

网络协议分析

(一) DHCP 协议

DHCP 的主要功能是为网络中的客户端自动分配 IP 地址、子网掩码等 TCP/IP 配置参数。通过 DHCP 客户端能够在无需手动配置的情况下自动获取这些网络参数，从而简化了网络配置管理。

在 Linux 操作系统中的 NAT 模式连接后，使用 `sudo dhclient -r` 释放 IP 地址，该命令会释放当前计算机的 IP 地址，客户端与 DHCP 服务器断开连接。执行命令 `sudo dhclient` 重新获取 IP 地址。

517	38.642103746	0.0.0.0	255.255.255.255	DHCP	342	DHCP Discover	- Transaction ID 0xa6fe8a25
518	38.642425629	10.0.2.2	10.0.2.15	DHCP	590	DHCP Offer	- Transaction ID 0xa6fe8a25
519	38.642591543	0.0.0.0	255.255.255.255	DHCP	342	DHCP Request	- Transaction ID 0xa6fe8a25
520	38.642754986	10.0.2.2	10.0.2.15	DHCP	590	DHCP ACK	- Transaction ID 0xa6fe8a25

DHCP Discover: 当 DHCP 客户端第一次连接到网络时，客户端通过广播方式请求 DHCP 服务器分配一个合适的 IP 地址及其他网络配置信息。由于客户端不知道哪个 DHCP 服务器在网络中提供服务，它必须使用广播来发送请求报文，以确保网络中所有的 DHCP 服务器都能接收到该请求。客户端此时尚未配置 IP 地址，所以其源 IP 地址为 0.0.0.0，表示客户端还没有有效的网络配置。

```

517.38.642102746. 0.0.0.0      255.255.255.255      DHCP      342 DHCP Discover - Transaction ID: 0xa6fe0a25
# User Datagram Protocol, Src Port: 68, Dst Port: 67
# Dynamic Host Configuration Protocol (Discover)
  Message type: Boot Request (1)
  Hardware type: Ethernet (0x01)
  Hardware address length: 6
  Hops: 0
  Transaction ID: 0xa6fe0a25
  Seconds elapsed: 0
# Bootp flags: 0x0000 (Unicast)
  Client IP address: 0.0.0.0
  Your (client) IP address: 0.0.0.0
  Next server IP address: 0.0.0.0
  Relay agent IP address: 0.0.0.0
  Client MAC address: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99)
  Client hardware address padding: 00000000000000000000
  Server host name not given
  Boot file name not given
  Magic cookie: DHCP
# Option: (53) DHCP Message Type (Discover)
# Option: (50) Requested IP Address (10.0.2.15)
# Option: (12) Host Name
# Option: (55) Parameter Request List
  Length: 13
  Parameter Request List Item: (1) Subnet Mask
  Parameter Request List Item: (28) Broadcast Address
  Parameter Request List Item: (2) Time Offset
  Parameter Request List Item: (3) Router
  Parameter Request List Item: (15) Domain Name
  Parameter Request List Item: (6) Domain Name Server
  Parameter Request List Item: (119) Domain Search
  Parameter Request List Item: (12) Host Name
  Parameter Request List Item: (44) NetBIOS over TCP/IP Name Server
  Parameter Request List Item: (47) NetBIOS over TCP/IP Scope
  Parameter Request List Item: (26) Interface MTU
  Parameter Request List Item: (121) Classless Static Route
  Parameter Request List Item: (42) Network Time Protocol Servers
# Option: (255) End
  Padding: 0000000000000000000000000000000000000000

```

可以看到该数据包中携带客户端请求、客户端随机生成的 ID, 用以标志本次 DHCP 通信、请求分配的 IP 地址以及设置请求选项列表 option, 客户端利用该选项指明需要从 DHCP 服务器获取哪些网络配置参数。

DHCP Offer: 在 DHCP 服务器收到 Discover 报文后, 就会在所配置的地址池中查找一个合适的 IP 地址, 加上相应的租约期限和其他配置信息, 建立 Offer 报文, 发给 DHCP 客户端, 告知用户可以为它提供 IP 地址。

从报文中可以看到服务器响应与标志 ID，对比发现和 Discover 报文中的相同。还可以看到服务器所分配的 IP 地址、DHCP 服务器地址、租约期限、DNS 服务器地址、子网掩码和默认网关。

[illegible]

