: cache8.12cn1822[0,0,304-0,H], cache54.12cn1822[1,0], vcache14.cn2328[0,0,200-0,H], vcache3.cn2328[2,0]\r\n
  Age: 3038\r\n
  X-Cache: HIT TCP_MEM_HIT dirn:0:208644898\r\n
  X-Swift-SaveTime: Sun, 20 Mar 2022 06:07:39 GMT\r\n
```

注意以上应答报文的这四个字段，这四个字段实际上都是为了进行HTTP缓存的。

`Last-Modified`和`ETag`相当于服务器先告诉客户端最后改变的时间，客户端自己要记下来，然后再次发送请求的时候要带上这些字段，服务器会进行相应的比较，来判断是否使用缓存，这属于HTTP的协商缓存。

`Expires`和`Cache-Control`相当于服务器直接告诉客户端什么时候可以直接使用缓存，不用问它了。这属于强制缓存。

（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。

这里以访问xjtu主页为例，首先输入：

```
ncat -C www.xjtu.edu.cn 80
```

然后命令行会提示继续输入，再输入：

```
GET / HTTP/1.1
Host: www.xjtu.edu.cn

```

注意这里要迅速点一下回车，然后就会迅速出现大量内容，经过观察，这显然是html网页内容，说明返回成功。

```
root@ubuntu:/home/hijack# nc -C www.xjtu.edu.cn 80
GET / HTTP/1.1
Host: www.xjtu.edu.cn

HTTP/1.1 200 OK
Date: Sun, 20 Mar 2022 09:27:13 GMT
Server: VWebServer
X-Frame-Options: SAMEORIGIN
Last-Modified: Sat, 19 Mar 2022 15:56:48 GMT
ETag: "e0a3-5da944f89d000"
Accept-Ranges: bytes
Content-Length: 57507
Cache-Control: max-age=600
Expires: Sun, 20 Mar 2022 09:37:13 GMT
Vary: Accept-Encoding
Content-Type: text/html
Content-Language: zh-CN

<!DOCTYPE HTML>
<HTML><HEAD><TITLE>西安交通大学</TITLE>



<META name="360-site-verification" content="845cb73defc117caad1186ca8fac8532"><script type="text/javascript">

