

计算机网络与应用

实验一 利用 Wireshark 分析网络协议

学 院： _____ 计软智学院 _____

专 业： _____ 人工智能专业 _____

学 号： _____ 58122310 _____

姓 名： _____ 唐梓烨 _____

二 0 二 四 年 五

一、实验目的

1. 加强对计算机网络通信协议的理解，用理论知识解决实际问题。
2. 学习 Wireshark 的基本操作，抓取和分析有线局域网的数据包，熟悉一些应用层命令和协议。
3. 通过运用 Wireshark 对网络活动进行分析，观察 TCP 协议报文，分析通信时序，理解 TCP 的工作过程，掌握 TCP 工作原理与实现；学会运用 Wireshark 分析 TCP 连接管理、流量控制和拥塞控制的过程，发现 TCP 的性能问题。

二、实验任务

（一）应用层协议分析

1. 学会使用 Wireshark 抓包软件，会使用过滤器。
2. 学习 Wireshark 基本操作：重点掌握捕获过滤器和显示过滤器。分析 HTTP 和 DNS 协议。
3. 测试 curl 命令，访问一个 web 页面。（选做）
4. 利用 telnet 命令测试 get 命令，访问 www.baidu.com。（选做）
5. 利用 telnet 命令测试 SMTP 服务，解析其过程。（选做）
6. 测试 tracert 命令，并解析其过程。
7. 使用 nslookup 查询域名信息，简要分析。

（二）传输层协议分析

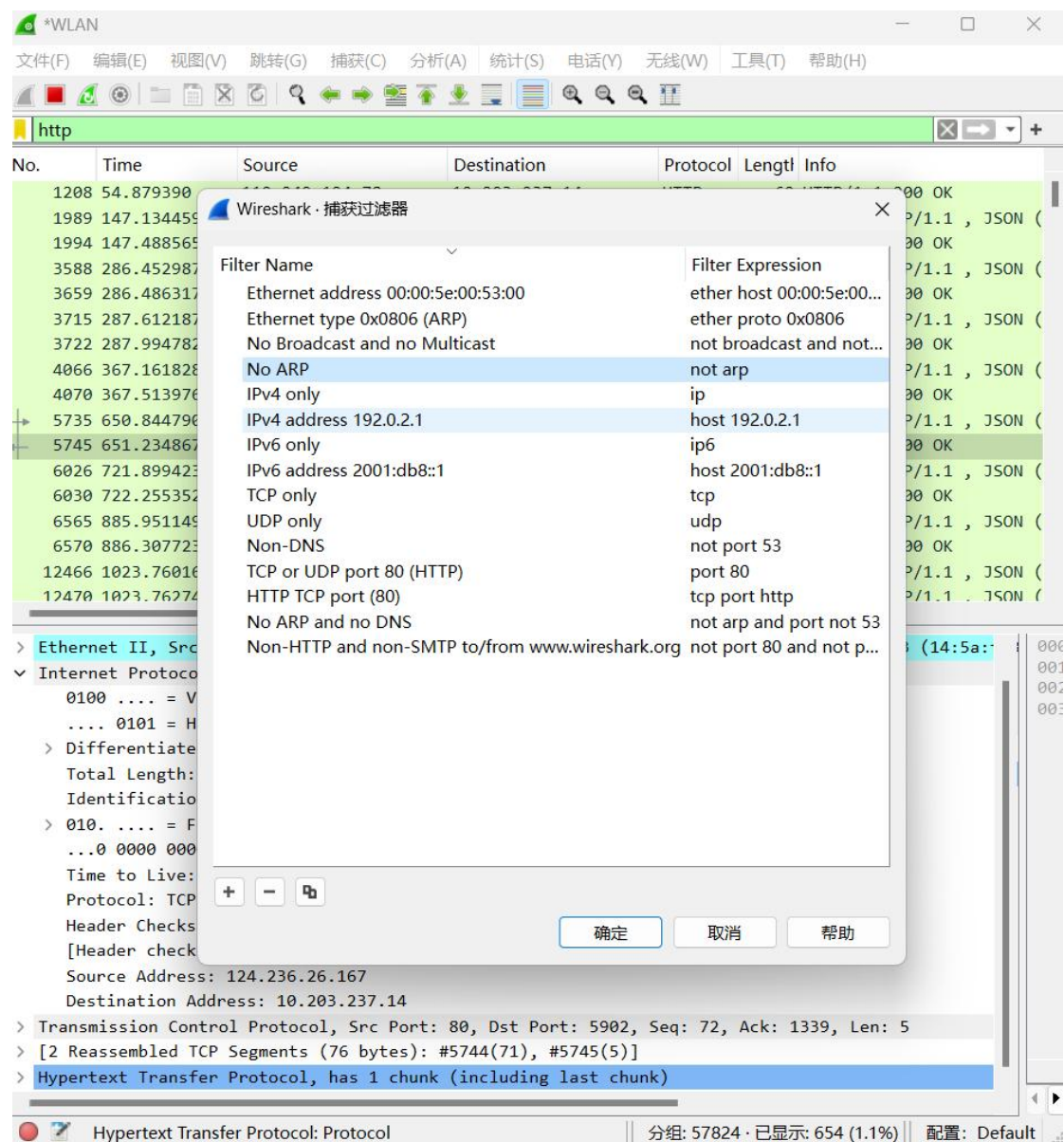
1. TCP 数据流的追踪。
2. TCP 连接的建立。
3. TCP 连接的终止。
4. TCP 连接的重置。
5. 两台实验机本地相互连接，在实验机中仿真不同的网络条件，观察 TCP 的各种控制现象。（选做）

