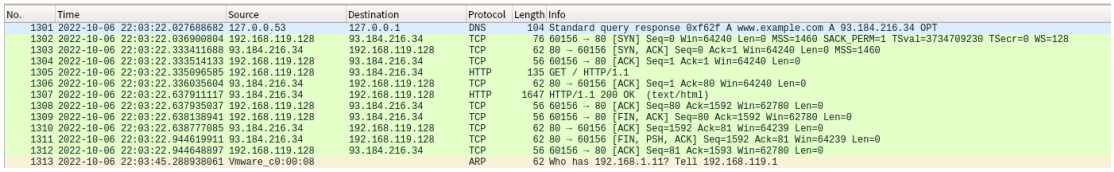
计算机网络第一次实验：

实验内容及数据处理

一．TCP

启动 Wireshark,开始监听数据包。在终端中执行 curl -4 http://www.example.com 指令，并从 Wireshark 中找到 TCP 建立时三次握手以及连接释放时的数据包，分别截图并填写下表。Seq 号和 Ack 号填相对的，Flags 填括号里的部分。

实验现象：



No.	Time	Source	Destination	Protocol	Length	Info
1301	2022-10-06 22:03:22.027688682	127.0.0.53	127.0.0.1	DNS	104	Standard query response 0xf62f A www.example.com A 93.184.216.34 OPT
1302	2022-10-06 22:03:22.036900804	192.168.119.128	93.184.216.34	TCP	76	60156 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 TSval=3734709230 TSecr=0 WS=128
1303	2022-10-06 22:03:22.333411688	93.184.216.34	192.168.119.128	TCP	62	80 → 60156 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
1304	2022-10-06 22:03:22.333514133	192.168.119.128	93.184.216.34	TCP	56	60156 → 80 [ACK] Seq=1 Ack=1 Win=64240 Len=0
1305	2022-10-06 22:03:22.335096585	192.168.119.128	93.184.216.34	HTTP	135	GET / HTTP/1.1
1306	2022-10-06 22:03:22.336035604	93.184.216.34	192.168.119.128	TCP	62	80 → 60156 [ACK] Seq=1 Ack=80 Win=64240 Len=0
1307	2022-10-06 22:03:22.637911117	93.184.216.34	192.168.119.128	HTTP	1647	HTTP/1.1 200 OK (text/html)
1308	2022-10-06 22:03:22.637935037	192.168.119.128	93.184.216.34	TCP	56	60156 → 80 [ACK] Seq=80 Ack=1592 Win=62780 Len=0
1309	2022-10-06 22:03:22.638138941	192.168.119.128	93.184.216.34	TCP	56	60156 → 80 [FIN, ACK] Seq=80 Ack=1592 Win=62780 Len=0
1310	2022-10-06 22:03:22.638777085	93.184.216.34	192.168.119.128	TCP	62	80 → 60156 [ACK] Seq=1592 Ack=81 Win=64239 Len=0
1311	2022-10-06 22:03:22.944619911	93.184.216.34	192.168.119.128	TCP	62	80 → 60156 [FIN, PSH, ACK] Seq=1592 Ack=81 Win=64239 Len=0
1312	2022-10-06 22:03:22.944648897	192.168.119.128	93.184.216.34	TCP	56	60156 → 80 [ACK] Seq=81 Ack=1593 Win=62780 Len=0
1313	2022-10-06 22:03:45.288938061	Vmware_c0:00:08		ARP	62	Who has 192.168.1.1? Tell 192.168.119.1

