

# 利用 Wireshark 观察网络 报文

# 1.Wireshark 简介

Wireshark 是一个网络封包分析软件。网络封包分析软件的功能是抓取网络封包，并尽可能显示出最为详细的网络封包资料。

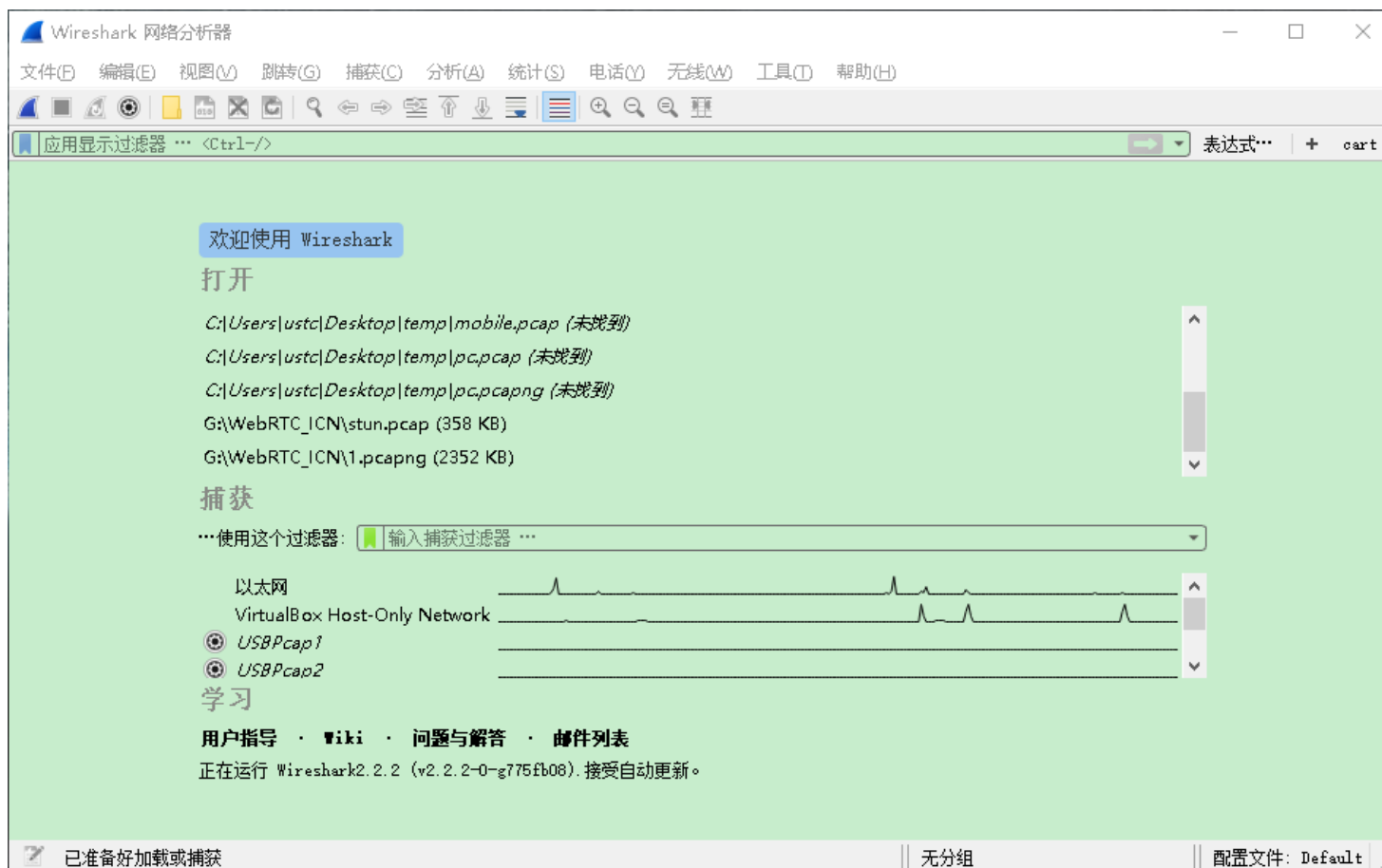
# 1.Wireshark 简介

## 抓包原理

- 网卡接收到一个报文后，会遍历系统中已经注册的 sniffer, 并调用其处理函数。
- wireshark 通过注册一种虚拟的底层网络协议来获得报文副本。

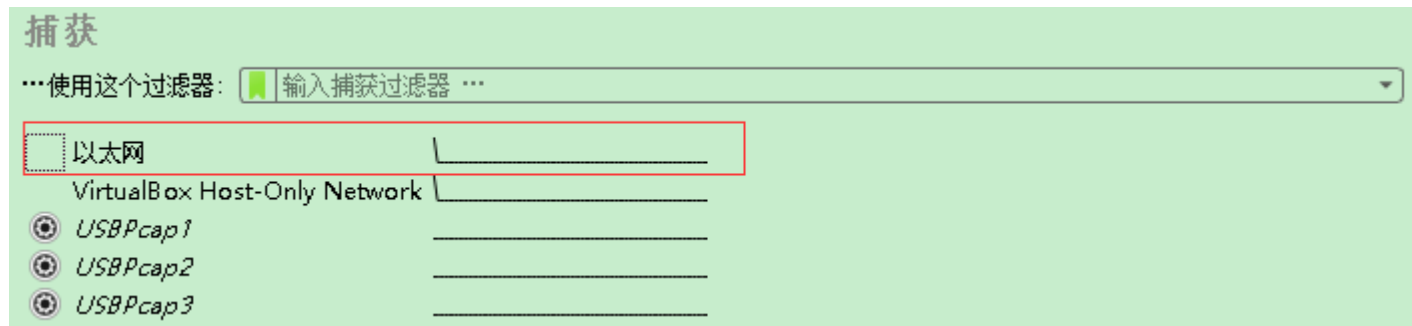
## 2. 打开 Wireshark

可在网上下载最新版本并安装。官网链接：<https://www.wireshark.org/download.html>



### 3. 选择捕获网卡

- wireshark 是捕获机器上的某一块网卡的网络包，当你的机器上有多块网卡的时候，你需要选择一个网卡。
- 在主界面上“捕获”一栏中即可选择，比如选择以太网。