三、实验内容（含解析）

（依次描述每个任务的完成过程：包括执行命令语句、过程截图等）

（一）应用层协议分析

1. 设置捕获过滤器



在“捕获→捕获过滤器”中可以设置 Wireshark 的捕获过滤器，捕获过滤器的作用是在抓包捕获时进行筛选。Wireshark 提供了如图所示的捕获过滤器设置预设，用户可自行选择，还可按左下角+号自行添加。选择 No ARP 后重新开始捕获，即可将 arp 数据包排除在外。

2. 设置显示过滤器

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.203.237.14	1.12.12.12	TCP	55	6721 → 443 [ACK] Seq=1 Ac
2	0.055832	1.12.12.12	10.203.237.14	TCP	66	443 → 6721 [ACK] Seq=1 Ac
3	3.443007	10.203.237.14	113.240.75.249	TCP	66	6993 → 443 [SYN] Seq=0 Wi
4	3.533942	113.240.75.249	10.203.237.14	TCP	66	443 → 6993 [SYN, ACK] Seq
5	3.534043	10.203.237.14	113.240.75.249	TCP	54	6993 → 443 [ACK] Seq=1 Ac
6	3.534324	10.203.237.14	113.240.75.249	TLSv1.2	571	Client Hello (SNI=tpstele
7	3.873739	10.203.237.14	113.240.75.249	TCP	571	[TCP Retransmission] 6993
8	3.893370	113.240.75.249	10.203.237.14	TCP	60	443 → 6993 [ACK] Seq=1 Ac
9	3.893370	113.240.75.249	10.203.237.14	TLSv1.2	195	Server Hello, Change Ciph
10	3.893370	113.240.75.249	10.203.237.14	TCP	195	[TCP Retransmission] 443
11	3.893440	10.203.237.14	113.240.75.249	TCP	66	6993 → 443 [ACK] Seq=518
12	3.893704	10.203.237.14	113.240.75.249	TLSv1.2	1494	Change Cipher Spec, Encry
13	3.893704	10.203.237.14	113.240.75.249	TCP	1494	6993 → 443 [ACK] Seq=1958
14	3.893704	10.203.237.14	113.240.75.249	TCP	1494	6993 → 443 [ACK] Seq=3398
15	3.893704	10.203.237.14	113.240.75.249	TCP	1494	6993 → 443 [ACK] Seq=4838
16	3.893704	10.203.237.14	113.240.75.249	TCP	1494	6993 → 443 [ACK] Seq=6278
17	3.893704	10.203.237.14	113.240.75.249	TLSv1.2	700	Application Data
18	3.918329	113.240.75.249	10.203.237.14	TCP	66	[TCP Dup ACK 8#1] 443 → 6
19	3.920475	113.240.75.249	10.203.237.14	TCP	60	443 → 6993 [ACK] Seq=142
20	3.922010	113.240.75.249	10.203.237.14	TCP	60	443 → 6993 [ACK] Seq=142

在软件上方输入 '!arp' 即可将捕获的数据包当中的 arp 数据包排除显示，要筛选 IP 可使用 ip.addr==?，要筛选协议可直接键入 http, tcp 等，十分方便。

3. 分析 http 协议

2538	182.631477	10.203.237.14	36.155.132.3	HTTP	130	GET / HTTP/1.1
2541	183.001923	36.155.132.3	10.203.237.14	HTTP	1395	HTTP/1.1 200 OK (text/h
2829	240.143266	10.203.237.14	104.123.71.90	HTTP	165	GET /connecttest.txt HTT
2834	240.517172	104.123.71.90	10.203.237.14	HTTP	241	HTTP/1.1 200 OK (text/p
3138	275.643977	10.203.237.14	104.123.71.136	HTTP	165	GET /connecttest.txt HTT
3158	276.029061	104.123.71.136	10.203.237.14	HTTP	241	HTTP/1.1 200 OK (text/p

> Frame 2538: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bits) on interface \Device\NPF_

> Ethernet II, Src: LiteonTechno_74:8a:93 (14:5a:fc:74:8a:93), Dst: IETF-VRRP-VRID_65 (00:00:5e:00:01

> Internet Protocol Version 4, Src: 10.203.237.14, Dst: 36.155.132.3

> Transmission Control Protocol, Src Port: 7117, Dst Port: 80, Seq: 1, Ack: 1, Len: 76

> Hypertext Transfer Protocol

> GET / HTTP/1.1\r\n

Host: www.baidu.com\r\n

User-Agent: curl/8.4.0\r\n

Accept: */*\r\n

\r\n

[Full request URI: http://www.baidu.com/]

[HTTP request 1/1]

[Response in frame: 2541]