Figure 1 **监听到的数据包列表**

根据摘要填写下表：

项目	IP 地址	端口号
发送方 IP 地址和端口号	192.168.119.128	60156
接收方 IP 地址和端口号	93.184.216.34	80

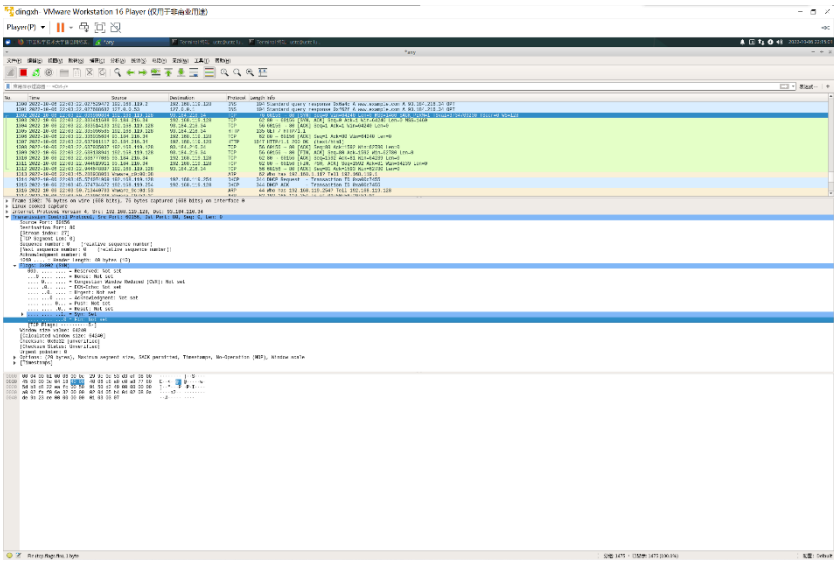


Figure 2 **数据包内容：（以握手包 1 为例）**

项目	握手包 1	握手包 2	握手包 3	释放包 1	释放包 2	释放包 3	释放包 4
Seq 号	0	0	1	80	1592	1592	81
Ack 号	0	1	1	1592	81	81	1593
Flags	SYN	SYN,ACK	ACK	FIN,ACK	ACK	FIN,PSH,ACK	ACK
Window	64240	64240	64240	62780	64239	64239	62780

（实验文档所给表格只有 3 个释放包，了解到 TCP 释放连接过程一般需要四次握手，这与实际监听到的情况也比较符合，所以在原表格基础上增加了“释放包 4”）

二． HTTP/HTTPS

在终端中执行 curl -v http://www.example.com 指令，该指令会显示详细的请求报文和响应报文。可以用 Web 浏览器访问网址 http://www.example.com，结合显示的网页理解报文的内容，注意观察 HTTP 协议版本、HTTP 方法类型、状态码与内容类型

启动 Wireshark，开始监听数据包，在终端中分别执行下表中的指令，并从 Wireshark 中找到对应的 HTTP/HTTPS 数据包，分别截图并填写下表。