if(/AppleWebKit.*Mobile/i.test(navigator.userAgent) ||(/MIDP|SymbianOS|NOKIA|SAMSUNG|LG|NEC|TCL|Alcatel|BIRD|DBTEL|Dop
test(navigator.userAgent))){
    if(window.location.href.indexOf("?mobile")<0){
        try{
            if(/Android|Windows Phone|webOS|iPhone|iPod|BlackBerry/i.test(navigator.userAgent)){
                window.location.href="http://mob.xjtu.edu.cn/";
            }
        }catch(e){}
    }
```

2. FTP协议分析(省略)

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】

HTTP缓存分为强制缓存和协商缓存两种，强制缓存客户端不用发起请求，直接使用缓存，协商缓存是每次是否使用缓存都需要与服务器端进行询问请求。

强制缓存的常见技术有`Expires`和`Cache-Control`，其中`Expires`的值是一个时间，表示这个时间前缓存都是有效的，不需要请求；`Cache-Control`有很多属性值，常用的有`max-age`设置了资源的有效时间，这个时间不到都不需要发出请求，另外`immutable`也是`Cache-Control`的一个属性，表示这个资源一直都不用再请求了，就是永远都不会改变，直接使用缓存就行。（`max-age`比`Expire`优先级高）

协商缓存的常见技术有`ETag`和`Last-Modified`，`ETag`其实就是给资源算一个hash值或者版本号，对应的常用`request header`为`If-Modified-Since`。`Last-Modified`是加上资源修改的时间。

强制缓存和协商缓存都存在的情况下，先判断强制缓存是否生效，如果生效不用发起请求，直接使用缓存，如果强制缓存不生效再判断协商缓存。

【DNS缓存】

有DNS的地方就会有DNS缓存，例如操作系统、浏览器、DNS服务器，首先要了解DNS缓存的原理。

这里以浏览器访问网站为例，说明DNS解析中缓存的作用：

1. 首先搜索浏览器自身的DNS缓存,如果存在，则域名解析到此完成。
2. 如果浏览器自身的缓存里面没有找到对应的条目，那么会尝试读取操作系统的hosts文件看是否存在对应的映射关系,如果存在，则域名解析到此完成。
3. 如果本地hosts文件不存在映射关系，则查找本地DNS服务器(ISP服务器,或者自己手动设置的DNS服务器),如果存在,域名到此解析完成。
4. 如果本地DNS服务器还没找到的话,它就会向根服务器发出请求,进行递归查询。

不管是DNS缓存还是HTTP缓存都可以大大提高访问速度。

2、NAT对FTP传输的影响，比较HTTP与FTP的特点；

NAT使得同一个公网IP可以衍生出多个内网IP，这对于FTP传输的影响主要是端口的转换，需要将理论上的21、20端口转换为实际使用的端口，因为同一个网关下的多个主机不可能公用一个20、21端口。主要问题就是要解决FTP的NAT穿越。

【比较HTTP和FTP】

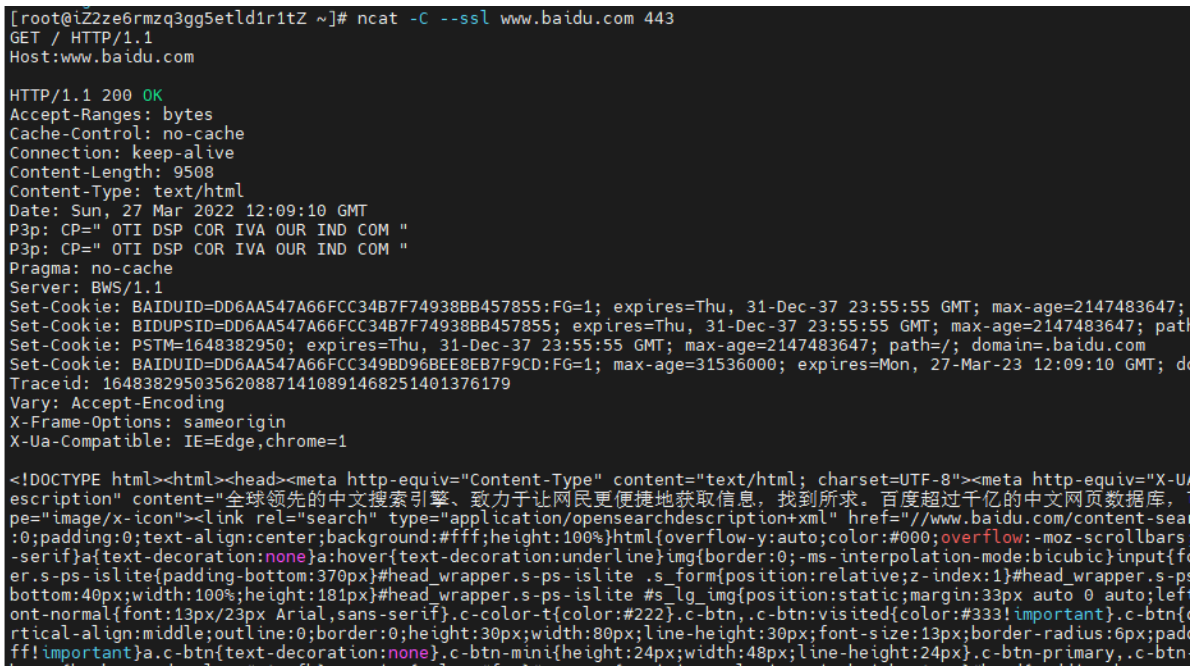
1. HTTP仅需要建立一条TCP连接即可，FTP需要建立两条，分别是控制连接和数据连接。
2. FTP协议有两种工作方式，主动式和被动式。二者区别在于数据连接的建立方式不同。
3. HTTP是基于请求响应的方式的，通过请求URL进行资源的定位，返回资源。

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。

首先访问百度：