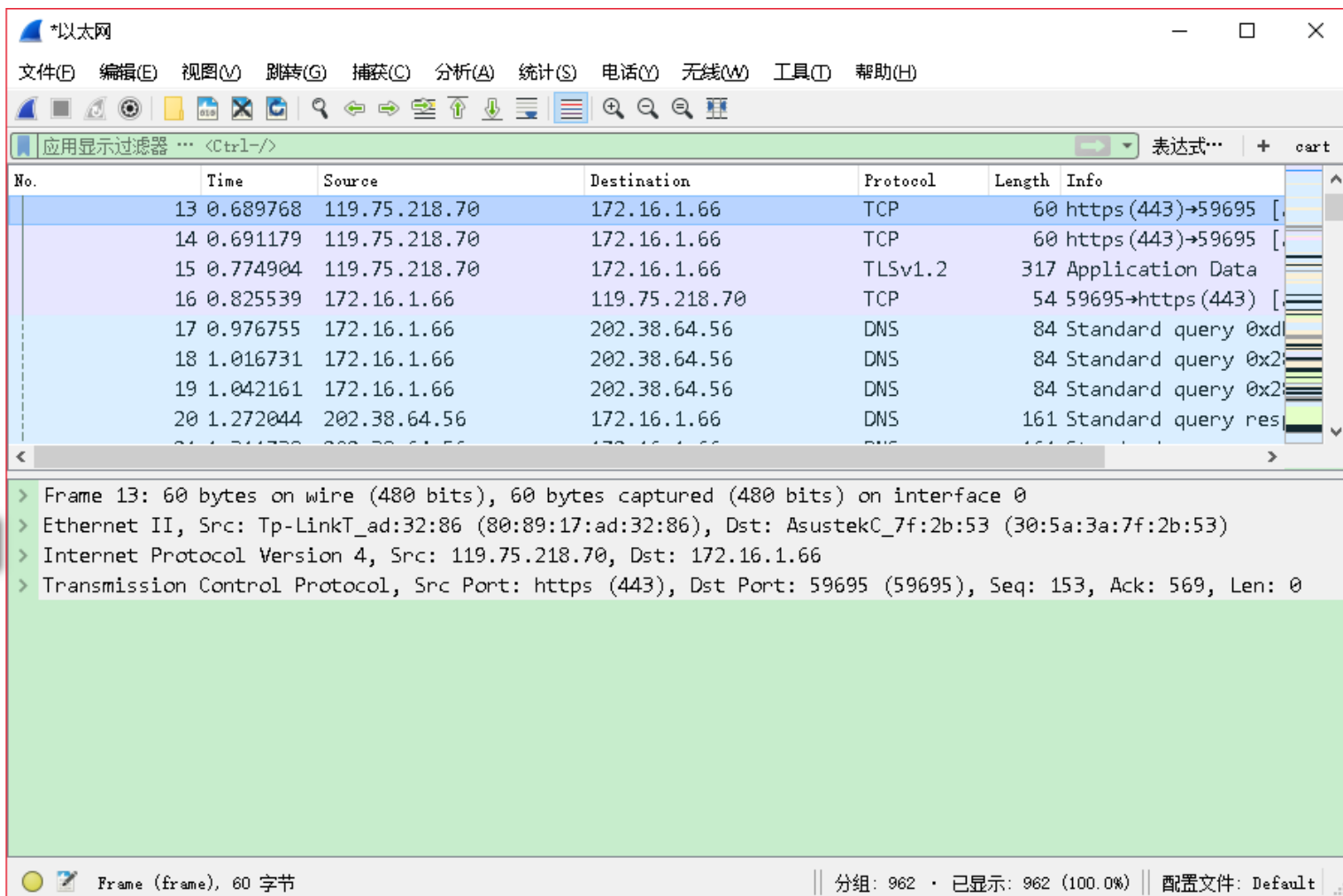


# 4. Wireshark 窗口介绍

显示过滤器

封包列表

封包详细信息



# 5. Wireshark 与对应的 TCP/IP 五层模型



TCP/IP 5层模型

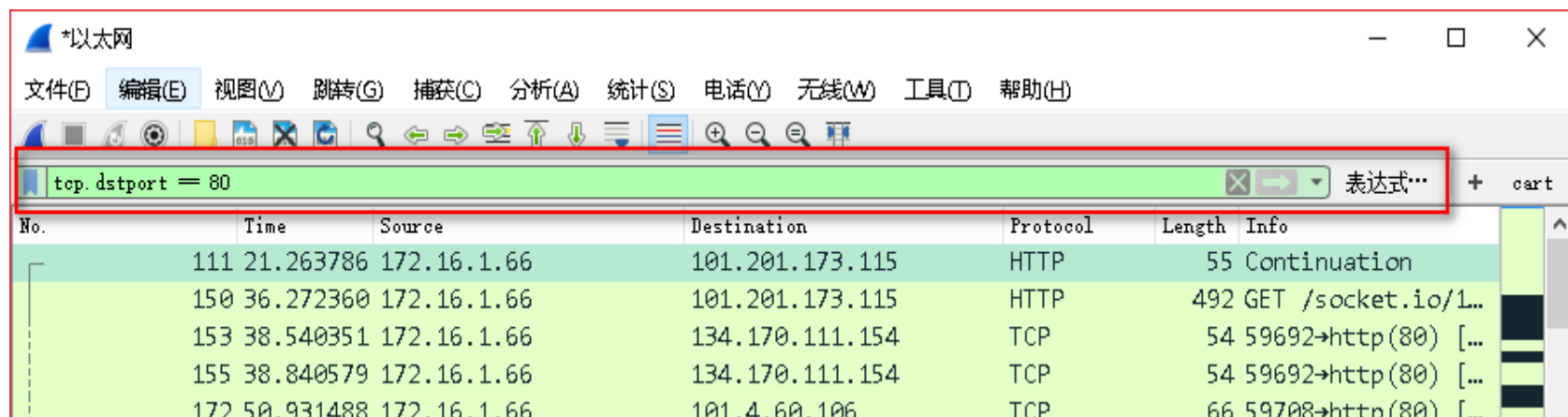
Wireshark interface showing a packet capture on the Ethernet interface. The filter is `tcp.dstport == 80`. The packet list shows several packets, including an HTTP continuation and a GET request. The packet details pane for packet 111 shows the following layers:

- Frame (frame), 55 字节
- Ethernet II, Src: AsustekC\_7f:2b:53 (30:5a:3a:7f:2b:53), Dst: Tp-LinkT\_ad:32:86 (80:89:17:ad:32:86)
- Internet Protocol Version 4, Src: 172.16.1.66, Dst: 101.201.173.115
- Transmission Control Protocol, Src Port: 55649 (55649), Dst Port: http (80), Seq: 1, Ack: 1, Len: 1
- Hypertext Transfer Protocol

The status bar at the bottom indicates: 分组: 962 · 已显示: 34 (3.5%) · 已丢弃: 0 (0.0%) | 配置文件: Default

# 6. 使用显示过滤器

- 有时候，我们会捕获到大量的冗余信息，有几千甚至几万条记录，以至于很难找到自己需要的部分。
- 所以就需要使用显示过滤器，来帮助我们过滤出我们需要的信息。
- 比如下图使用 `tcp.dstport == 80` 这个表达式来过滤出 TCP 目的端口是 80 的数据包。





# 6. 使用显示过滤器

比较操作符：

== eq

!= ne

> gt

< lt

>= ge

contains

~ matches

& bitwise\_and

逻辑运算符

&& and

|| or

^^ xor

! not

注意 ip.addr != 172.16.1.111 的意义：

存在一个地址（源 ip 地址或目的 ip 地址）不等于 172.16.1.111.

# 6. 使用显示过滤器

一些常用的过滤规则如下：

- 1. 协议过滤

- 比如 tcp ，只显示应用 TCP 协议的数据包。

- 2.IP 过滤

- 比如 ip.src == 192.168.1.102 ，显示源 IP 地址为 192.168.1.102 的数据包
- ip.dst == 192.168.1.102, 显示目标 IP 地址为 192.168.1.102 的数据包

# 6. 使用显示过滤器

## • 3. 端口过滤

- `tcp.port == 80`, 显示 TCP 协议端口为 80 的包
- `tcp.srcport == 80`, 只显示 TCP 协议的源端口为 80 的数据包