这是用户端向 www.baidu.com 发送的 http 请求。具体解释如下：

i. GET / HTTP/1.1\r\n：这一行是请求行，包含三个部分：

GET：请求方法，表示获取 Request URI 所标识的资源。

/: 正在请求的资源；在这个案例中，它是服务器的根文档。

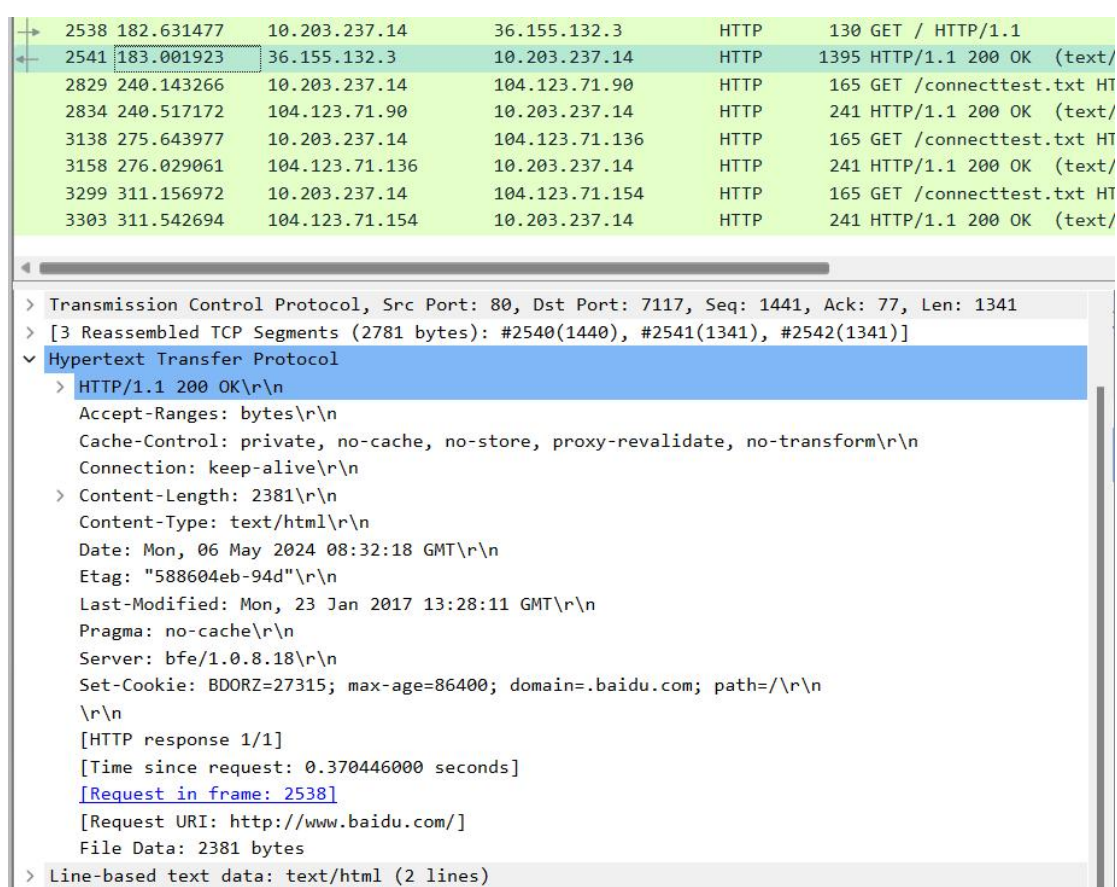
HTTP/1.1: HTTP 协议的版本。

ii. Host: www.baidu.com\r\n: 这一行是一个请求头，指定了要请求的服务器的域名 (www.baidu.com)。

iii. User-Agent: curl/8.4.0\r\n: 这一行是一个请求头，说明发起请求的客户端软件 (curl)。

iv. Accept: */*\r\n: 这一行是请求头，指定客户端能接受哪些类型的数据。

/: 表示客户端可以接受任何类型的数据。



```
2538 182.631477 10.203.237.14 36.155.132.3 HTTP 130 GET / HTTP/1.1
2541 183.001923 36.155.132.3 10.203.237.14 HTTP 1395 HTTP/1.1 200 OK (text/
2829 240.143266 10.203.237.14 104.123.71.90 HTTP 165 GET /connecttest.txt HT
2834 240.517172 104.123.71.90 10.203.237.14 HTTP 241 HTTP/1.1 200 OK (text/
3138 275.643977 10.203.237.14 104.123.71.136 HTTP 165 GET /connecttest.txt HT
3158 276.029061 104.123.71.136 10.203.237.14 HTTP 241 HTTP/1.1 200 OK (text/
3299 311.156972 10.203.237.14 104.123.71.154 HTTP 165 GET /connecttest.txt HT
3303 311.542694 104.123.71.154 10.203.237.14 HTTP 241 HTTP/1.1 200 OK (text/

> Transmission Control Protocol, Src Port: 80, Dst Port: 7117, Seq: 1441, Ack: 77, Len: 1341
> [3 Reassembled TCP Segments (2781 bytes): #2540(1440), #2541(1341), #2542(1341)]
> Hypertext Transfer Protocol
> HTTP/1.1 200 OK\r\n
  Accept-Ranges: bytes\r\n
  Cache-Control: private, no-cache, no-store, proxy-revalidate, no-transform\r\n
  Connection: keep-alive\r\n
> Content-Length: 2381\r\n
  Content-Type: text/html\r\n
  Date: Mon, 06 May 2024 08:32:18 GMT\r\n
  Etag: "588604eb-94d"\r\n
  Last-Modified: Mon, 23 Jan 2017 13:28:11 GMT\r\n
  Pragma: no-cache\r\n
  Server: bfe/1.0.8.18\r\n
  Set-Cookie: BDORZ=27315; max-age=86400; domain=.baidu.com; path=/\r\n
  \r\n
  [HTTP response 1/1]
  [Time since request: 0.370446000 seconds]
  [Request in frame: 2538]
  [Request URI: http://www.baidu.com/]
  File Data: 2381 bytes
> Line-based text data: text/html (2 lines)
```