```
ncat -C --ssl www.baidu.com 443
GET / HTTP/1.1
Host:www.baidu.com
```



访问gitee

```
ncat -C --ssl www.gitee.com 443
GET / HTTP/1.1
Host:www.gitee.com
```

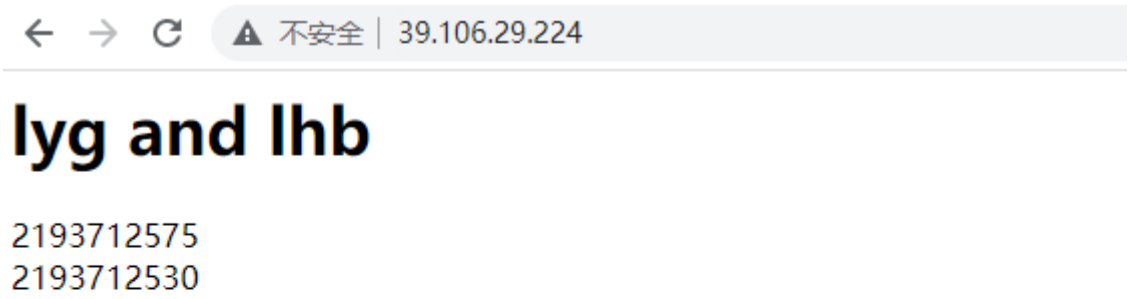
```
[root@iZ2ze6rmzq3gg5etld1r1tZ ~]# ncat -C --ssl www.gitee.com 443
GET / HTTP/1.1
Host:gitee.com

HTTP/1.1 200 OK
Date: Sun, 27 Mar 2022 12:10:16 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Server: nginx
Vary: Accept-Encoding
X-XSS-Protection: 1; mode=block
X-Content-Type-Options: nosniff
X-UA-Compatible: chrome=1
Expires: Sun, 1 Jan 2000 01:00:00 GMT
Pragma: must-revalidate, no-cache, private
Cache-Control: no-cache
Set-Cookie: user_locale=zh-CN; domain=.gitee.com; path=/; expires=Thu, 27 Mar 2042 12:10:16 -0000
Set-Cookie: oschina_new_user=false; path=/; expires=Thu, 27 Mar 2042 12:10:16 -0000
Set-Cookie: gitee-session-n=M0Nma21BRGRmcDVMV2VlQi96Qm1CR2RJQW5zbHN6ZU9RaUtONzNzTmJzSDFoTkoxMFprZ0lOaHdXTUpvRWgzZjB6RWVkOVdhd
NFI4WkxueEcrbWZ5MkJyTDVobnlRZHZ4eEdkUXJLQ0xIYVBwQ0dYYkpidklTbXlQcHMwYVZtUkQyLS1XMWVwcWp4b250ZFlScVNXY2RtY25BPT0%3D--c1089fbc1
X-Request-Id: 5c219249dc6c16ed2fba52b74653b64e
X-Runtime: 0.207373
X-Frame-Options: SAMEORIGIN