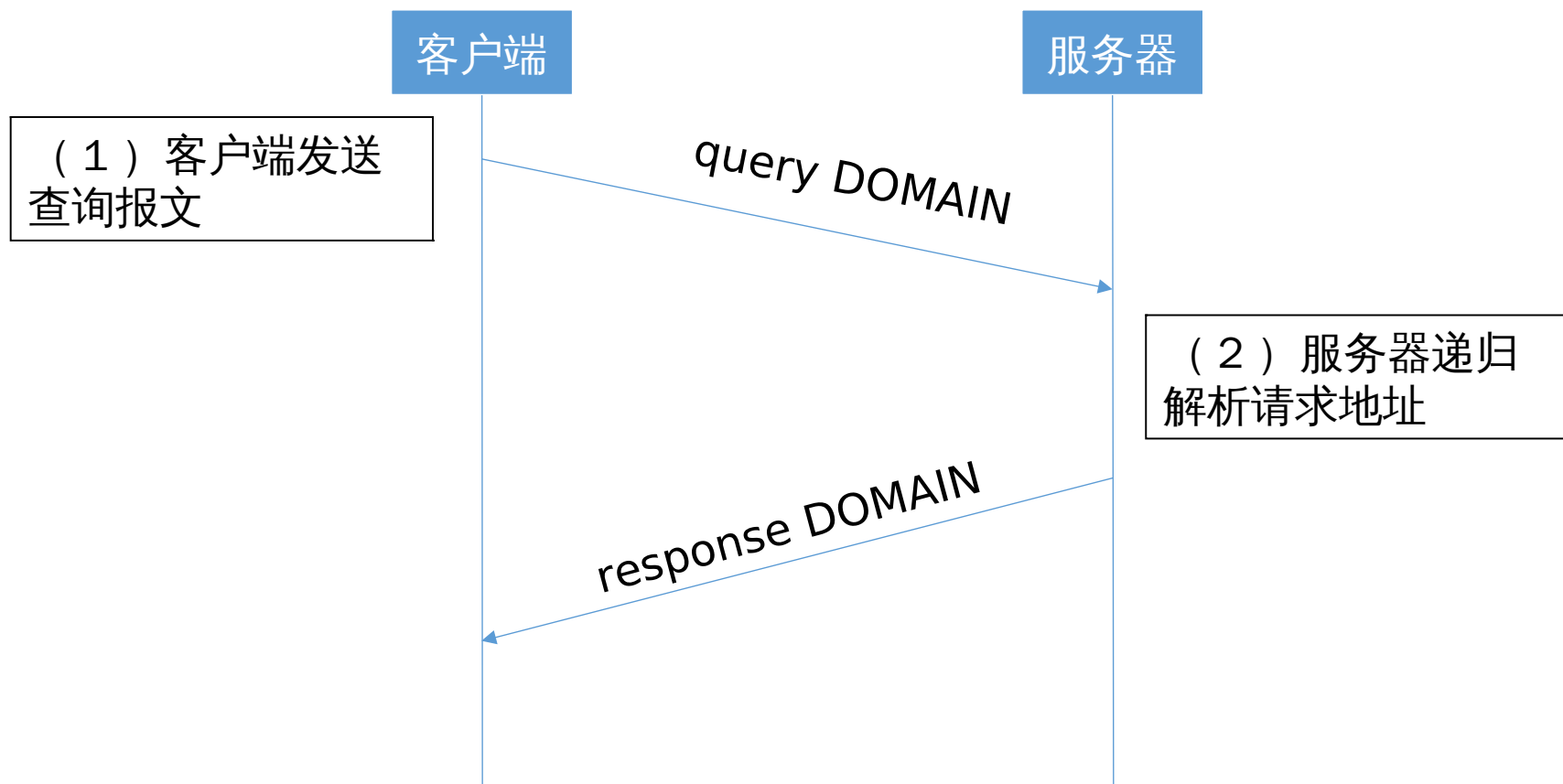
## • 4. HTTP 模式过滤

- `http.request.method == "GET"`, 显示 HTTP GET 方法的包。

## • 5. 逻辑运算符

- `ip.src == 192.168.1.102 or ip.dst == 192.168.1.102`, 查看源 IP 地址或者目标 IP 地址是 192.168.1.102 的数据包
- `not` 优先级最高, `and` 和 `or` 优先级相同. 可以使用括号.

# 7. 举例：观察 DNS 解析过程



# 7. 举例：观察 DNS 解析过程

- 使用 nslookup 命令进行 DNS 查询或者直接在浏览器访问。

```
# oven @ oven in ~ [12:24:16]
$ nslookup github.com
Server:      172.16.1.1
Address:     172.16.1.1#53

Non-authoritative answer:
Name:   github.com
Address: 192.30.253.113
Name:   github.com
Address: 192.30.253.112

# oven @ oven in ~ [12:24:24]
$ █
```