DHCP Request: 在 DHCP 中，客户端在收到多个 DHCP Offer 报文后，需要选择一个最合适的服务器进行 IP 地址的分配。通常是选择第一个报文的服务器作为自己的目标服务器，并向该服务器发送一个广播的 Request 请求报文，通告选择的服务器，希望获得所分配的 IP 地址。

DHCP ACK: 当 DHCP 服务器 接收到 DHCP Request 请求报文后，它会根据报文中包含的用户 MAC 地址 查找是否有对应的租约记录。如果找到了有效的租约记录，服务器便会发送 DHCP ACK 应答报文，告知客户端可以使用所分配的 IP 地址。可以发现中间携带了最终分配的 IP 地址。

（二）ARP 协议

ARP 用于根据 IP 地址获取对应的物理地址。当主机访问 `www.xjtu.edu.cn` 时，首先需要通过 DNS 服务器 进行域名解析。但由于目标服务器的 IP 地址与本机不在同一网段，主机先会通过 ARP 协议获取网关的 MAC 地址，然后由网关转发数据包。

19	5.125492743	PcsCompu_4c:dc:99	RealtekU_12:35:02	ARP	42 Who has 10.0.2.2? Tell 10.0.2.15
20	5.126176114	RealtekU_12:35:02	PcsCompu_4c:dc:99	ARP	60 10.0.2.2 is at 52:54:00:12:35:02

ARP 请求报文：由于主机尚未知道网关的 MAC 地址，它会将包含网关 IP 地址的 ARP 请求报文广播到局域网中的所有主机，等待接收返回的消息，以便获取网关的 MAC 地址。

19	5.125492743	PcsCompu_4c:dc:99	RealtekU_12:35:02	ARP	42 Who has 10.0.2.2? Tell 10.0.2.15
Frame 19: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface enp0s3, id 0					
Ethernet II, Src: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)					
Address Resolution Protocol (request)					
Hardware type: Ethernet (1)					
Protocol type: IPv4 (0x0800)					
Hardware size: 6					
Protocol size: 4					
Opcode: request (1)					
Sender MAC address: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99)					
Sender IP address: 10.0.2.15					
Target MAC address: 00:00:00:00:00:00 (00:00:00:00:00:00)					
Target IP address: 10.0.2.2					

ARP 响应报文：网关接收到 ARP 请求 后，发现请求中包含的 IP 地址与自己的匹配，便将自身的 MAC 地址填入 ARP 响应报文，并将该报文发送给请求主机。最终，主机在收到 ARP 响应后，将请求的 IP 地址 和对应的 MAC 地址 存储到 ARP 表 中，进行动态更新和维护。

20	5.126176114	RealtekU_12:35:02	PcsCompu_4c:dc:99	ARP	60 10.0.2.2 is at 52:54:00:12:35:02
Frame 20: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface enp0s3, id 0					
Ethernet II, Src: RealtekU_12:35:02 (52:54:00:12:35:02), Dst: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99)					
Address Resolution Protocol (reply)					
Hardware type: Ethernet (1)					
Protocol type: IPv4 (0x0800)					
Hardware size: 6					
Protocol size: 4					
Opcode: reply (2)					
Sender MAC address: RealtekU_12:35:02 (52:54:00:12:35:02)					
Sender IP address: 10.0.2.2					
Target MAC address: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99)					
Target IP address: 10.0.2.15					

（三）DNS 协议

DNS 是互联网的一项核心服务。它负责将域名和 IP 地址相互映射，允许终端设备将易于理解的 URL 转换为机器可以识别的 IP 地址，从而方便用户访问互联网。当主机访问 `www.xjtu.edu.cn` 时，需要向 DNS 服务器 查询该域名对应的 IP 地址。

1	0.000000000	10.0.2.15	223.5.5.5	DNS	86 Standard query 0x4dbb A www.xjtu.edu.cn OPT
2	0.000048403	10.0.2.15	223.5.5.5	DNS	86 Standard query 0x783c AAAA www.xjtu.edu.cn OPT
3	0.003822481	223.5.5.5	10.0.2.15	DNS	114 Standard query response 0x783c AAAA www.xjtu.edu.cn AAAA 2001...
4	0.004443660	223.5.5.5	10.0.2.15	DNS	102 Standard query response 0x4dbb A www.xjtu.edu.cn A 202.117.1...