这是 www.baidu.com 向用户端发送的 http 响应，具体解释如下：

i. HTTP/1.1 200 OK\r\n: 这是状态行，包含三个部分：

HTTP/1.1: 表明响应使用的 HTTP 协议版本。

200: 状态码，表示请求已成功被服务器处理。

OK: 状态消息，是状态码的文本描述，同样表示请求成功。

接下来是响应头部分

ii. Accept-Ranges: bytes\r\n: 表明服务器可以接受的类型范围，这里是字节。

iii. Cache-Control: private, no-cache, no-store, proxy-revalidate, no-transform\r\n: 指定了缓存策略，这里禁止缓存此响应。

iv. Connection: keep-alive\r\n: 表明连接将保持活动状态，允许多个请求或响应在单个连接上交换。

v. Content-Length: 2381\r\n: 指示响应体的字节长度。

vi. Content-Type: text/html\r\n: 指示响应体的媒体类型，这里是 HTML 文档。

vii. Date: Mon, 06 May 2024 08:32:18 GMT\r\n: 响应生成的日期和时间。

viii. Etag: "588604eb-94d"\r\n: 响应的实体标签，用于缓存优化和资源验证。

ix. Last-Modified: Mon, 23 Jan 2017 13:28:11 GMT\r\n: 文档最后被修改的时间。

x. Pragma: no-cache\r\n: 古老的 HTTP/1.0 头部，用来防止缓存响应。

xi. Server: bfe/1.0.8.18\r\n: 响应的服务器软件和版本。

xii. Set-Cookie: BDORZ=27315; max-age=86400; domain=.baidu.com; path=/\r\n: 设置一个名为 BDORZ 的 cookie，有效期为 86400 秒（一天），适用于 `baidu.com` 域下的所有路径。

xiii. \r\n: 这个单独的回车换行符标志着响应头部的结束，之后是响应体。
其他额外信息：

HTTP response 1/1: 标示这是捕获的第一个 HTTP 响应。

Time since request: 0.370446000 seconds: 从请求发送到接收响应的时
间。

Request in frame: 2538: 发起请求的数据帧号。

Request URI: http://www.baidu.com/: 请求的统一资源标识符。

File Data: 2381 bytes: 响应数据的字节数，与 Content-Length 值相同。

4. 分析 DNS 协议

No.	Time	Source	Destination	Protocol	Length	Info
2527	179.915927	172.19.2.1	10.203.237.14	DNS	205	Standard query response
2533	182.476902	10.203.237.14	172.19.2.1	DNS	73	Standard query 0x8252 A
2534	182.617874	172.19.2.1	10.203.237.14	DNS	132	Standard query response
2550	183.460763	10.203.237.14	172.19.2.1	DNS	73	Standard query 0x8252 A

> Frame 2533: 73 bytes on wire (584 bits), 73 bytes captured (584 bits) on interface \Device\NPF_{7BF...}

> Ethernet II, Src: LiteonTechno_74:8a:93 (14:5a:fc:74:8a:93), Dst: IETF-VRRP-VRID_65 (00:00:5e:00:01...

> Internet Protocol Version 4, Src: 10.203.237.14, Dst: 172.19.2.1

> User Datagram Protocol, Src Port: 50450, Dst Port: 53

▼ Domain Name System (query)

Transaction ID: 0x8252

▼ Flags: 0x0100 Standard query

0... .. = Response: Message is a query

.000 0... .. = Opcode: Standard query (0)

.... ..0. = Truncated: Message is not truncated

.... ..1 = Recursion desired: Do query recursively

.... ..0... .. = Z: reserved (0)

.... ..0 = Non-authenticated data: Unacceptable

Questions: 1

Answer RRs: 0

Authority RRs: 0

Additional RRs: 0

▼ Queries

▼ www.baidu.com: type A, class IN

Name: www.baidu.com

[Name Length: 13]

[Label Count: 3]

Type: A (1) (Host Address)

Class: IN (0x0001)

[\[Response In: 2534\]](#)

图示了一个 DNS 查询的请求，用于解析域名 www.baidu.com:

Transaction ID: 0x8252: 查询的事务 ID，用于标识请求和响应，以确保它们匹配。

Flags: 0x0100 Standard query: 这些标志位表明这是一个标准查询。

包括标志了这是一个查询请求而非响应，操作码为 0 表示这是一个标准查询，消息没有被截断，希望递归查询，非认证数据是不可接受的。

查询中包含一个问题，回答资源、权威资源、额外资源都是 0，因为这是查询请求。

Queries: 表示查询问题区域，包含

Name: www.baidu.com: 被查询的域名。

Type: A (1) (Host Address): 查询类型为 A，表示查询的是 IPv4 地址。

Class: IN (0x0001): 类别为 IN，表示是互联网地址。

No.	Time	Source	Destination	Protocol	Length	Info
2527	179.915927	172.19.2.1	10.203.237.14	DNS	205	Standard query response
2533	182.476902	10.203.237.14	172.19.2.1	DNS	73	Standard query 0x8252 A
2534	182.617874	172.19.2.1	10.203.237.14	DNS	132	Standard query response
2550	183.460763	10.203.237.14	172.19.2.1	DNS	73	Standard query 0x6450 A

> Frame 2534: 132 bytes on wire (1056 bits), 132 bytes captured (1056 bits) on interface \Device\NPF

> Ethernet II, Src: IETF-VRRP-VRID_65 (00:00:5e:00:01:65), Dst: LiteonTechno_74:8a:93 (14:5a:fc:74:8a

> Internet Protocol Version 4, Src: 172.19.2.1, Dst: 10.203.237.14

> User Datagram Protocol, Src Port: 53, Dst Port: 50450

▼ Domain Name System (response)

Transaction ID: 0x8252

▼ Flags: 0x8180 Standard query response, No error

1... .. = Response: Message is a response

.000 0... .. = Opcode: Standard query (0)