# 7. 举例：观察 DNS 解析过程

Capturing from enp4s0

No.	Time	Source	Destination	Protocol	Length	Info
101	7.128930084	172.16.1.111	172.16.1.1	DNS	83	Standard query 0x4c82 A collector.githubapp.com
102	7.128941494	172.16.1.111	172.16.1.1	DNS	83	Standard query 0xe696 AAAA collector.githubapp.com
103	7.129584980	172.16.1.111	172.16.1.1	DNS	70	Standard query 0x88ba A github.com
104	7.129590238	172.16.1.111	172.16.1.1	DNS	70	Standard query 0x48c1 AAAA github.com
105	7.130079997	172.16.1.111	172.16.1.1	DNS	81	Standard query 0x1666 A assets-cdn.github.com
106	7.130087285	172.16.1.111	172.16.1.1	DNS	81	Standard query 0x9070 AAAA assets-cdn.github.com
107	7.133238123	172.16.1.1	172.16.1.111	DNS	154	Standard query response 0x48c1 AAAA github.com SOA ns-1707.awsdns-21.co.uk
108	7.133478827	172.16.1.1	172.16.1.111	DNS	174	Standard query response 0x9070 AAAA assets-cdn.github.com CNAME github.map.fastly.net SOA ns1.fastly.net
112	7.313035716	172.16.1.1	172.16.1.111	DNS	102	Standard query response 0x88ba A github.com A 192.30.253.112 A 192.30.253.113
114	7.313945893	172.16.1.111	172.16.1.1	DNS	90	Standard query 0x00f6 A avatars0.githubusercontent.com
115	7.313951516	172.16.1.111	172.16.1.1	DNS	90	Standard query 0xbefe AAAA avatars0.githubusercontent.com CNAME github.map.fastly.net SOA ns1.fastly.net
117	7.315638279	172.16.1.1	172.16.1.111	DNS	143	Standard query response 0xbefe AAAA avatars0.githubusercontent.com CNAME github.map.fastly.net SOA ns1.fastly.net
118	7.329839721	172.16.1.1	172.16.1.111	DNS	141	Standard query response 0x00f6 A avatars0.githubusercontent.com CNAME github.map.fastly.net A 151.101.76.133
119	7.329898791	172.16.1.1	172.16.1.111	DNS	132	Standard query response 0x1666 A assets-cdn.github.com CNAME github.map.fastly.net A 151.101.76.133
120	7.330677598	172.16.1.111	172.16.1.1	DNS	90	Standard query 0xf61a A avatars3.githubusercontent.com
121	7.330681363	172.16.1.111	172.16.1.1	DNS	90	Standard query 0x6ec6 A avatars1.githubusercontent.com
122	7.330682018	172.16.1.111	172.16.1.1	DNS	90	Standard query 0x98d0 AAAA avatars1.githubusercontent.com
123	7.330684283	172.16.1.111	172.16.1.1	DNS	90	Standard query 0x2621 AAAA avatars3.githubusercontent.com
129	7.480180298	172.16.1.1	172.16.1.111	DNS	183	Standard query response 0x98d0 AAAA avatars1.githubusercontent.com CNAME github.map.fastly.net SOA ns1.fastly.net
130	7.481644810	172.16.1.1	172.16.1.111	DNS	141	Standard query response 0x6ec6 A avatars1.githubusercontent.com CNAME github.map.fastly.net A 151.101.76.133
131	7.482027661	172.16.1.1	172.16.1.111	DNS	183	Standard query response 0x2621 AAAA avatars3.githubusercontent.com CNAME github.map.fastly.net SOA ns1.fastly.net
132	7.482301417	172.16.1.1	172.16.1.111	DNS	141	Standard query response 0xf61a A avatars3.githubusercontent.com CNAME github.map.fastly.net A 151.101.76.133
133	7.484101615	172.16.1.111	172.16.1.1	DNS	90	Standard query 0xcafc A avatars2.githubusercontent.com
134	7.484123195	172.16.1.111	172.16.1.1	DNS	90	Standard query 0xb41d AAAA avatars2.githubusercontent.com
135	7.484970764	172.16.1.111	172.16.1.1	DNS	86	Standard query 0x6826 A camo.githubusercontent.com
