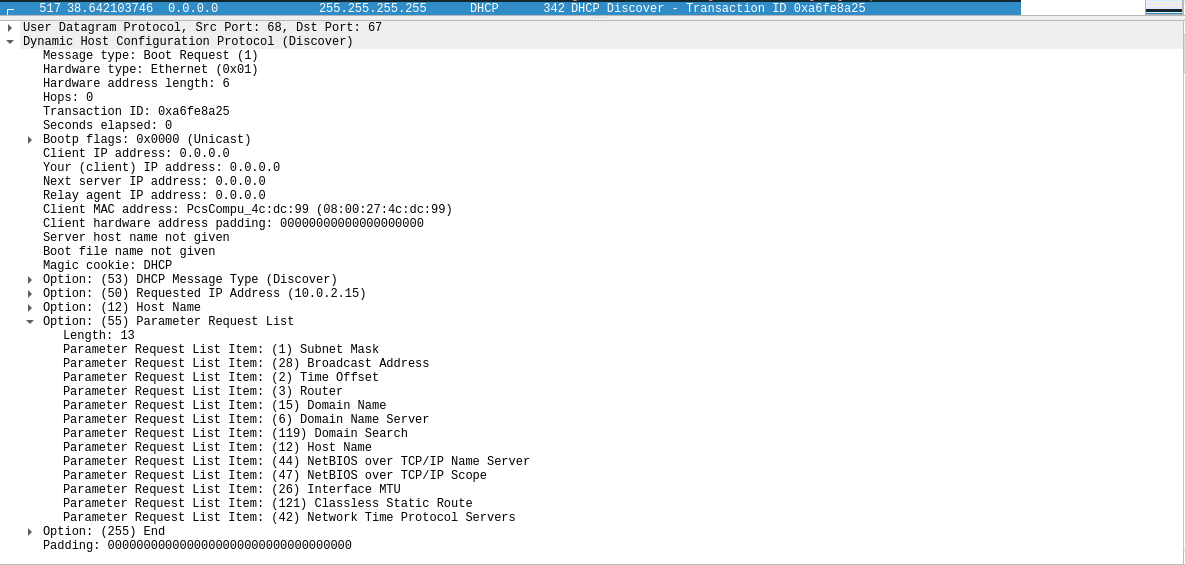
# 网络协议分析

（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。  


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。



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。

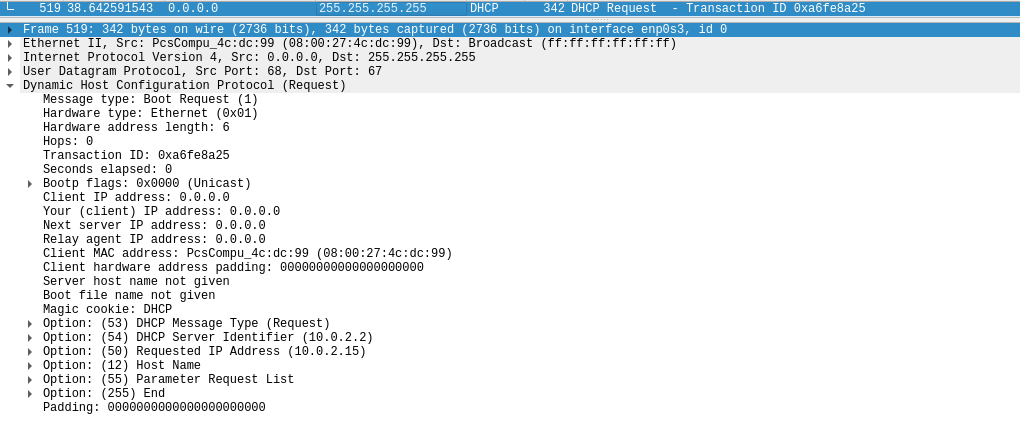
DHCP Offer：在DHCP服务器收到Discover报文后，就会在所配置的地址池中查找一个合适的IP地址，加上相应的租约期限和其他配置信息，建立Offer报文，发给DHCP客户端，告知用户可以为其提供IP地址。