.... .0.. .. = Authoritative: Server is not an authority for domain

.... ..0. = Truncated: Message is not truncated

.... ..1 = Recursion desired: Do query recursively

.... ..1... .. = Recursion available: Server can do recursive queries

.... ..0.. = Z: reserved (0)

.... ..0. = Answer authenticated: Answer/authority portion was not authenticated by

.... ..0 = Non-authenticated data: Unacceptable

.... ..0000 = Reply code: No error (0)

Questions: 1

Answer RRs: 3

Authority RRs: 0

Additional RRs: 0

▼ Queries

> www.baidu.com: type A, class IN

▼ Answers

> www.baidu.com: type CNAME, class IN, cname www.a.shifen.com

> www.a.shifen.com: type A, class IN, addr 36.155.132.3

> www.a.shifen.com: type A, class IN, addr 36.155.132.76

[Request In: 2533]

[Time: 0.140972000 seconds]

图示了一个 DNS 查询的响应：

Transaction ID: 0x8252: 查询的事务 ID，与前面的请求相匹配

Flags: 0x8180 Standard query response, No error: 这些标志位表明这是一个标准响应，没有错误。

包括标志了这是一个响应，操作码为 0 表示这是一个标准查询，表明服务器不是该域名的权威服务器，消息未被截断，服务器可以进行递归查询，响应未被服务器认证，响应码为 0 表示没有错误。

响应中包含一个问题，回答资源数为 3，权威和额外资源数为 0。

Queries: 查询问题区域，与上面相同。

Answers: 回答问题区域，包含 3 个回答资源

首条回答记录，表明 www.baidu.com 是一个别名，实际的域名是 www.a.shifen.com

第二条回答记录，解析了 www.a.shifen.com 的 IPv4 地址为 36.155.132.3

第三条回答记录，解析了 www.a.shifen.com 的另一个 IPv4 地址为

36.155.132.76

5. 测试 curl 命令，访问一个 web 页面

在 powershell 中使用 curl 访问 www.google.com，输出如下

```
Windows PowerShell
PS C:\Users\16013> curl www.google.com

StatusCode      : 200
StatusDescription : OK
Content         : <!doctype html><html itemscope="" itemtype="http://schema.org/WebPage" lang="zh-CN"><head><meta con
Content="text/html; charset=UTF-8" http-equiv="Content-Type"><meta content="/images/branding/googleg/1
x/...
RawContent      : HTTP/1.1 200 OK
                  Connection: close
                  Content-Security-Policy-Report-Only: object-src 'none';base-uri 'self';script-src 'nonce-xPMVfmj10q
                  TSL8mEYqx1KQ' 'strict-dynamic' 'report-sample' 'unsafe-eval' 'uns...
Forms           : {}
Headers         : {[Connection, close], [Content-Security-Policy-Report-Only, object-src 'none';base-uri 'self';scrip
                  t-src 'nonce-xPMVfmj10qTSL8mEYqx1KQ' 'strict-dynamic' 'report-sample' 'unsafe-eval' 'unsafe-inline'
                  https://report-uri https://csp.withgoogle.com/csp/gws/other-hp], [Cache-Control, private, max
                  -age=0], [Content-Type, text/html; charset=UTF-8]}...
Images          : {@{innerHTML=; innerText=; outerHTML=<IMG id=hplogo style="PADDING-BOTTOM: 14px; PADDING-TOP: 28px;
                  PADDING-LEFT: 0px; PADDING-RIGHT: 0px" alt=Google src="/images/branding/googlelogo/1x/googlelogo_w
                  hite_background_color_272x92dp.png" width=272 height=92>; outerText=; tagName=IMG; id=hplogo; style
                  =PADDING-BOTTOM: 14px; PADDING-TOP: 28px; PADDING-LEFT: 0px; PADDING-RIGHT: 0px; alt=Google; src=/i
                  mages/branding/googlelogo/1x/googlelogo_white_background_color_272x92dp.png; width=272; height=92}}
InputFields     : {@{innerHTML=; innerText=; outerHTML=<INPUT type=hidden value=zh-CN name=hl>; outerText=; tagName=I
                  NPUT; type=hidden; value=zh-CN; name=hl}, @{{innerHTML=; innerText=; outerHTML=<INPUT type=hidden va
                  lue=hp name=source>; outerText=; tagName=INPUT; type=hidden; value=hp; name=source}, @{{innerHTML=;
                  innerText=; outerHTML=<INPUT type=hidden name=biw>; outerText=; tagName=INPUT; type=hidden; name=bi
                  w}, @{{innerHTML=; innerText=; outerHTML=<INPUT type=hidden name=bih>; outerText=; tagName=INPUT; ty
                  pe=hidden; name=bih}}...}
Links           : {@{innerHTML=<SPAN class=gbtb2></SPAN><SPAN class=gbts>搜索</SPAN>; innerText=搜索; outerHTML=<A id
                  =gb_1 class="gbzt gbz0l gbp1" href="https://www.google.com.hk/webhp?tab=ww"><SPAN class=gbtb2></SPA
                  N><SPAN class=gbts>搜索</SPAN></A>; outerText=搜索; tagName=A; id=gb_1; class=gbzt gbz0l gbp1; href
                  =https://www.google.com.hk/webhp?tab=ww}, @{{innerHTML=<SPAN class=gbtb2></SPAN><SPAN class=gbts>图
                  片</SPAN>; innerText=图片; outerHTML=<A id=gb_2 class=gbzt href="https://www.google.com.hk/imghp?hl
                  =zh-CN&tab=wi"><SPAN class=gbtb2></SPAN><SPAN class=gbts>图片</SPAN></A>; outerText=图片; tagNa
                  me=A; id=gb_2; class=gbzt; href=https://www.google.com.hk/imghp?hl=zh-CN&tab=wi}, @{{innerHTML=<
                  SPAN class=gbtb2></SPAN><SPAN class=gbts>地图</SPAN>; innerText=地图; outerHTML=<A id=gb_8 class=gb
                  zt href="http://ditu.google.cn/maps?hl=zh-CN&tab=w1"><SPAN class=gbtb2></SPAN><SPAN class=gbts>
                  地图</SPAN></A>; outerText=地图; tagName=A; id=gb_8; class=gbzt; href=http://ditu.google.cn/maps?hl
                  =zh-CN&tab=w1}, @{{innerHTML=<SPAN class=gbtb2></SPAN><SPAN class=gbts>Play</SPAN>; innerText=Pl
                  ay; outerHTML=<A id=gb_78 class=gbzt href="https://play.google.com/?hl=zh-CN&tab=w8"><SPAN clas
                  s=gbtb2></SPAN><SPAN class=gbts>Play</SPAN></A>; outerText=Play; tagName=A; id=gb_78; class=gbzt; h
```