136	7.484989503	172.16.1.111	172.16.1.1	DNS	86	Standard query 0x843d AAAA camo.githubusercontent.com
137	7.486337488	172.16.1.1	172.16.1.111	DNS	137	Standard query response 0x6826 A camo.githubusercontent.com CNAME github.map.fastly.net A 151.101.76.133
138	7.486562482	172.16.1.1	172.16.1.111	DNS	179	Standard query response 0x843d AAAA camo.githubusercontent.com CNAME github.map.fastly.net SOA ns1.fastly.net
154	7.663035070	172.16.1.1	172.16.1.111	DNS	198	Standard query response 0x4c82 A collector.githubapp.com CNAME analytics-collector-28944298.us-east-1.elb.amazonaws.com A 54.209.228.2 A 52.21.179.218 A 34.201.185.117
155	7.663217404	172.16.1.1	172.16.1.111	DNS	232	Standard query response 0xe696 AAAA collector.githubapp.com CNAME analytics-collector-28944298.us-east-1.elb.amazonaws.com SOA ns-1119.awsdns-11.org
168	7.713549739	172.16.1.1	172.16.1.111	DNS	141	Standard query response 0xcafc A avatars2.githubusercontent.com CNAME github.map.fastly.net A 151.101.76.133
169	7.713797433	172.16.1.1	172.16.1.111	DNS	183	Standard query response 0xb41d AAAA avatars2.githubusercontent.com CNAME github.map.fastly.net SOA ns1.fastly.net
267	8.810690826	172.16.1.111	172.16.1.1	DNS	89	Standard query 0xa6e2 A github-cloud.s3.amazonaws.com
268	8.810701799	172.16.1.111	172.16.1.1	DNS	89	Standard query 0xa8f3 AAAA github-cloud.s3.amazonaws.com
269	8.810708557	172.16.1.111	172.16.1.1	DNS	93	Standard query 0x920e A user-images.githubusercontent.com
270	8.810715904	172.16.1.111	172.16.1.1	DNS	93	Standard query 0x4218 AAAA user-images.githubusercontent.com
285	8.994368420	172.16.1.1	172.16.1.111	DNS	144	Standard query response 0x920e A user-images.githubusercontent.com CNAME github.map.fastly.net A 151.101.76.133
288	9.035275934	172.16.1.1	172.16.1.111	DNS	191	Standard query response 0xa8f3 AAAA github-cloud.s3.amazonaws.com CNAME s3-1-w.amazonaws.com SOA ns-978.awsdns-58.net
298	9.144716035	172.16.1.111	172.16.1.1	DNS	75	Standard query 0x4e45 A live.github.com
299	9.144725029	172.16.1.111	172.16.1.1	DNS	75	Standard query 0x5e57 AAAA live.github.com
303	9.191802044	172.16.1.1	172.16.1.111	DNS	186	Standard query response 0x4218 AAAA user-images.githubusercontent.com CNAME github.map.fastly.net SOA ns1.fastly.net
324	9.327435929	172.16.1.1	172.16.1.111	DNS	107	Standard query response 0x4e45 A live.github.com A 192.30.253.125 A 192.30.253.124
325	9.327688918	172.16.1.1	172.16.1.111	DNS	140	Standard query response 0x5e57 AAAA live.github.com SOA ns1.p16.dynect.net
336	9.364950183	172.16.1.1	172.16.1.111	DNS	126	Standard query response 0xa6e2 A github-cloud.s3.amazonaws.com CNAME s3-1-w.amazonaws.com A 52.216.160.243
368	9.722786781	172.16.1.111	172.16.1.1	DNS	74	Standard query 0x8ad0 A api.github.com
369	9.722804120	172.16.1.1	172.16.1.1	DNS	74	Standard query 0xaea8 AAAA api.github.com
370	9.847165801	172.16.1.1	172.16.1.111	DNS	106	Standard query response 0x8ad0 A api.github.com A 192.30.253.117 A 192.30.253.116
371	9.847429332	172.16.1.1	172.16.1.111	DNS	139	Standard query response 0xaea8 AAAA api.github.com SOA ns1.p16.dynect.net