DNS 查询报文可以了解到数据包中存在的域名服务器地址，与 DHCP 服务器分配的地址相同。其中还携带查询的域名，type A 表示记录 DNS 域名对应的 IP 地址，class IN 表示查询为互联网地址。

1	0.000000000	10.0.2.15	223.5.5.5	DNS	86 Standard query 0x4dbb A www.xjtu.edu.cn OPT
2	0.000048403	10.0.2.15	223.5.5.5	DNS	86 Standard query 0x783c AAAA www.xjtu.edu.cn OPT
3	0.003822481	223.5.5.5	10.0.2.15	DNS	114 Standard query response 0x783c AAAA www.xjtu.edu.cn AAAA 2001...
4	0.004443660	223.5.5.5	10.0.2.15	DNS	102 Standard query response 0x4dbb A www.xjtu.edu.cn A 202.117.1...
Frame 1: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface enp0s3, id 0					
Ethernet II, Src: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)					
Internet Protocol Version 4, Src: 10.0.2.15, Dst: 223.5.5.5					
User Datagram Protocol, Src Port: 33754, Dst Port: 53					
Domain Name System (query)					
Transaction ID: 0x4dbb					
Flags: 0x0100 Standard query					
Questions: 1					
Answer RRs: 0					
Authority RRs: 0					
Additional RRs: 1					
Queries					
www.xjtu.edu.cn: type A, class IN					
Additional records					
[Response In: 4]					

DNS 响应报文返回域名所对应的 IP 地址。


```

4 0.004443660 223.5.5.5 10.0.2.15 DNS 102 Standard query response 0x4dbb A www.xjtu.edu.cn A 202.117.1...
Frame 4: 102 bytes on wire (816 bits), 102 bytes captured (816 bits) on interface enp0s3, id 0
  Ethernet II, Src: RealtekU 12:35:02 (52:54:00:12:35:02), Dst: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99)
  Internet Protocol Version 4, Src: 223.5.5.5, Dst: 10.0.2.15
  User Datagram Protocol, Src Port: 53, Dst Port: 33754
  Domain Name System (response)
    Transaction ID: 0x4dbb
    Flags: 0x8180 Standard query response, No error
    Questions: 1
    Answer RRs: 1
    Authority RRs: 0
    Additional RRs: 1
  Queries
    www.xjtu.edu.cn: type A, class IN
  Answers
    www.xjtu.edu.cn: type A, class IN, addr 202.117.1.13
  Additional records
    [Request In: 1]
    [Time: 0.004443660 seconds]

```

(四) TCP 协议

TCP 是为了在不可靠的互联网络上提供可靠的端到端字节流而专门设计的一个传输层协议。

在这个 Wireshark 抓包中，我们可以看到与访问 www.xjtu.edu.cn 的三次握手过程。在 TCP 连接建立过程中，主机先向服务器发送连接请求报文，此时报文中的 SYN 位置为 1。服务器接收到该报文后，会回复一个 SYN 和 ACK 位均置 1 的报文，以此确认主机发送的第一个 SYN 报文段。最后，主机再发送一个 ACK 位置 1 的确认报文，至此，双方完成连接建立。

491	3.254012107	10.0.2.15	202.117.1.13	TCP	74	34744 → 80	[SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM=1 T...
492	3.257329753	202.117.1.13	10.0.2.15	TCP	60	80 → 34744	[SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=1460
493	3.257373564	10.0.2.15	202.117.1.13	TCP	54	34744 → 80	[ACK] Seq=1 Ack=1 Win=64240 Len=0

三次握手不仅实现了可靠连接的搭建，还让双方确认了各自的初始序号。主机在发送连接建立请求报文时，携带的序号 Seq=0。服务器响应连接请求时，一方面确认了主机的起始序号 ACK=1，另一方面也发送了自身的起始序号 Seq=0。主机在最后的确认报文中，对服务器的起始序号 0 进行了确认 ACK=1。