6. 利用 telnet 命令测试 get 命令，访问 www.baidu.com

telnet 命令使用 GET 访问 www.baidu.com 步骤如下：

- 在‘启用或关闭 Windows 功能’中开启 Telnet 客户端
- 打开终端/cmd 输入 telnet www.baidu.com 80
- 输入 GET / HTTP/1.1 并按下回车
- 输入 Host: www.baidu.com 并按两次回车
- 屏幕上显示出 GET 返回的内容

```
s="mnava s-top-more-btn"><a href="//www.baidu.com/more/" name="tj_briicon" class="s-bri c-font-normal c-color-t" target="_blank">链接</a></div></div><div id="u1" class="s-top-right s-isindex-wrap"><a class="s-top-login-btn c-btn c-btn-primary c-btn-mini lb" style="position:relative;overflow:visible" name="tj_login" href="//www.baidu.com/bdorz/login.gif?login&tpl=mn&u=http%3A%2F%2Fwww.baidu.com%2F%3Fbdorz_come%3D1">百度</a></div><div id="head_wrapper" class="head_wrapper s-isindex-wrap s-ps-islite"><div class="s_form"><div class="s_form_wrapper"><div id="lg" class="s-p-top"><map name="mp"><area style="outline:0" hrefocus="true" shape="rect" coords="0,0,270,129" href="//www.baidu.com/s?wd=%E7%99%B%E5%BA%A%E7%83%AD%E6%90%9C&sa=ire_dl_gh_logo_texting&rsv_dl=igh_logo_pcs" target="_blank" title="徽魂徽消€消魂紅洪响8徽村"></map></div><a href="//www.baidu.com/" id="result_logo"></a><form id="form" name="f" action="//www.baidu.com/s" class="fm"><input type="hidden" name="ie" value="utf-8"> <input type="hidden" name="f" value="8"> <input type="hidden" name="rsv_bp" value="1"> <input type="hidden" name="rsv_idx" value="1"> <input type="hidden" name="ch" value=""> <input type="hidden" name="tn" value="baidu"> <input type="hidden" name="bar" value=""> <span class="s_ipt_wr quickdelete-wrap"><input id="kw" name="wd" class="s_ipt" value="" maxlength="255" autocomplete="off"> </span><span class="s_btn_wr"><input type="submit" id="su" value="徽魂害消€消? class="bg_s_btn"> </span><input type="hidden" name="rn" value=""> <input type="hidden" name="fenlei" value="256"> <input type="hidden" name="oq" value=""> <input type="hidden" name="rsv_pq" value="b9ff093e000e419"> <input type="hidden" name="rsv_t" value="3635FYbdbC8tLWmudZmYaUnaucNe+RzTzNEGqg/JuniQU10WL5mtMQehIrU"> <input type="hidden" name="rqlang" value="cn"> <input type="hidden" name="rsv_enter" value="1"> <input type="hidden" name="rsv_dl" value="ib"></form></div></div><div id="bottom_layer" class="s-bottom-layer s-isindex-wrap"><div class="s-bottom-layer-content"><p class="lh"><a class="text-color" href="//ir.baidu.com/" target="_blank">About Baidu</a></p><p class="lh"><a class="text-color" href="//www.baidu.com/duty" target="_blank">百度义务</a></p><p class="lh"><a class="text-color" href="//help.baidu.com/" target="_blank">帮助</a></p><p class="lh"><a class="text-color" href="//www.beian.gov.cn/portal/registerSystemInfo?recordcode=11000002000001" target="_blank">京公网安备 11000002000001号</a></p><p class="lh"><a class="text-color" href="//beian.miit.gov.cn/" target="_blank">沪公网安备 3101070002000011号</a></p><p class="lh"><a class="text-color" href="//www.beian.miit.gov.cn/" target="_blank">沪公网安备 3101070002000011号</a></p><p class="lh"><a class="text-color" href="//www.baidulicence.com/licence/" target="_blank">百度移动应用</a></p></div></div></div></div><script type="text/javascript">var date=new Date,year=date.getFullYear();document.getElementById("year").innerText="漏"+year+" Baidu"</script></body></html>
```