Frame 101: 83 bytes on wire (664 bits), 83 bytes captured (664 bits) on interface 0  
Ethernet II, Src: Micro-St\_c0:e2:80 (4c:cc:6a:c0:e2:80), Dst: AsustekK\_Cc:e6:a0 (b0:6e:bf:c6:e6:a0)  
Internet Protocol Version 4, Src: 172.16.1.111, Dst: 172.16.1.1  
User Datagram Protocol, Src Port: 51326, Dst Port: 53  
Domain Name System (query)

0000 b0 6e bf c6 e6 a0 4c cc 6a c0 e2 80 08 00 45 05 n-1-L-j-...E-  
0010 80 45 99 b0 40 00 00 11 46 6f ac 10 01 6f ac 10 E-00-0-Fg...o-  
0020 01 01 c8 7e 00 35 00 31 b3 b5 4c 82 01 00 00 01 -5-1-L-...  
0030 00 00 00 00 00 00 63 6f 6c 65 63 74 6f 72 c collector  
0040 09 67 69 74 68 75 62 61 70 70 93 63 6f 6d 00 00 githuba pp.com

Internet Protocol Version 4 (ip), 20 bytes

Packets: 2733 · Displayed: 48 (1.8%)

Profile: Default

# 7. 举例：观察 DNS 解析过程

dns.qry.name == github.com

The image shows a Wireshark packet capture of DNS traffic. The top pane displays a list of packets, with packet 112 selected. The middle pane shows the details of packet 112, which is a DNS response from 172.16.1.1 to 172.16.1.1. The bottom pane shows the raw packet data in hexadecimal and ASCII.