同时，为实现最佳传输性能，TCP 在建立连接阶段，还需就双方可接受的最大报文长度以及窗口缩放因子等参数进行协商。

1738	8.411795880	10.0.2.15	202.117.1.13	TCP	54	34770 → 80	[FIN, ACK] Seq=6151 Ack=131467 Win=65535 Len=0
1739	8.411923244	10.0.2.15	202.117.1.13	TCP	54	34764 → 80	[FIN, ACK] Seq=6236 Ack=279364 Win=65535 Len=0
1740	8.411936497	10.0.2.15	202.117.1.13	TCP	54	34752 → 80	[FIN, ACK] Seq=3400 Ack=600199 Win=65535 Len=0
1741	8.411944492	10.0.2.15	202.117.1.13	TCP	54	34750 → 80	[FIN, ACK] Seq=5146 Ack=484019 Win=65535 Len=0
1747	8.412072001	202.117.1.13	10.0.2.15	TCP	60	80 → 34770	[ACK] Seq=131467 Ack=6152 Win=65535 Len=0
1748	8.412072026	202.117.1.13	10.0.2.15	TCP	60	80 → 34764	[ACK] Seq=279364 Ack=6237 Win=65535 Len=0
1749	8.412072051	202.117.1.13	10.0.2.15	TCP	60	80 → 34752	[ACK] Seq=600199 Ack=3401 Win=65535 Len=0
1750	8.412137608	202.117.1.13	10.0.2.15	TCP	60	80 → 34750	[ACK] Seq=484019 Ack=5147 Win=65535 Len=0
1766	8.412761517	10.0.2.15	202.117.1.13	TCP	54	34782 → 80	[FIN, ACK] Seq=3885 Ack=427959 Win=65535 Len=0
1776	8.413124551	202.117.1.13	10.0.2.15	TCP	60	80 → 34782	[ACK] Seq=427959 Ack=3886 Win=65535 Len=0
1786	8.413729581	10.0.2.15	202.117.1.13	TCP	54	34778 → 80	[FIN, ACK] Seq=4262 Ack=361154 Win=65535 Len=0
1791	8.413958949	202.117.1.13	10.0.2.15	TCP	60	80 → 34778	[ACK] Seq=361154 Ack=4263 Win=65535 Len=0
1798	8.416119616	202.117.1.13	10.0.2.15	TCP	60	80 → 34770	[FIN, ACK] Seq=131467 Ack=6152 Win=65535 Len=0
1799	8.416139192	10.0.2.15	202.117.1.13	TCP	54	34770 → 80	[ACK] Seq=6152 Ack=131468 Win=61648 Len=0
1800	8.416119730	202.117.1.13	10.0.2.15	TCP	60	80 → 34764	[FIN, ACK] Seq=279364 Ack=6237 Win=65535 Len=0
1801	8.416157246	10.0.2.15	202.117.1.13	TCP	54	34764 → 80	[ACK] Seq=6237 Ack=279365 Win=36188 Len=0
1802	8.416119769	202.117.1.13	10.0.2.15	TCP	60	80 → 34752	[FIN, ACK] Seq=600199 Ack=3401 Win=65535 Len=0
1803	8.416170884	10.0.2.15	202.117.1.13	TCP	54	34752 → 80	[ACK] Seq=3401 Ack=600200 Win=10088 Len=0
1804	8.416119810	202.117.1.13	10.0.2.15	TCP	60	80 → 34750	[FIN, ACK] Seq=484019 Ack=5147 Win=65535 Len=0
1805	8.416183709	10.0.2.15	202.117.1.13	TCP	54	34750 → 80	[ACK] Seq=5147 Ack=484020 Win=22572 Len=0
1808	8.416608976	202.117.1.13	10.0.2.15	TCP	60	80 → 34782	[FIN, ACK] Seq=427959 Ack=3886 Win=65535 Len=0
1809	8.416639167	10.0.2.15	202.117.1.13	TCP	54	34782 → 80	[ACK] Seq=3886 Ack=427960 Win=57284 Len=0
1810	8.416609035	202.117.1.13	10.0.2.15	TCP	60	80 → 34778	[FIN, ACK] Seq=361154 Ack=4263 Win=65535 Len=0
1811	8.416642638	10.0.2.15	202.117.1.13	TCP	54	34778 → 80	[ACK] Seq=4263 Ack=361155 Win=10400 Len=0

在 TCP 断开连接报文中可以发现 TCP 使用对称的连接释放方式，即对每个方向的连接单独释放。当关闭一个方向的连接时，发起方会发送一个带有 FIN 标志位的 TCP 报文。接收方在接收到 FIN 报文后，会发送 ACK 报文以确认，并通知应用程序该方向的通信已结束。当两个方向的连接都完成关闭后，TCP 两端的进程会删除该连接的记录，从而彻底释放该连接。

在上图中，对于每个连接只捕获到了连接释放的三个报文，这可能是由于 TCP 报文采用负载应答的方式，即服务器在收到 FIN 报文后，由于已无更多数据要传送给主机，服务器同时释放到主机的连接，故发送 ACK 报文的同时也将 FIN 位置 1。