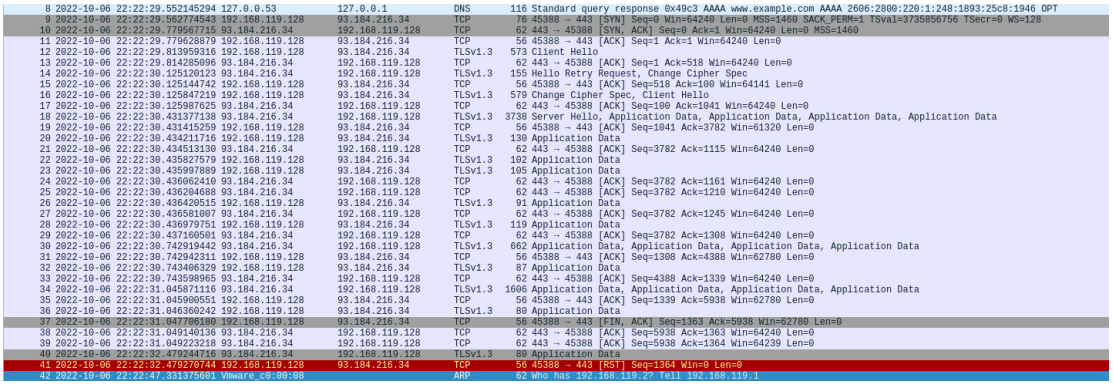


Figure 3 curl -v https://www.example.com **数据包列表**

```
> GET / HTTP/2
> Host: www.example.com
> User-Agent: curl/7.58.0
> Accept: */*
>
* TLSv1.3 (IN), TLS Unknown, Certificate Status (22):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS Unknown, Certificate Status (22):
* TLSv1.3 (IN), TLS handshake, Newsession Ticket (4):
* TLSv1.3 (IN), TLS Unknown, Unknown (23):
* Connection state changed (MAX_CONCURRENT_STREAMS updated)!
* TLSv1.3 (OUT), TLS Unknown, Unknown (23):
* TLSv1.3 (IN), TLS Unknown, Unknown (23):
* TLSv1.3 (IN), TLS Unknown, Unknown (23):
* TLSv1.3 (IN), TLS Unknown, Unknown (23):
< HTTP/2 200
< age: 553056
< cache-control: max-age=604800
< content-type: text/html; charset=UTF-8
< date: Fri, 07 Oct 2022 15:20:25 GMT
< etag: "3147526947+ident"
< expires: Fri, 14 Oct 2022 15:20:25 GMT
< last-modified: Thu, 17 Oct 2019 07:18:26 GMT
< server: ECS (sab/56F3)
< vary: Accept-Encoding
< x-cache: HIT
< content-length: 1256
```

Figure 4 curl -v https://www.example.com（Wireshark 中抓取的数据包无法查看协议类型及内容类型等信息，以上图片来自终端）

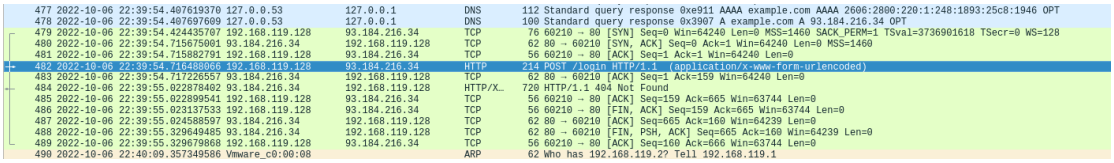


Figure 5 curl -v -d “user=test” -X POST http://www.example.com **数据包列表**

指令	协议版本	方法类型	状态码	内容类型
curl -v http://www.example.com	HTTP/1.1	GET	200	text/html
curl -v https://www.example.com	HTTPS	GET	200	text/html
curl -v -d “user=test” -X POST http://www.example.com	HTTP/1.1	POST	404	text/html

三． DNS

1. 启动 Wireshark，开始监听数据包。在终端执行 curl http://www.example.com 指令，并从 Wireshark 中找到对应的 DNS 数据包，分别截图并填写下表。

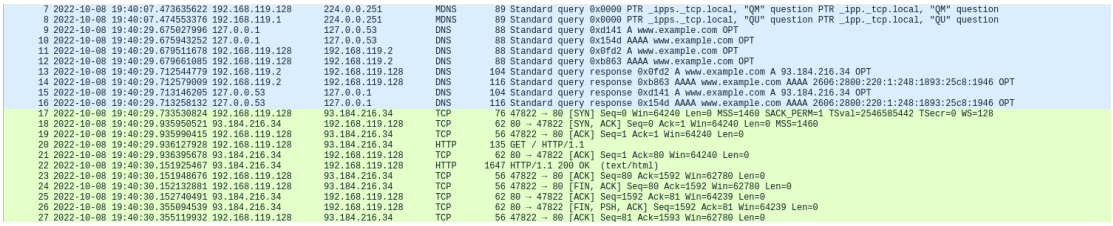


Figure 6 **抓取到的** DNS **数据包**

项目	数据
本机 IP 地址和端口号	IP 地址：192.168.119.128　端口号：60210
DNS 服务器 IP 地址和端口号	IP 地址：192.168.119.2 端口号：53
传输层协议类型	UDP
目标服务器 URI	www.example.com
目标服务器 IP 地址	93.184.216.34

2. 指定 8.8.8.8 为 DNS 服务器，根据下面的要求，使用 dig 查询对应的 DNS 记录，并填写缺失的命令和结果。

查询目标	命令	结果
www.baidu.com 的 IP v4 地址	dig www.baidu.com @8.8.8.8	14.215.177.38 14.215.177.39
jw.ustc.edu.cn 的 IPv6 地址	dig -t aaaa jw.ustc.edu.cn @8.8.8.8	2001:da8:d800:642::248
202.38.75.11 的域名	dig -x 202.38.75.11 @8.8.8.8	infonet.ustc.edu.cn
mail.ustc.edu.cn 的邮件交换记录（MX）	dig -t mx mail.ustc.edu.cn	smtp2.ustc.edu.cn. smtp1.ustc.edu.cn. smtp.ustc.edu.cn.
i.ustc.edu.cn 的 CNAME	dig -t cname i.ustc.edu.cn	revproxy.ustc.edu.cn.
example.com 的域名服务器	dig -t ns example.com	a.iana-servers.net. b.iana-servers.net.