No.	Time	Source	Destination	Protocol	Length	Info
103	7.129584080	172.16.1.111	172.16.1.1	DNS	70	Standard query 0x88ba A github.com
104	7.129598238	172.16.1.111	172.16.1.1	DNS	70	Standard query 0x48c1 AAAA github.com
107	7.133238123	172.16.1.1	172.16.1.111	DNS	154	Standard query response 0x48c1 AAAA github.com SOA ns-1707.awsdns-21.co.uk
112	7.31935716	172.16.1.1	172.16.1.111	DNS	102	Standard query response 0x88ba A github.com A 192.30.253.112 A 192.30.253.113

Frame 112: 102 bytes on wire (816 bits), 102 bytes captured (816 bits) on interface 0  
Ethernet II, Src: AsustekC\_6c:e6:a0 (b0:6e:bf:6c:e6:a0), Dst: Micro-St\_c0:e2:80 (4c:cc:6a:c0:e2:80)  
Internet Protocol Version 4, Src: 172.16.1.1, Dst: 172.16.1.111  
User Datagram Protocol, Src Port: 53, Dst Port: 56190  
Domain Name System (response)  
Transaction ID: 0x88ba  
Flags: 0x8180 Standard query response, No error  
1... .. = Response: Message is a response  
0000... .. = 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 the server  
... ..0... .. = Non-authenticated data: Unacceptable  
... ..0000 = Reply code: No error (0)  
Questions: 1  
Answer RRs: 2  
Authority RRs: 0  
Additional RRs: 0  
Queries  
github.com: type A, class IN  
Name: github.com  
[Name Length: 10]  
[Label Count: 2]  
Type: A (Host Address) (1)  
Class: IN (0x0001)  
Answers  
github.com: type A, class IN, addr 192.30.253.112  
Name: github.com  
Type: A (Host Address) (1)  
Class: IN (0x0001)  
Time to live: 60  
Data length: 4  
Address: 192.30.253.112  
github.com: type A, class IN, addr 192.30.253.113  
Name: github.com  
Type: A (Host Address) (1)  
Class: IN (0x0001)  
Time to live: 60  
Data length: 4  
Address: 192.30.253.113  
[Request In: 103]  
[Time: 0.183451636 seconds]

# 8. 实验内容

利用 wireshark 观察在浏览器访问 <https://www.github.com> 时的交互过程。

## 实验要求

- (1) 保存抓包结果为文件，文件名为“学号 + 姓名 + wireshark+1+cap.pcap”。  
(5%)
- (2) 分析对该网址的 DNS 解析过程。
  - 使用显示过滤器，使之只显示该过程中的报文。给出你定义的显示过滤器和过滤后的截图（包含整个 wireshark 窗口）。（10%）
  - 对显示的每个数据包给出解释，格式为“ No: 解释”。解释为：查询“域名”或回复“ No” 的查询，回复内容为“ ip 列表”。（No 为 wireshak 中数据包的“ No.” 列的值）（10%）
  - 交互过程中可能出现多次 DNS query，给出其原因。（30%）



# 8. 实验内容

## (3) 分析 https 握手过程 .

- 使用显示过滤器，使之只显示与 `https://www.github.com` 交互的 https 报文 . 给出你定义的显示过滤器和过滤后的截图 ( 包含整个 wireshark 窗口 ). (15%)
- 参照第 12 页的交互图，画出握手过程的交互图 . 给出每次交互对应的数据包的 No. (30%)

## (4) 注意

- 访问过程中，页面中有的元素需要从其他位置获取，因此需要访问其他域名，如从 `https://avatars0.githubusercontent.com` 获取用户头像 . 这种情况不需要考虑 .
- 截图时，如果报文过多，只需要显示最开始的若干报文 .
- 步骤 (2), (3) 的结果以一个 word 文档的形式提交，文件名为“学号 + 姓名 + wireshark+1+report.docx” .
- wireshark 抓包实验包含多个部分，本次为第一个部分 . (1)(2)(3) 中的百分比为该项占此部分实验分值的百分比 .
- 将抓包文件和 word 文档放在一个目录下，目录名为“学号 + 姓名 + 实验 4.1”，将此目录压缩为 zip 再提交 . 请在 2018 年 11 月 11 日之前 ( 不包含 11 月 11 日 ) 提交 .