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。

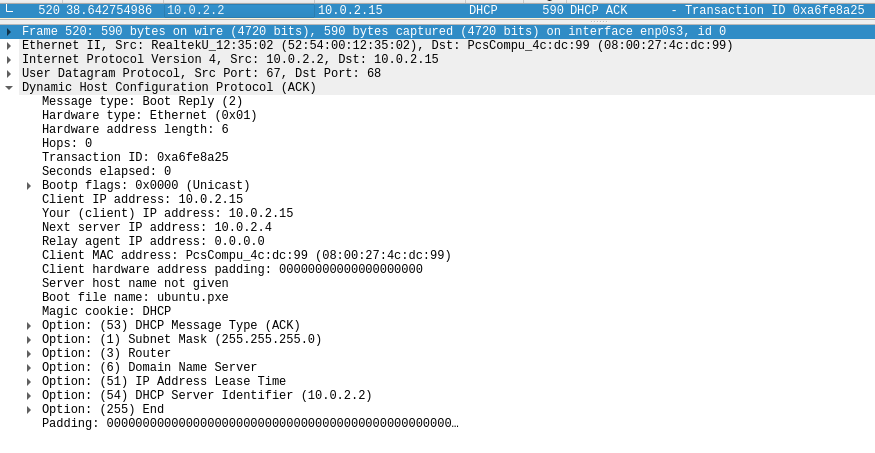


DHCP Request：在DHCP中，客户端在收到多个DHCP Offer报文后，需要选择一个最合适的服务器进行 IP 地址的分配。通常是选择第一个报文的服务器作为自己的目标服务器，并向该服务器发送一个广播的Request请求报文，通告选择的服务器，希望获得所分配的IP地址。

发现标志ID保持不变，但由于此时尚未分配到IP地址，故源地址仍为 0.0.0.0。报文中还存放客户端MAC地址，用以标志客户端。以及目标DHCP服务器的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。

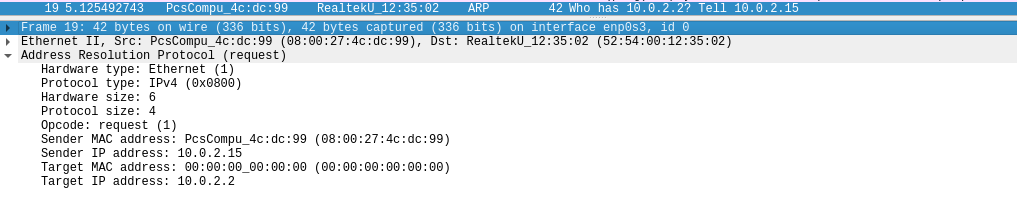


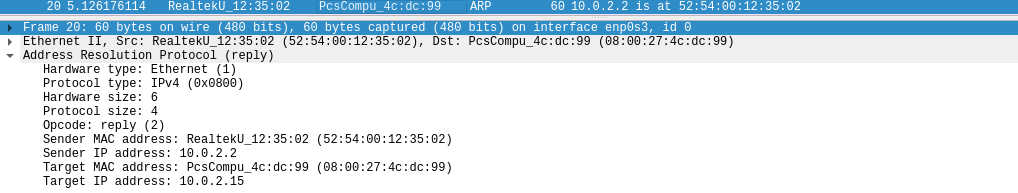
（二）ARP协议

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](http://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，主机会先通过ARP协议获取网关的MAC地址，然后由网关转发数据包。



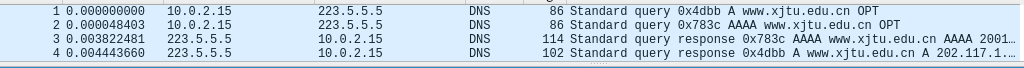
ARP请求报文：由于主机尚未知道网关的MAC地址，它会将包含网关IP地址的ARP请求报文广播到局域网中的所有主机，等待接收返回的消息，以便获取网关的MAC地址。

