DNS、TCP、UDP 协议分析

一、实验目的

- 了解 DNS、TCP、UDP 协议的工作原理

二、实验任务

- 通过 Wireshark 分析 DNS、TCP、UDP 协议

三、实验计划

实验时间	实验内容
第一周	DNS 协议分析
第二周	TCP 协议分析
第三周	UDP 协议分析

四、实验过程

DNS 协议

预备知识

nslookup 工具及使用

`nslookup` 工具在现在的大多数 Linux/Unix 和 Microsoft 平台中都有，它允许主机查询任何指定的 DNS 服务器的 DNS 记录。DNS 服务器可以是根 DNS 服务器，顶级域 DNS 服务器，权威 DNS 服务器或中间 DNS 服务器。要完成此任务，nslookup 将 DNS 查询发送到指定的 DNS 服务器，然后接收 DNS 回复，并显示结果。

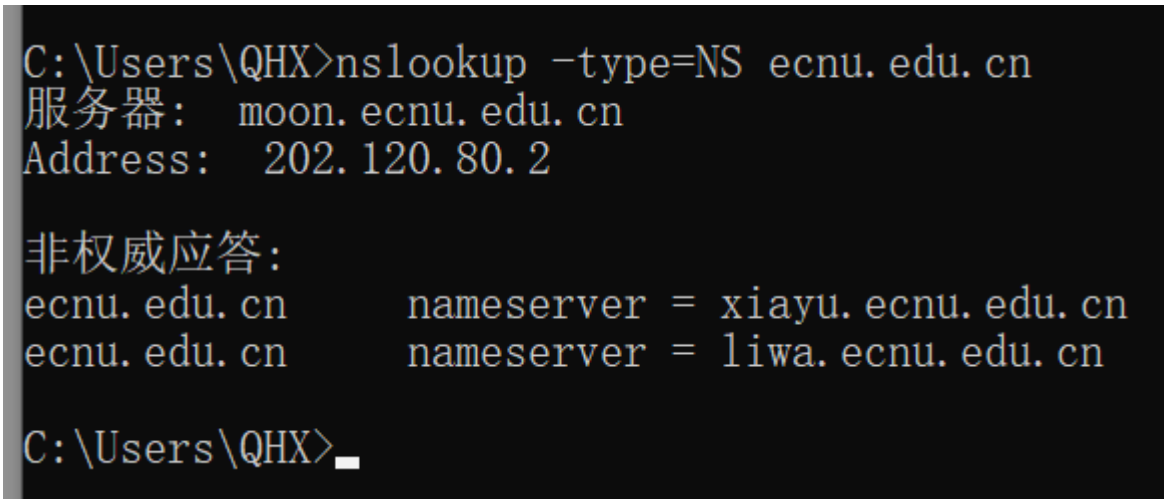
下面截图显示了三个不同 `nslookup` 命令的结果（显示在 Mac终端，Win类似）。运行 `nslookup` 时，如果没有指定 DNS 服务器，则 `nslookup` 会将查询发送到默认的本地DNS 服务器。

- `nslookup www.ecnu.edu.cn`

  这个命令是说，请告诉我主机 www.ecnu.edu.cn 的 IP 地址。如上图所示，此命令的响应提供两条信息：（1）提供响应的 DNS 服务器的名称和 IP 地址；（2）响应本身，即 www.ecnu.edu.cn 的主机名和 IP 地址。虽然响应来自ecnu的本地 DNS 服务器，但本地 DNS 服务器很可能会迭代地联系其他几个 DNS 服务器来获得结果。