在传输过程中，若服务器收到主机发送的 HTTP 请求报文后，便将完整的图片数据分为几个 TCP 报文传送给主机，并在最后发送一个 HTTP 响应报文表示一个对象的传输完成。主机在收到服务器发来的报文后，向服务器发送一个确认报文表示成功收到数据。

1222	3.588866326	10.0.2.15	202.117.1.13	HTTP	453 GET /img/tab-3.jpg HTTP/1.1
1223	3.589108725	202.117.1.13	10.0.2.15	TCP	60 80 → 34764 [ACK] Seq=258715 Ack=5854 Win=65535 Len=0
1224	3.590538603	202.117.1.13	10.0.2.15	TCP	2934 80 → 34782 [PSH, ACK] Seq=407996 Ack=3502 Win=65535 Len=2880 ...
1225	3.590547261	10.0.2.15	202.117.1.13	TCP	54 34782 → 80 [ACK] Seq=3502 Ack=410876 Win=65535 Len=0
1226	3.590538700	202.117.1.13	10.0.2.15	TCP	1494 80 → 34750 [PSH, ACK] Seq=463177 Ack=4242 Win=65535 Len=1440 ...
1227	3.590628400	202.117.1.13	10.0.2.15	TCP	1494 80 → 34782 [PSH, ACK] Seq=410876 Ack=3502 Win=65535 Len=1440 ...
1228	3.590628522	202.117.1.13	10.0.2.15	TCP	2934 80 → 34752 [PSH, ACK] Seq=582896 Ack=2590 Win=65535 Len=2880 ...
1229	3.590633195	10.0.2.15	202.117.1.13	TCP	54 34752 → 80 [ACK] Seq=2590 Ack=585776 Win=65535 Len=0
1230	3.590628554	202.117.1.13	10.0.2.15	HTTP	1410 HTTP/1.1 200 OK (PNG)

(五) HTTP 协议

HTTP 协议是基于 TCP 协议的应用层协议，是客户端和服务端进行数据传输的一种规则。

对 HTTP 协议使用 TCP 协议过程总结如下：

域名解析 → TCP 三次握手建立连接 → 客户端发起 http 请求 → 服务器响应 http 请求并发送 Web 页面文件 → 浏览器解析文件并请求资源 → 浏览器对页面进行渲染呈现给用户 → TCP 四次挥手释放连接

HTTP 请求报文：如图所示，HTTP 请求报文采用 GET 方法，向 Web 服务器请求获取文件，其 URL 为 <http://www.xjtu.edu.cn/>，使用的协议版本是 1.1。这个版本具备保持 TCP 连接的特性，这意味着同一对客户与服务器之间后续的请求和响应，都能通过该连接进行发送。甚至来自同一个 Web 服务器的多个 Web 页面，也可以借助单个持久 TCP 连接来传输。请求报文的头部字段涵盖了服务器主机 Host、连接类型 Connection 等关键信息。

+	494	3.258619340	10.0.2.15	202.117.1.13	HTTP	415 GET / HTTP/1.1
▶	Frame 494: 415 bytes on wire (3320 bits), 415 bytes captured (3320 bits) on interface enp0s3, id 0					
▶	Ethernet II, Src: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99), Dst: RealtekU_12:35:02 (52:54:00:12:35:02)					
▶	Internet Protocol Version 4, Src: 10.0.2.15, Dst: 202.117.1.13					
▶	Transmission Control Protocol, Src Port: 34744, Dst Port: 80, Seq: 1, Ack: 1, Len: 361					
▼	Hypertext Transfer Protocol					
▼	GET / HTTP/1.1\r\n					
▶	[Expert Info (Chat/Sequence): GET / HTTP/1.1\r\n]					
	Request Method: GET					
	Request URI: /					
	Request Version: HTTP/1.1					
	Host: 202.117.1.13\r\n					
	User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:133.0) Gecko/20100101 Firefox/133.0\r\n					
	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n					
	Accept-Language: zh-TW,zh;q=0.8,en-US;q=0.5,en;q=0.3\r\n					
	Accept-Encoding: gzip, deflate\r\n					
	Connection: keep-alive\r\n					
	Upgrade-Insecure-Requests: 1\r\n					
	Priority: u=0, i\r\n					
	\r\n					
	[Full request URI: http://202.117.1.13/]					
	[HTTP request 1/1]					
	[Response in frame: 502]					

HTTP 响应报文方面，图中的 HTTP 响应报文回复的状态码是 200，对应的状态语句为 OK，属于处理成功响应类别，表明请求动作已被成功接收、理解并接受。除了给出头部字段的值，响应报文在主体部分还呈现出请求的 Web 页面源码。