 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。

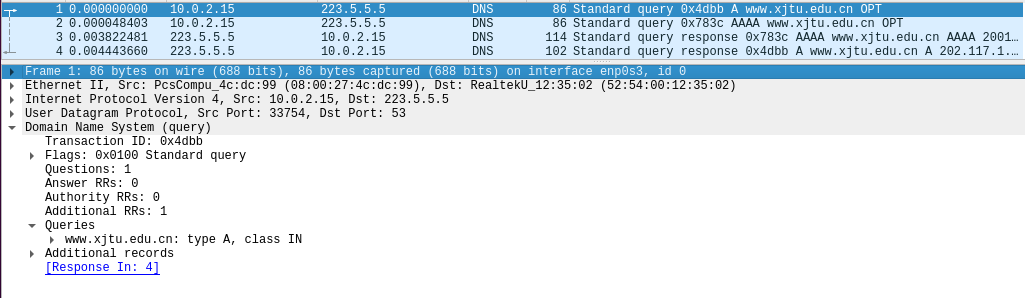


（三）DNS 协议

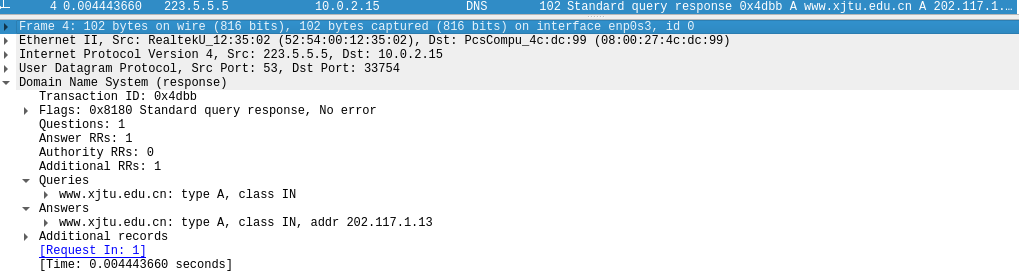
DNS是互联网的一项核心服务。它负责将域名和IP地址相互映射，允许终端设备将易于理解的URL转换为机器可以识别的IP地址，从而方便用户访问互联网。当主机访问 [www.xjtu.edu.cn](http://www.xjtu.edu.cn/)时，需要向DNS服务器 查询该域名对应的IP地址。



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。



DNS响应报文返回域名所对应的IP地址。



（四）TCP协议

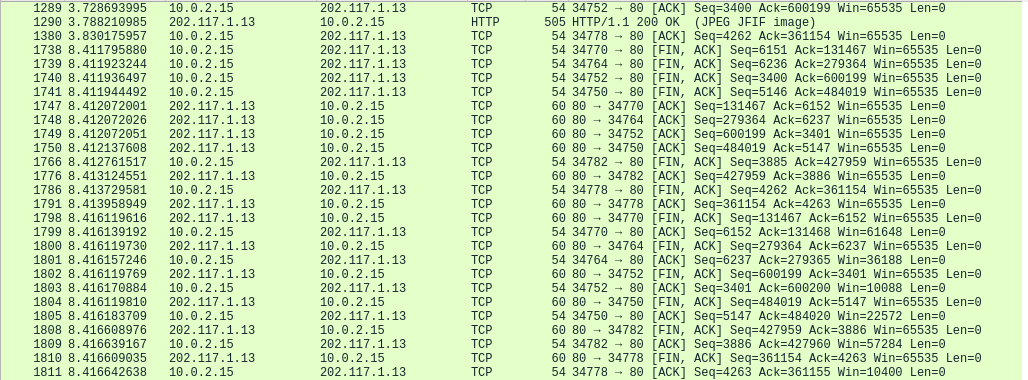
TCP是为了在不可靠的互联网络上提供可靠的端到端字节流而专门设计的一个传输层协议。