bb2
<!DOCTYPE html>
<html lang='zh-CN'>
<head>
<script src="https://assets.gitee.com/assets/static/sentry-5.1.0-a823fb0be1b61c5d7ca4a89f0536cb0a.js"></script>
<script src="/static/javascripts/polyfill-7.4.3.min.js"></script>
<title>Gitee - 基于 Git 的代码托管和研发协作平台</title>
<meta charset='utf-8'>
<meta content='always' name='referrer'>
<meta content='Gitee' property='og:site_name'>
<meta content='Object' property='og:type'>
<meta content='http://gitee.com/' property='og:url'>
<meta content='https://gitee.com/static/images/logo_themecolor.png' itemprop='image' property='og:image'>
<meta content='Gitee - 基于 Git 的代码托管和研发协作平台' itemprop='name' property='og:title'>
<meta content='Gitee.com（码云） 是 OSCHINA.NET 推出的代码托管平台，支持 Git 和 SVN，提供免费的私有仓库托管。目前已有超过 800
<meta content='码云,Gitee,代码托管,git,开源,内源,开源项目托管,免费代码托管,Git代码托管,企业代码管理' name='Keywords'>
<meta content='Gitee.com（码云） 是 OSCHINA.NET 推出的代码托管平台，支持 Git 和 SVN，提供免费的私有仓库托管。目前已有超过 800
<meta content='pc,mobile' name='applicable-device'>
```

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。

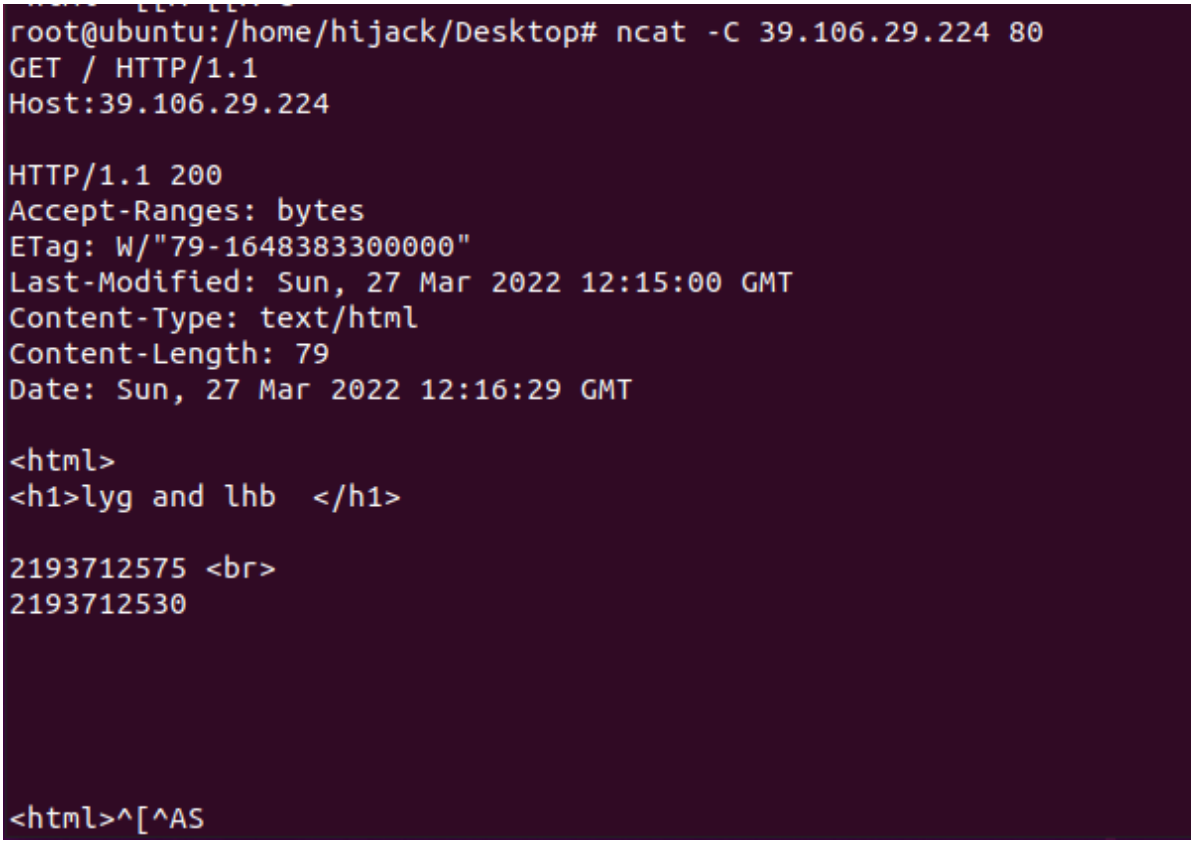
将页面改为：



在虚拟机中进行访问:

```
ncat -C --ssl 39.106.29.224 443
GET / HTTP/1.1
Host:39.106.29.224
```

观察得到的结果：



可以得到我们修改后的html文件。