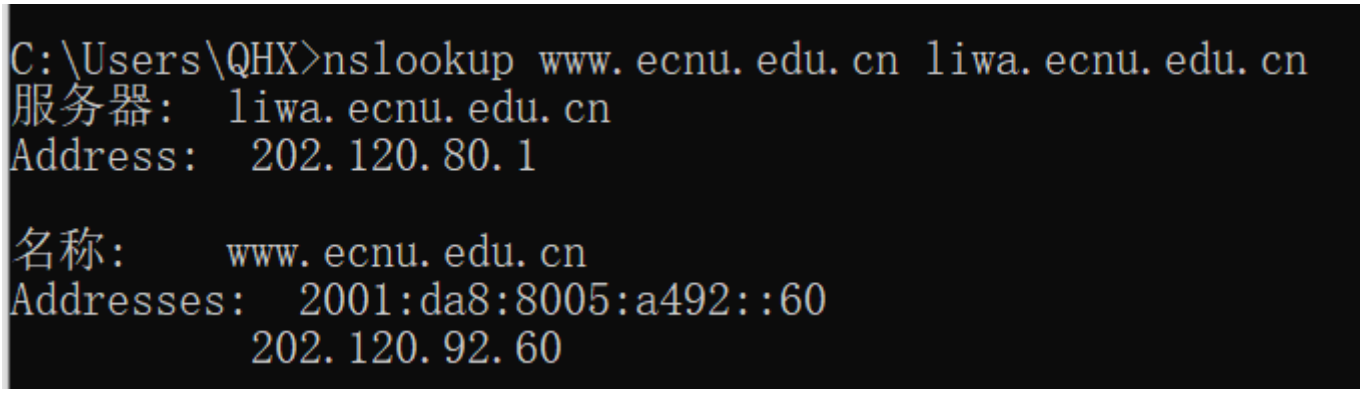
- `nslookup -type=NS ecnu.edu.cn`：查询权威DNS

  在这个例子中，我们添加了选项 `-type=NS` 和一级域名 `ecnu.edu.cn`。这将使得 `nslookup` 将 `NS（域名服务器记录，Name Server）` 记录发送到默认的本地 DNS 服务器。换句话说，“请给我发送 `ecnu.edu.cn` 的权威 DNS 的主机名”（当不使用 `-type` 选项时，`nslookup` 使用默认值，即查询 A 类记录。）上图中，首先显示了提供响应的 DNS 服务器（这是默认本地 DNS 服务器）以及两个 `ecnu` 域名服务器。这些服务器中的每一个确实都是校园主机的权威 DNS 服务器。然而，`nslookup` 也表明该响应是非权威的，这意味着这个响应来自某个服务器的缓存，而不是来自权威 ecnu DNS 服务器。



- `nslookup www.ecnu.edu.cn liwa.ecnu.edu.cn`

  在这个例子中，我们希望将查询请求发送到 DNS 服务器 `liwa.ecnu.edu.cn`，而不是默认的本地 DNS 服务器 。因此，查询和响应事务直接发生在我们的主机和 `liwa.ecnu.edu.cn` 之间。在这个例子中，DNS 服务器 `liwa.ecnu.edu.cn` 提供主机 www.ecnu.edu.cn 的 IP 地址信息。



- `nslookup` 语法：`nslookup -option1 -option2 host-to-find dns-server`

一般来说，`nslookup` 可以不添加选项，或者添加一两个甚至更多选项。正如我们在上面的示例中看 到的，`dns-server` 也是可选的；如果这项没有提供，查询将发送到默认的本地 DNS 服务器。

DNS协议

- 识别主机有两种方式：主机名、IP地址。前者便于记忆(如www.baidu.com)，但路由器很难处理(主机名长度不定)；后者定长、有层次结构，便于路由器处理，但难以记忆。
- 折中的办法就是建立IP地址与主机名间的映射，这就是域名系统DNS做的工作。
- DNS通常由其他应用层协议使用(如HTTP、SMTP、FTP)，将主机名解析为IP地址。
- 在本实验中，我们将仔细查看 DNS 报文的细节。

DNS 报文

- **报文格式**

  DNS只有两种报文：查询报文、响应报文，两者有着相同格式，如下：

- **捕获的DNS报文**

> 实验开始前请先清空dns缓存
> win: ipconfig/flushdns
> mac: sudo killall -HUP mDNSResponder; sudo dscacheutil -flushcache