3. dig 不带域名可以用来查询根域名服务器，截图查询结果。

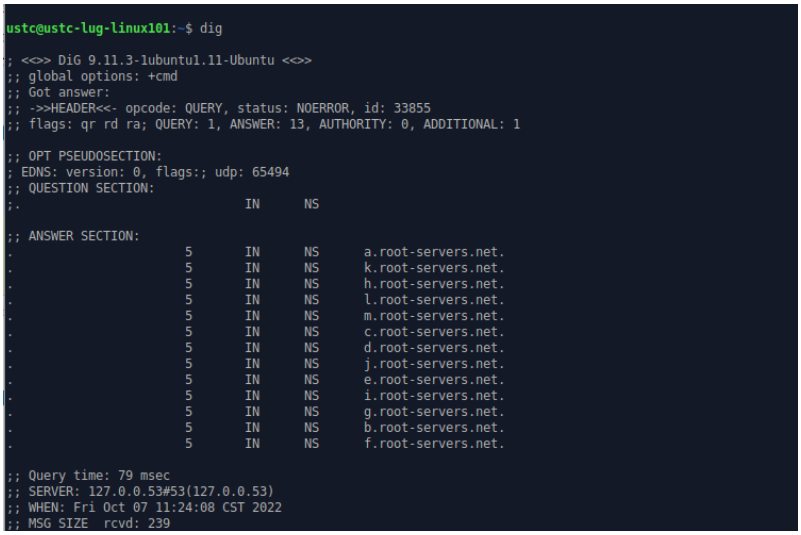


Figure 7 dig **不带域名查询跟服务器**

4. 使用 dig,发起对<your-student-id>.ustc.edu.cn 的查询请求，能获得有效的 DNS 查询结果吗？记录回复的 status 字段，截图说明。
   不能获得有效的 DNS 查询结果，因为不存在对应的域名。
   status: NXDOMAIN