+	502	3.264656180	202.117.1.13	10.0.2.15	HTTP	570 HTTP/1.1 200 OK (text/html)
▶	Frame 502: 570 bytes on wire (4560 bits), 570 bytes captured (4560 bits) on interface enp0s3, id 0					
▶	Ethernet II, Src: RealtekU_12:35:02 (52:54:00:12:35:02), Dst: PcsCompu_4c:dc:99 (08:00:27:4c:dc:99)					
▶	Internet Protocol Version 4, Src: 202.117.1.13, Dst: 10.0.2.15					
▶	Transmission Control Protocol, Src Port: 80, Dst Port: 34744, Seq: 13141, Ack: 362, Len: 516					
▶	[4 Reassembled TCP Segments (13656 bytes): #496(2920), #498(4380), #500(5840), #502(516)]					
▼	Hypertext Transfer Protocol					
▼	HTTP/1.1 200 OK\r\n					
▶	[Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]					
	Response Version: HTTP/1.1					
	Status Code: 200					
	[Status Code Description: OK]					
	Response Phrase: OK					
	Date: Tue, 25 Feb 2025 16:53:44 GMT\r\n					
	Server: *****\r\n					
	X-Frame-Options: SAMEORIGIN\r\n					
	Upgrade: h2c,h2\r\n					
	Connection: Upgrade, Keep-Alive\r\n					
	Last-Modified: Tue, 25 Feb 2025 11:27:37 GMT\r\n					
	Accept-Ranges: bytes\r\n					
	Cache-Control: max-age=600\r\n					
	Expires: Tue, 25 Feb 2025 17:03:44 GMT\r\n					
	Vary: Accept-Encoding\r\n					
	Content-Encoding: gzip\r\n					
	ETag: "ddf9-62ef5be7d1634-gzip"\r\n					
▶	Content-Length: 13180\r\n					
	Keep-Alive: timeout=5, max=100\r\n					
	Content-Type: text/html\r\n					
	Content-Language: zh-CN\r\n					
	\r\n					
	[HTTP response 1/1]					
	[Time since request: 0.006036840 seconds]					
	[Request in frame: 494]					
	[Request URI: http://202.117.1.13/]					
	Content-encoded entity body (gzip): 13180 bytes -> 56825 bytes					
	File Data: 56825 bytes					
▶	Line-based text data: text/html (1010 lines)					

其中，可以看到最后的 HTTP/1.1 200 OK (text/html) 是 HTTP 响应的状态行和响应内容类型描述。200 OK 是状态码和状态短语，表示请求成功，服务器成功处理了客户端的请求；(text/html) 说明响应内容是 HTML 格式的文本，也就是通常看到的网页内容。

(六) UDP 协议

UDP 为用户数据报协议，是一种简单的面向消息的传输层协议，通过校验和对标头和有效负载进行完整性验证，不过它并不保证向上层协议传递消息，而且在消息发送后，不会留存 UDP 消息的状态。

基于这些特性，若对传输可靠性有要求，就必须在用户应用程序中自行实现相关机制。在本次实验中，截获的 UDP 报文如图所示。同时，诸如 DNS、NBNS、MDNS、SSDP 等报文，同样是借助 UDP 协议进行传输的。

324	1.420302931	34.117.188.166	10.0.2.15	UDP	1155	443 → 51165	Len=1113
325	1.420728035	10.0.2.15	34.117.188.166	UDP	84	51165 → 443	Len=42
332	1.452495090	10.0.2.15	34.117.188.166	UDP	173	51165 → 443	Len=131
333	1.452507012	10.0.2.15	34.117.188.166	UDP	111	51165 → 443	Len=69
335	1.474467336	10.0.2.15	180.163.151.33	UDP	1294	51367 → 443	Len=1252
363	1.626747517	34.117.188.166	10.0.2.15	UDP	614	443 → 51165	Len=572
364	1.626747837	34.117.188.166	10.0.2.15	UDP	167	443 → 51165	Len=125
365	1.626747863	34.117.188.166	10.0.2.15	UDP	68	443 → 51165	Len=26
366	1.627257786	10.0.2.15	34.117.188.166	UDP	73	51165 → 443	Len=31
438	2.106743923	10.0.2.15	180.163.151.33	UDP	1294	51367 → 443	Len=1252
505	3.349878732	10.0.2.15	180.163.151.33	UDP	1294	51367 → 443	Len=1252