1. 考虑对访问百度页面的一个操作抓包，在浏览器输入 http://www.baidu.com/index.html 并回车（必要时需清空浏览器缓存），首先需要将 URL (**存放对象的服务器主机名和对象的路径名**) 解析成IP地址，具体步骤为：

```
同一台用户主机上运行着 DNS 应用的客户机端(如浏览器)
从上述URL抽取主机名[www.baidu.com](http://www.baidu.com/)，传给 DNS 应用的客户机端(浏览器)
该 DNS 客户机向 DNS 服务器发送一个包含主机名的请求(DNS 查询报文)
该 DNS 客户机收到一份回答报文(DNS 响应报文)，该报文包含该主机名对应的IP地址 202.120.80.2
浏览器由该 IP 地址定位的 HTTP 服务器发送一个 TCP 链接
```

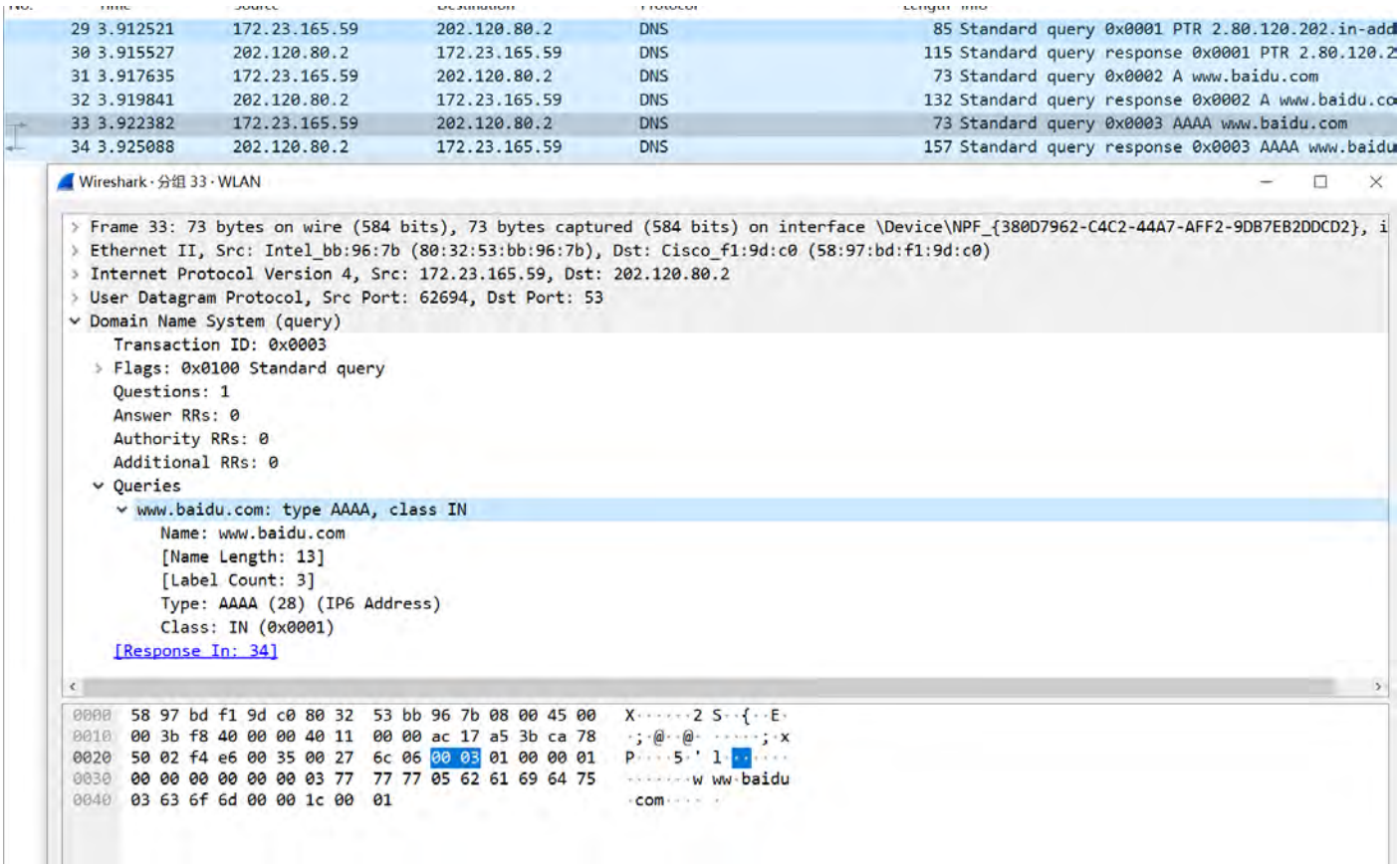
2. 或通过命令 `nslookup www.baidu.com`

用Wireshark捕获的DNS报文如下图，第一行（编号33）是DNS查询报文，第二行（编号34）是DNS响应报文。

实验操作

实验任务

- **task1: 运行 `nslookup` 来确定一个国外大学 (www.mit.edu) 的IP地址以及其权威 DNS 服务器，请在实验报告中附上操作截图并详细分析返回信息内容。**
- **task2: 运行 `nslookup`，使用task1中一个已获得的 DNS 服务器，来查询google服务器 (www.google.com)的 IP 地址(可直接查询)，请在实验报告中附上操作截图并详细分析返回信息内容。**
- **task3: 根据Wireshark抓取的报文信息（例，下图所示示例），分别分析DNS查询报文和响应报文的组成结构，参考上面的报文格式指出报文的每个部分（如，头部区域等），请将实验结果附在实验报告中。**
- **task4: 基于task3中得到的查询和响应报文进行分析，试问这里的查询是什么“Type"的，查询消息是否包含任何“answers"？试问这里的响应消息提供了多少个“answers”，这些“answers”具体包含什么？请将实验结果附在实验报告中。**