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。

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。

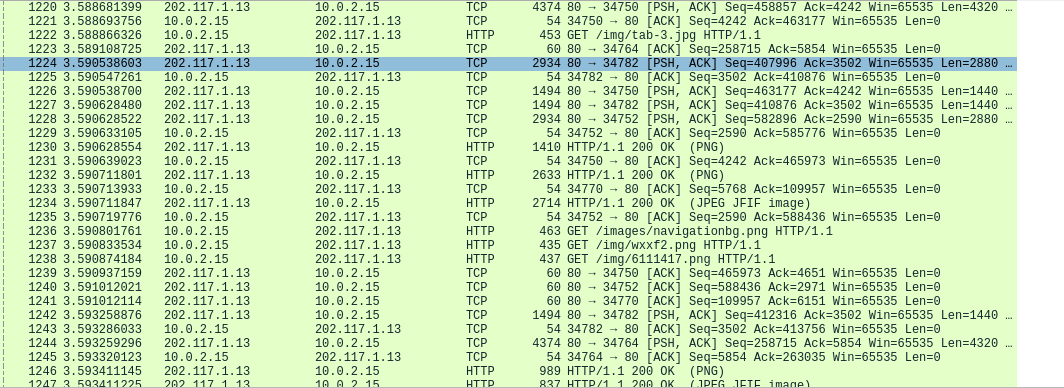
同时，为实现最佳传输性能，TCP在建立连接阶段，还需就双方可接受的最大报文长度以及窗口缩放因子等参数进行协商。



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。

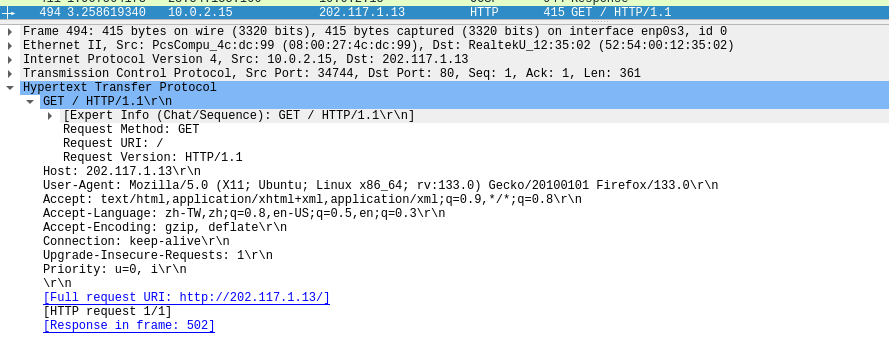


（五）HTTP 协议

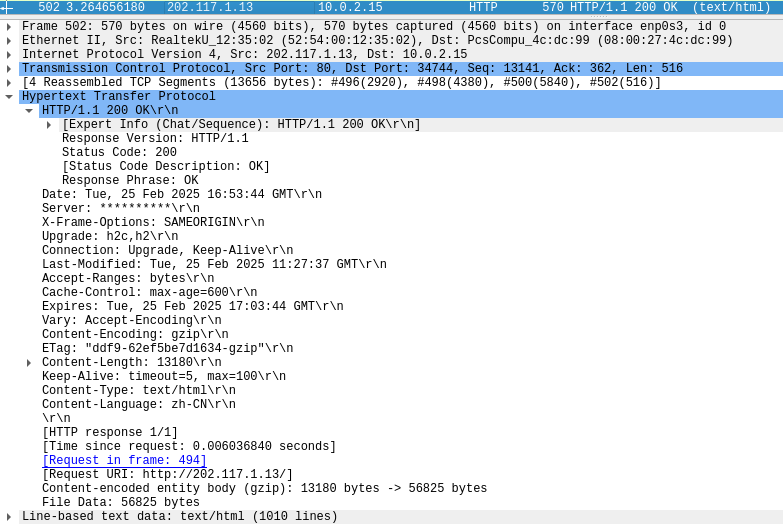
HTTP协议是基于TCP协议的应用层协议，是客户端和服务端进行数据传输的一种规则。 对HTTP协议使用TCP协议过程总结如下：

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/](http://www.xjtu.edu.cn/" \t "_blank)，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。



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。



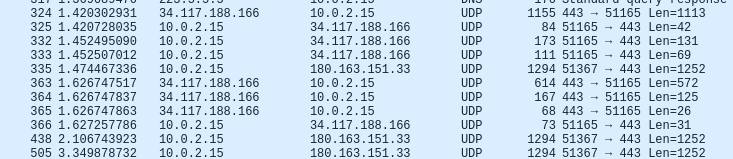
其中，可以看到最后的HTTP/1.1 200 OK (text/html)是HTTP响应的状态行和响应内容类型描述。200 OK是状态码和状态短语，表示请求成功，服务器成功处理了客户端的请求；(text/html)说明响应内容是HTML格式的文本，也就是通常看到的网页内容。