(七) STP 协议

生成树协议是 IEEE 802.1D 标准的一部分，用于防止交换网络中的环路，确保以太网交换机之间的数据流不会陷入循环，影响网络通信。STP 通过选择根桥，计算最短路径，阻塞冗余链路，从而形成无环的树状拓扑。如果主链路失效，STP 还能自动启用备用链路，实现故障恢复。

由于在虚拟机中始终无法捕获到 STP 协议，因此编写相关 python 程序发送 STP 报文并抓包观察报文结构，具体代码如下。

```
from scapy.all import *
def send_stp(interface="eth0"):
    stp_bpdu = (
        Ether(dst="01:80:C2:00:00:00", src="00:11:22:33:44:55") /
        LLC(dsap=0x42, ssap=0x42, ctrl=0x03) /
        STP(
            rootid=0x1234,
            rootmac="00:11:22:33:44:55",
            bridgeid=0x5678,
            bridgemac="00:11:22:33:44:55",
            hellotime=2,
            maxage=20
        )
    )
    sendp(stp_bpdu, iface=interface, loop=1, inter=2)
send_stp(interface="enp0s8")
```

代码使用 Scapy 发送 STP 的 BPDU 报文，用于模拟 STP 交换机通信。定义函数 send_stp 用于构造并发送 STP BPDU 数据包并使用 Ether() 构造二层 Ethernet 帧。使用 LLC() 逻辑链路控制层封装 STP BPDU，其中 STP BPDU 报文构造如下。

rootid=0x1234 为根桥 ID，rootmac="00:11:22:33:44:55" 设置根桥 MAC 地址。0x5678 设为当前设备的桥 ID，用于 STP 计算路径。"00:11:22:33:44:55" 为当前设备的 MAC 地址。

每 2 秒发送一次 BPDU，最大存活时间默认是 20 秒。抓包结果如下。

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	CMSYS_33:44:55	Spanning-tree-(for- STP	52 Conf.	Root = 4996/564/00:11:22:33:44:55	Cost = 0 Port = 0x0
2	2.001129664	CMSYS_33:44:55	Spanning-tree-(for- STP	52 Conf.	Root = 4996/564/00:11:22:33:44:55	Cost = 0 Port = 0x0
3	4.002259328	CMSYS_33:44:55	Spanning-tree-(for- STP	52 Conf.	Root = 4996/564/00:11:22:33:44:55	Cost = 0 Port = 0x0
38	0.005349321	CMSYS_33:44:55	Spanning-tree-(for- STP	52 Conf.	Root = 4996/564/00:11:22:33:44:55	Cost = 0 Port = 0x0
58	0.007912697	CMSYS_33:44:55	Spanning-tree-(for- STP	52 Conf.	Root = 4996/564/00:11:22:33:44:55	Cost = 0 Port = 0x0

Frame 2: 52 bytes on wire (416 bits), 52 bytes captured (416 bits) on interface enp0s8, id 0
Ethernet II, Src: CMSYS_33:44:55 (00:11:22:33:44:55), Dst: Spanning-tree-(for-bridges)_00 (01:80:c2:00:00:00)
Logical-Link Control
Spanning Tree Protocol
Protocol Identifier: Spanning Tree Protocol (0x0000)
Protocol Version Identifier: Spanning Tree (0)
BPDU Type: Configuration (0x00)
BPDU flags: 0x00
Root Identifier: 4996 / 564 / 00:11:22:33:44:55
Root Path Cost: 0
Bridge Identifier: 20480 / 1656 / 00:11:22:33:44:55
Bridge Priority: 20480
Bridge System ID Extension: 1656
Bridge System ID: CMSYS_33:44:55 (00:11:22:33:44:55)
Port Identifier: 0x0000
Message Age: 1
Max Age: 20
Hello Time: 2
Forward Delay: 15

（八）其它协议

在本机其他端口进行抓包还有其他协议出现，如在桥接模式下进行抓包。

```
nakno@nakno-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::a00:27ff:fe4c:dc99 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:4c:dc:99 txqueuelen 1000 (Ethernet)
    RX packets 561 bytes 78658 (78.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 656 bytes 59374 (59.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.1.100 netmask 255.255.255.0 broadcast 192.168.1.255
    inet6 fe80::a00:27ff:fe5d:e3db prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:5d:e3:db txqueuelen 1000 (Ethernet)
    RX packets 966 bytes 116072 (116.0 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 249 bytes 22982 (22.9 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 261 bytes 24724 (24.7 KB)
```