7. 测试 tracert 命令，并解析其过程

以 www.baidu.com 为例测试 tracert 命令，返回如图所示

```
PS C:\Users\16013> tracert www.baidu.com

通过最多 30 个跃点跟踪
到 www.a.shifen.com [153.3.238.102] 的路由:

  1      2 ms      3 ms      1 ms      10.208.64.1
  2      4 ms      2 ms      2 ms      10.80.128.141
  3      3 ms      10 ms     2 ms      10.80.128.149
  4      1 ms      2 ms      1 ms      10.80.3.10
  5      9 ms      3 ms      2 ms      153.3.60.1
  6      9 ms      6 ms      5 ms      221.6.2.173
  7      5 ms      3 ms      6 ms      122.96.66.97
  8      4 ms      7 ms      3 ms      153.3.228.130
  9      4 ms      5 ms      4 ms      153.37.96.250
 10      *          *          *          请求超时。
 11      *          *          *          请求超时。
 12      *          *          *          请求超时。
 13      4 ms      5 ms      14 ms     153.3.238.102
```

每一行输出包含跳数+响应时间+节点的 IP 地址或域名

跳数：显示从源主机到目标主机的每一跳（或节点）。每一跳通常代表一个路由器或交换机。

响应时间：通常会发送三个小的数据包到每个节点，显示每个数据包的往返时间（RTT）。可以代表到达每个节点的延迟。

节点的 IP 地址或域名：每一跳的节点可能会显示其 IP 地址或域名。有时，如果无法解析节点的名称，只显示 IP 地址。

图中所示第 1 跳为本地无线网关

```
无线局域网适配器 WLAN:

    连接特定的 DNS 后缀 . . . . . :
    IPv4 地址 . . . . . : 10.208.122.226
    子网掩码 . . . . . : 255.255.192.0
    默认网关 . . . . . : 10.208.64.1
```

2~4 跳为校园网内的交换机或路由器，为内网 IP

第 5 跳是校园网到公网的上行端口，该 IP 位于江苏南京

经过 6~12 跳等中转节点后到达百度的服务器 153.3.238.102

其中出现请求超时可能是因为这些路由器被配置为不响应 ICMP Echo Request 消息。

8. 使用 nslookup 查询域名信息，简要分析

以 www.baidu.com 为例测试 nslookup 命令，输出如图所示。

```
Windows PowerShell
PS C:\Users\16013> nslookup www.baidu.com
服务器: UnKnown
Address: 172.19.2.2

非权威应答:
名称: www.a.shifen.com
Addresses: 2409:8c20:6:1135:0:ff:b027:210c
           2409:8c20:6:1d55:0:ff:b09c:7d77
           36.155.132.3
           36.155.132.76
Aliases: www.baidu.com

PS C:\Users\16013> nslookup -qt=RP www.baidu.com
服务器: UnKnown
Address: 172.19.2.2

非权威应答:
www.baidu.com canonical name = www.a.shifen.com

a.shifen.com
primary name server = ns1.a.shifen.com
responsible mail addr = baidu_dns_master.baidu.com
serial = 2405060023
refresh = 5 (5 secs)
retry = 5 (5 secs)
expire = 2592000 (30 days)
default TTL = 3600 (1 hour)
PS C:\Users\16013> |
```

从输出可以得知 www.baidu.com 是 www.a.shifen.com 的别名，图中还显示了 www.baidu.com 的两个 IPv6 地址和两个 IPv4 地址。数据来源不是权威 DNS 服务器，而是来自 DNS 服务器的缓存或其他来源。

primary name server = ns1.a.shifen.com: 指出该域名的主要 DNS 服务器

responsible mail addr = baidu_dns_master.baidu.com: 负责管理该域名的邮箱地址

serial, refresh, retry, expire, default TTL: 这些是关于域名的时间参数，用于控制 DNS 记录的刷新和过期等行为。serial 是序列号（版本号），每当区域文件有更改时，这个数字就会增加（2405060023 表示

2024.05.06 第 23 版)；refresh 是刷新时间，表示从属服务器多久检查一次更新；retry 是重试时间，表示从属服务器尝试联系主服务器失败，它应该多久后重试；expire 是过期时间，表示从属服务器多久没有联系到主服务器后，应该停止回答关于这个区域的查询；default TTL 是默认生存时间，指定 DNS 记录的默认生存时间，即记录在缓存中保持有效的时间长度。如果没有特定指定的 TTL，将会使用这个默认值。

(二) 传输层协议分析

1. TCP 数据流的追踪

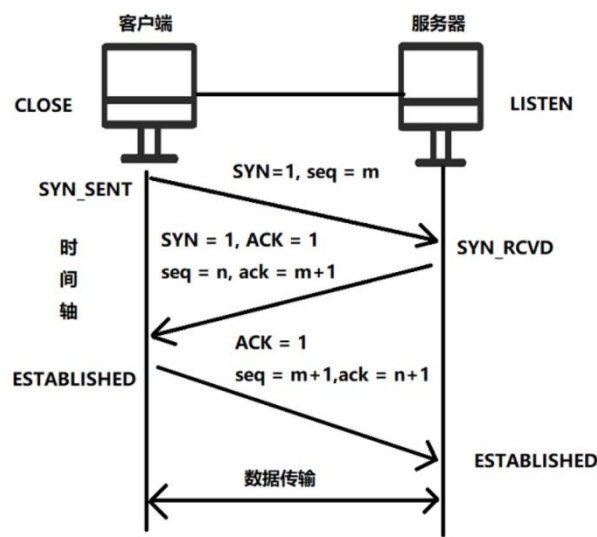
打开 Wireshark，在终端中输入 curl www.baibu.com，让 Wireshark 的抓包