（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。



1. 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

)

)

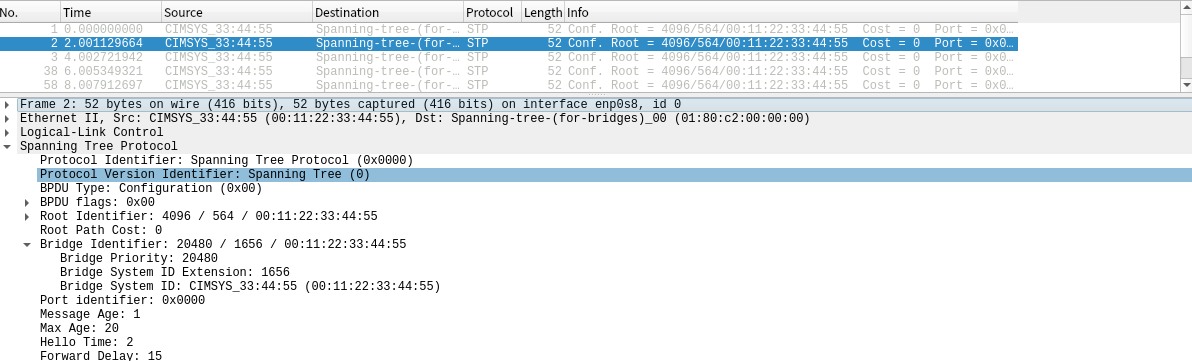
sendp(stp\_bpdu, iface=interface, loop=1, inter=2)

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。

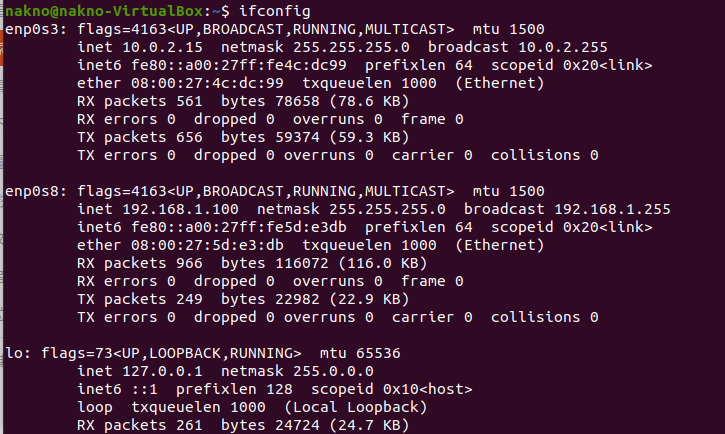
rootid=0x1234为根桥 ID，rootmac="00:11:22:33:44:55"设置根桥 MAC 地址。0x5678设为当前设备的桥 ID，用于 STP 计算路径。"00:11:22:33:44:55"为当前设备的 MAC 地址。

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。



（八）其它协议

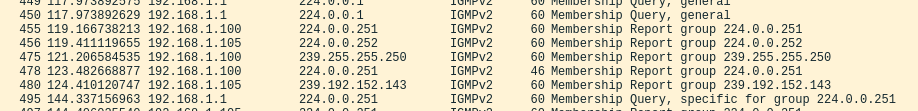
在本机其他端口进行抓包还有其他协议出现，如在桥接模式下进行抓包。



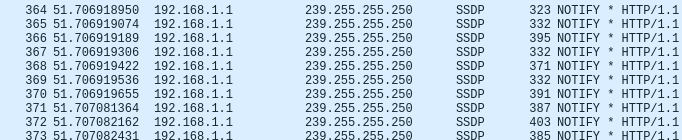
MDNS协议，其主要作用是在缺乏传统DNS服务器的情况下，让局域网内的主机能够相互发现并实现通信。MDNS 使用的端口是 5353 ，遵循 DNS 协议，沿用现有的 DNS 信息结构、语法以及资源记录类型，并且没有新增操作代码或响应代码。



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。



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。



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。



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。



（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了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。