MDNS 协议，其主要作用是在缺乏传统 DNS 服务器的情况下，让局域网内的主机能够相互发现并实现通信。MDNS 使用的端口是 5353，遵循 DNS 协议，沿用现有的 DNS 信息结构、语法以及资源记录类型，并且没有新增操作代码或响应代码。

2	1.365182259	fe80::3d0:8676:fa2d...	ff02::fb	MDNS	105 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QU" que...
3	1.427970296	192.168.1.100	224.0.0.251	MDNS	85 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QU" que...
4	2.375361524	fe80::3d0:8676:fa2d...	ff02::fb	MDNS	105 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QM" que...
5	2.453870491	192.168.1.100	224.0.0.251	MDNS	85 Standard query 0x0000 PTR _microsoft_mcc._tcp.local, "QM" que...

IGMP 协议即全称互联网组管理协议，专门负责 IPv4 组播成员的管理。其核心功能是在接收者主机和与之直接相邻的组播路由器之间，建立并维护组播组成员的关系。

450	117.973892629	192.168.1.1	224.0.0.1	IGMPv2	60 Membership Query, general
455	119.166738213	192.168.1.100	224.0.0.251	IGMPv2	60 Membership Report group 224.0.0.251
456	119.411119655	192.168.1.105	224.0.0.252	IGMPv2	60 Membership Report group 224.0.0.252
475	121.206584535	192.168.1.100	239.255.255.250	IGMPv2	60 Membership Report group 239.255.255.250
478	123.482668877	192.168.1.100	224.0.0.251	IGMPv2	46 Membership Report group 224.0.0.251
480	124.410120747	192.168.1.105	239.192.152.143	IGMPv2	60 Membership Report group 239.192.152.143
495	144.337156963	192.168.1.1	224.0.0.251	IGMPv2	60 Membership Query, specific for group 224.0.0.251

SSDP 协议是简单服务发现协议，这是一种应用层协议，为在局部网络中发现设备提供了相应机制，方便用户快速定位和连接网络中的设备。

364	51.706918950	192.168.1.1	239.255.255.250	SSDP	323 NOTIFY * HTTP/1.1
365	51.706919074	192.168.1.1	239.255.255.250	SSDP	332 NOTIFY * HTTP/1.1
366	51.706919189	192.168.1.1	239.255.255.250	SSDP	395 NOTIFY * HTTP/1.1
367	51.706919306	192.168.1.1	239.255.255.250	SSDP	332 NOTIFY * HTTP/1.1
368	51.706919422	192.168.1.1	239.255.255.250	SSDP	371 NOTIFY * HTTP/1.1
369	51.706919536	192.168.1.1	239.255.255.250	SSDP	332 NOTIFY * HTTP/1.1
370	51.706919655	192.168.1.1	239.255.255.250	SSDP	391 NOTIFY * HTTP/1.1

TLS 协议即安全传输层协议，主要用于在两个通信应用程序之间保障数据传输的保密性和完整性，防止数据在传输过程中被窃取或篡改。

259	30.284631635	49.7.63.220	192.168.1.100	TLSv1.2	467 Application Data
512	152.457330484	59.82.31.175	192.168.1.100	TLSv1.2	113 Application Data
513	152.457330598	59.82.31.175	192.168.1.100	TLSv1.2	289 Application Data
523	154.014922287	59.110.73.31	192.168.1.100	TLSv1.2	78 Application Data

NBNS 协议是是 TCP/IP 上的 NetBIOS (NetBT) 协议族的组成部分。它在基于 NetBIOS 名称访问的网络中，提供主机名和地址的映射方法，便于网络设备通过名称进行相互访问。

1162	331.541832766	192.168.1.100	192.168.1.255	NBNS	92 Name query NB LAPTOP-PF3QLJJB<1c>
1164	332.258497402	192.168.1.100	192.168.1.255	NBNS	92 Name query NB LAPTOP-PF3QLJJB<1c>
1165	333.069618063	192.168.1.100	192.168.1.255	NBNS	92 Name query NB LAPTOP-PF3QLJJB<1c>

（八）总结

通过本次实验，我掌握了使用 Wireshark 进行抓包的方法，深入了解了各协议请求与响应报文的格式。同时，通过分析抓包数据帮助我更好地理解网络通信的基本原理和协议的具体实现方式，让我对计算机网络原理有了更进一步的理解，使我受益匪浅。