在 Wireshark 中设置显示过滤器为 http，过滤出上一步操作对应的 http 流，右键追踪流选择 TCP 流即可

tcp.stream eq 31						
No.	Time	Source	Destination	Protocol	Length	Info
334	41.112808	10.208.122.226	153.3.238.110	TCP	66	10188 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
335	41.120623	153.3.238.110	10.208.122.226	TCP	66	80 → 10188 [SYN, ACK] Seq=0 Ack=1 Win=8192 Len=0 MSS=1452 WS=32 SACK_PERM
336	41.120725	10.208.122.226	153.3.238.110	TCP	54	10188 → 80 [ACK] Seq=1 Ack=1 Win=132096 Len=0
337	41.120818	10.208.122.226	153.3.238.110	HTTP	130	GET / HTTP/1.1
338	41.124668	153.3.238.110	10.208.122.226	TCP	60	80 → 10188 [ACK] Seq=1 Ack=77 Win=78464 Len=0
339	41.167975	153.3.238.110	10.208.122.226	TCP	1494	80 → 10188 [PSH, ACK] Seq=1 Ack=77 Win=78464 Len=1440 [TCP segment of a reassembled PDU]
340	41.167975	153.3.238.110	10.208.122.226	HTTP	1395	HTTP/1.1 200 OK (text/html)
341	41.168055	10.208.122.226	153.3.238.110	TCP	54	10188 → 80 [ACK] Seq=77 Ack=2782 Win=132096 Len=0
342	41.168307	10.208.122.226	153.3.238.110	TCP	54	10188 → 80 [FIN, ACK] Seq=77 Ack=2782 Win=132096 Len=0
343	41.171797	153.3.238.110	10.208.122.226	TCP	60	80 → 10188 [ACK] Seq=2782 Ack=78 Win=78464 Len=0
344	41.171797	153.3.238.110	10.208.122.226	TCP	60	80 → 10188 [FIN, ACK] Seq=2782 Ack=78 Win=78464 Len=0
345	41.171854	10.208.122.226	153.3.238.110	TCP	54	10188 → 80 [ACK] Seq=78 Ack=2783 Win=132096 Len=0

可以看到在 http 请求发送前，客户端与服务端进行了三次握手。最后四条记录为 TCP 的四次挥手。

三次握手示例图：



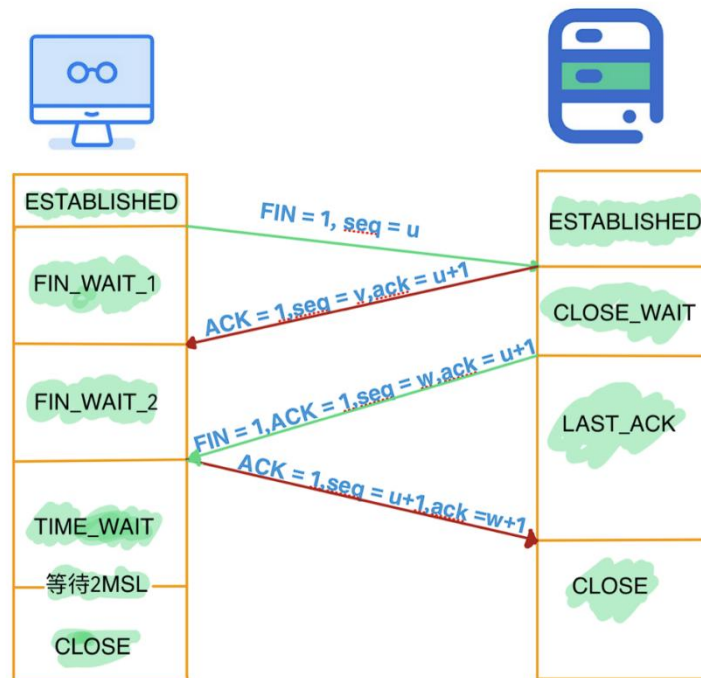
从抓包数据得知第一次握手客户端向服务端发送了标志位 SYN 为 1, Seq=0 的 TCP 报文。

第二次握手服务端向客户端发送了标志位 SYN、ACK 为 1, Ack=0+1=1, Seq=0 的 TCP 报文。

第三次握手客户端向服务端发送了标志位 ACK 为 1, Seq=0+1=1, Ack=0+1=1 的 TCP 报文。

在三次握手阶段, 发送端会发送 Seq=j 的报文, 接收端返回 Ack=j+1 的报文, 发送端接收到接收端返回的报文并检测 Ack 是否为 j+1, 表示接收端是否确认连接请求。

四次挥手示例图:



挥手请求可以由客户端或服务端发起, 在本次抓包中由客户端发起。

第一次挥手客户端向服务端发送了标志位 FIN、ACK 为 1, Ack=2782, Seq=77 的 TCP 报文。

第二次挥手服务端向客户端发送了标志位 ACK 为 1, Seq=2782, Ack=77+1=78 的 TCP 报文。

第三次挥手服务端向客户端发送了标志位 FIN、ACK 为 1, Seq=2782, Ack=77+1=78 的 TCP 报文。

第四次挥手客户端向服务端发送了标志位 ACK 为 1, Seq=2782+1=2783, Ack=77+1=78 的 TCP 报文。

四次挥手实际为客户端和服务端双向四次进行分手请求和分手确认。

四、实验总结

通过这次实验我学会了 Wireshark 的基础用法，并借助 Wireshark 分析 HTTP 和 DNS 报文，观察 TCP 连接，包括 TCP 的三次握手和四次挥手。在本次实验中我还学会了使用 windows 自带的一些网络分析指令，通过这些指令我对数据包在交换机间的跳转、DNS 的实际应用有了更深刻的认识。