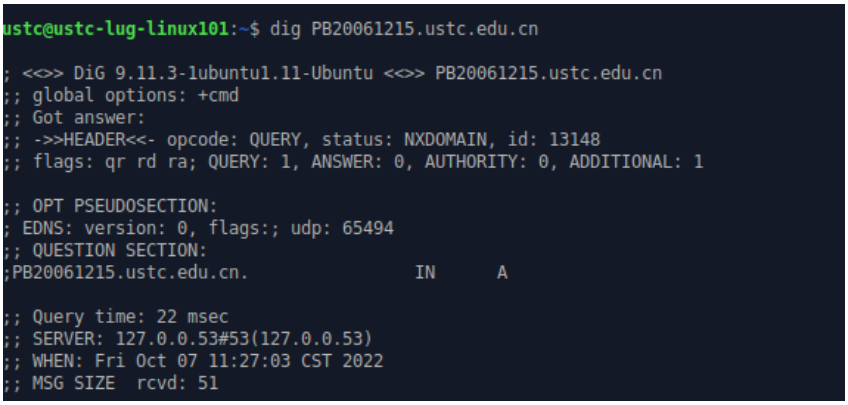


Figure 8 DNS **未获得有效结果**

四．FTP

1. 主动模式：

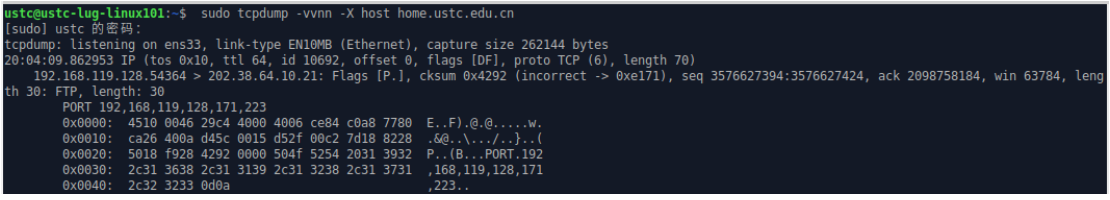


Figure 9 **主动模式下的数据包**

注意到：PORT 后的参数为：192，168，119，128，171，223
计算得到端口号为：171*256+223=43999

2. 被动模式

```
23:52:53.555394 IP (tos 0x0, ttl 128, id 9793, offset 0, flags [none], proto TCP (6), length 89)
    202.38.64.10.21 > 192.168.119.128.39188: Flags [P.], cksum 0x63c2 (correct), seq 89:138, ack 41, win 64238, length 49: FTP, length: 49
        227 Entering Passive Mode (202,38,64,10,125,81)
        0x0000:  4500 0059 2641 0000 8006 d204 ca26 400a  E..Y&A.......&@.
        0x0010:  c0a8 7780 0015 9914 1fa8 05a5 dc26 eea5  ..w..........&..
        0x0020:  5018 faee 63c2 0000 3232 3720 456e 7465  P...c...227.Ente
        0x0030:  7269 6e67 2050 6173 7369 7665 204d 6f64  ring.Passive.Mod
        0x0040:  6520 2832 3032 2c33 382c 3634 2c31 302c  e.(202,38,64,10,
        0x0050:  3132 352c 3831 290d 0a                   125,81)..
23:52:53.555417 IP (tos 0x10, ttl 64, id 5804, offset 0, flags [DF], proto TCP (6), length 40)
    192.168.119.128.39188 > 202.38.64.10.21: Flags [.], cksum 0x4274 (incorrect -> 0xeadd), seq 41, ack 138, win 63784, length 0
        0x0000:  4510 0028 16ac 4000 4006 e1ba c0a8 7780  E..(..@.@.....w.
        0x0010:  ca26 400a 9914 0015 dc26 eea5 1fa8 05d6  .&@......&......
        0x0020:  5010 f928 4274 0000                      P..(Bt..
23:52:53.555661 IP (tos 0x0, ttl 64, id 4835, offset 0, flags [DF], proto TCP (6), length 60)
    192.168.119.128.59340 > 202.38.64.10.32081: Flags [S], cksum 0x4288 (incorrect -> 0xe282), seq 2468915186, win 64240, options [mss 1460,sackOK,TS v
al 3424828374 ecr 0,nop,wscale 7], length 0
        0x0000:  4500 003c 12e3 4000 4006 e57f c0a8 7780  E..<..@.@.....w.
        0x0010:  ca26 400a e7cc 7d51 9328 a7f2 0000 0000  .&@...}Q.(......
        0x0020:  a002 faf0 4288 0000 0204 05b4 0402 080a  ....B...........
        0x0030:  cc22 bbd6 0000 0000 0103 0307            ."............
23:52:53.558555 IP (tos 0x0, ttl 128, id 9794, offset 0, flags [none], proto TCP (6), length 44)
    202.38.64.10.32081 > 192.168.119.128.59340: Flags [S.], cksum 0x2315 (correct), seq 1015044860, ack 2468915187, win 64240, options [mss 1460], leng
th 0
        0x0000:  4500 002c 2642 0000 8006 d230 ca26 400a  E..,&B.....0.&@.
        0x0010:  c0a8 7780 7d51 e7cc 3c80 5afc 9328 a7f3  ..w.}Q..<.Z..(..
        0x0020:  6012 faf0 2315 0000 0204 05b4 0000       `...#.........
23:52:53.558583 IP (tos 0x0, ttl 64, id 4836, offset 0, flags [DF], proto TCP (6), length 40)
    192.168.119.128.59340 > 202.38.64.10.32081: Flags [.], cksum 0x4274 (incorrect -> 0x3ad2), seq 1, ack 1, win 64240, length 0
        0x0000:  4500 0028 12e4 4000 4006 e592 c0a8 7780  E..(..@.@.....w.
        0x0010:  ca26 400a e7cc 7d51 9328 a7f3 3c80 5afd  .&@...}Q.(..<.Z.
        0x0020:  5010 faf0 4274 0000                      P...Bt..
23:52:53.559135 IP (tos 0x10, ttl 64, id 5805, offset 0, flags [DF], proto TCP (6), length 46)
    192.168.119.128.39188 > 202.38.64.10.21: Flags [P.], cksum 0x427a (incorrect -> 0x3e28), seq 41:47, ack 138, win 63784, length 6: FTP, length: 6
        LIST
        0x0000:  4510 002e 16ad 4000 4006 e1b3 c0a8 7780  E.....@.@.....w.
        0x0010:  ca26 400a 9914 0015 dc26 eea5 1fa8 05d6  .&@......&......
        0x0020:  5018 f928 427a 0000 4c49 5354 0d0a       P..(Bz..LIST..
23:52:53.559325 IP (tos 0x0, ttl 128, id 9795, offset 0, flags [none], proto TCP (6), length 40)
    202.38.64.10.21 > 192.168.119.128.39188: Flags [.], cksum 0xe911 (correct), seq 138, ack 47, win 64238, length 0
        0x0000:  4500 0028 2643 0000 8006 d233 ca26 400a  E..(&C.....3.&@.
        0x0010:  c0a8 7780 0015 9914 1fa8 05d6 dc26 eeab  ..w..........&..
        0x0020:  5010 faee e911 0000 0000 0000 0000       P.............
23:52:53.561893 IP (tos 0x0, ttl 128, id 9796, offset 0, flags [none], proto TCP (6), length 79)
    202.38.64.10.21 > 192.168.119.128.39188: Flags [P.], cksum 0xd3c4 (correct), seq 138:177, ack 47, win 64238, length 39: FTP, length: 39
        150 Here comes the directory listing.
```

注意到：Entering Passive Mode 后的参数为 202,38,64,10,125,81
计算得到端口号：125*256+81=32081
与后续数据包对比，一致。

思考题：

1. 解释 HTTP 中的幂等是什么意思？GET 操作是幂等的吗？POST 呢？
   答：幂等：假如在不考虑诸如错误或者过期等问题情况下，若干次请求的副作用与单次请求相同或根本没有副作用，那么这些请求方法就能够被视为“幂等的”
   GET 方法用于获取资源，不会影响到资源的变化，所以是幂等的。
   POST 方法表示创建资源，两次相同的 POST 请求会在服务器端创建两份资源，他们具有不同的 URI，所以不具备幂等性。
2. HTTPS 抓到的数据包与之前 HTTP 抓到的有什么不同？这是什么原因导致的？
   答：分别对 HTTPS 和 HTTP 协议传输过程中抓取到的数据包鼠标右键单击，选择追踪流->TCP 流，可以看到 HTTP 数据包是以明文传输，而 HTTPS 数据包进行了加密。
   原因：HTTP 是明文传输，直接使用 TCP 进行通信。而 HTTPS 在 HTTP 和 TCP 之间引入安全套接字（SSL）进行信息交换，属于加密传输。
3. FTP 实验用的 tcpdump 指令整体可以达到什么效果？每个参数的含义分别是什么？
   答：该指令整体的效果：抓取主机和 home.ustc.edu.cn 之间传输的数据包并进行分析和显示。
   -vv:输出详细的报文信息
   -nn:指定将每个监听到的数据包中的域名转换成 IP、端口从应用名称转换成端口号后显示
   -X: 告诉 `tcpdump` 命令，需要把协议头和包内容都原原本本的显示出来
4. 解释从输入网址，到浏览器显示网页，在应用层依次发生了什么？
   答：首先，应用层根据输入的 URL 调用 DNS，查询得到对应的 IP 地址，然后将 IP 交给

传输层，调用 TCP 服务与目标地址建立连接。连接建立后，根据 HTTP/HTTPS 协议，向代理服务器发送请求。如果代理服务器中不含相应资源，则向 Web 服务器转发请求报文。代理服务器/Web 服务器收到请求，向浏览器发送相应资源。浏览器收到响应报文后，解析 html 文件，将其显示出来。