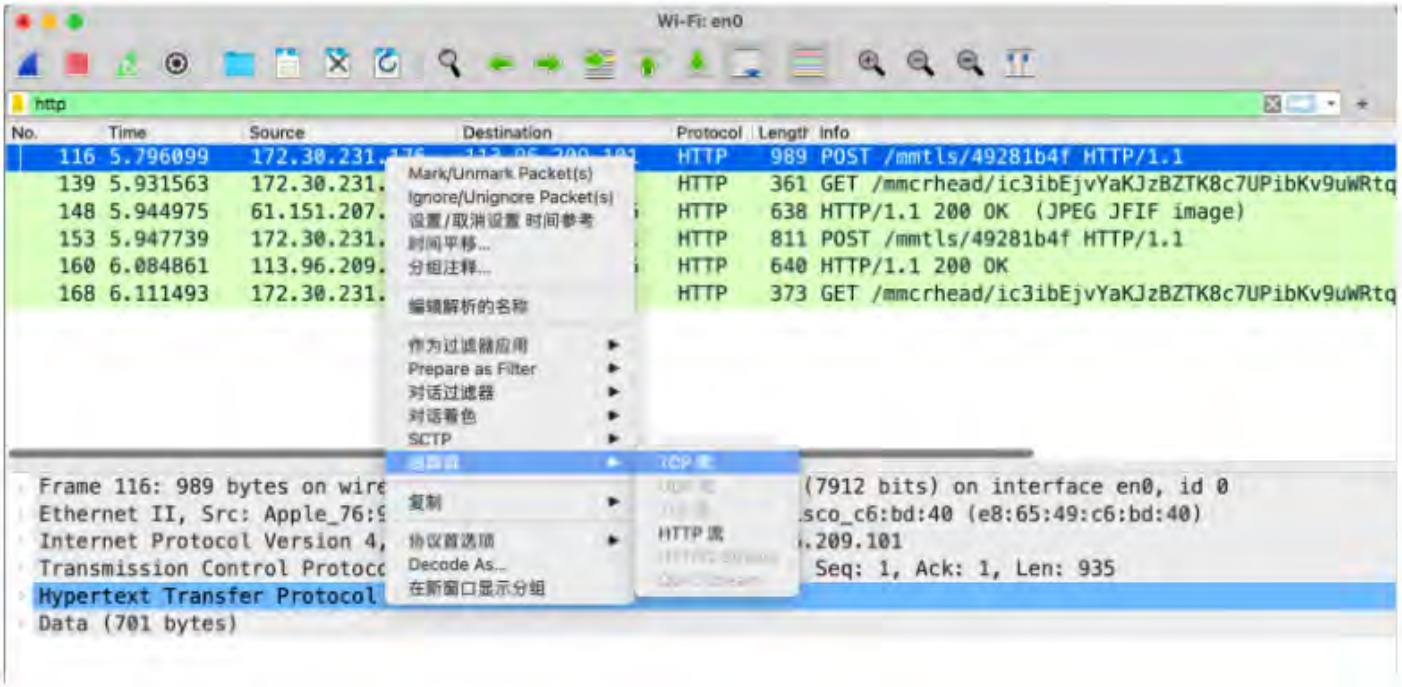


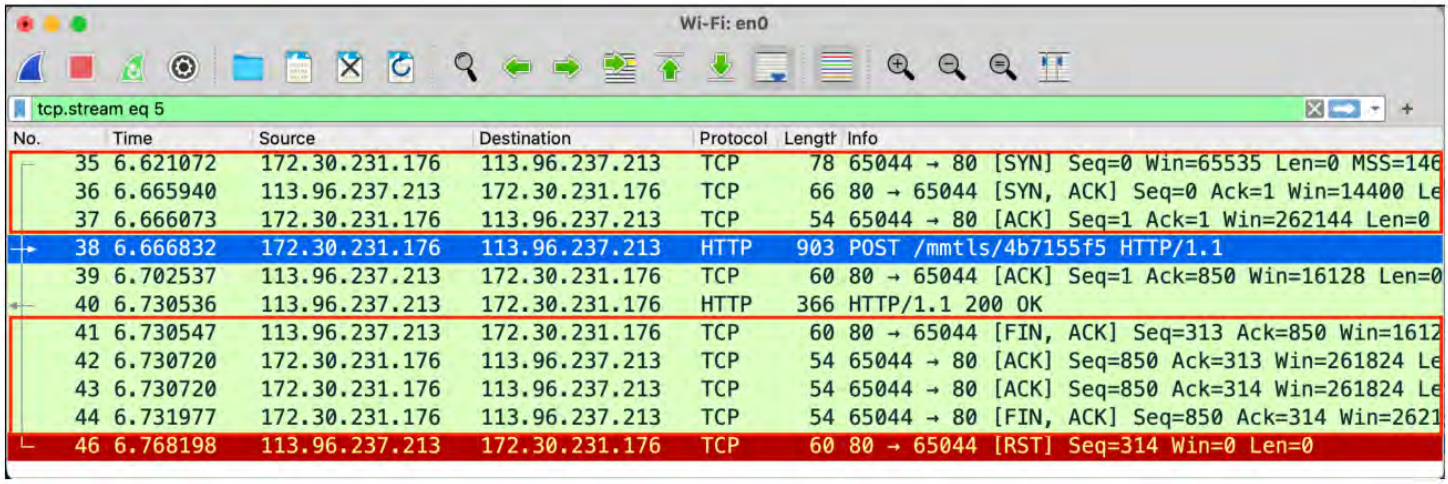
TCP 协议

预备知识

TCP是因特网运输层的**面向连接的可靠的运输协议。**TCP 被称为是面向连接的,这是因为在一个应用进程可以开始 向另一个应用进程发送数据之前，这两个进程必须先**相互“握手”**，即它们必须相互发送某些预备报文段，以建立确保数据传输的参数。作为 TCP 连接建立的一部分，连接的双方都将初始化与 TCP 连接相关的许多 TCP 状态变量。

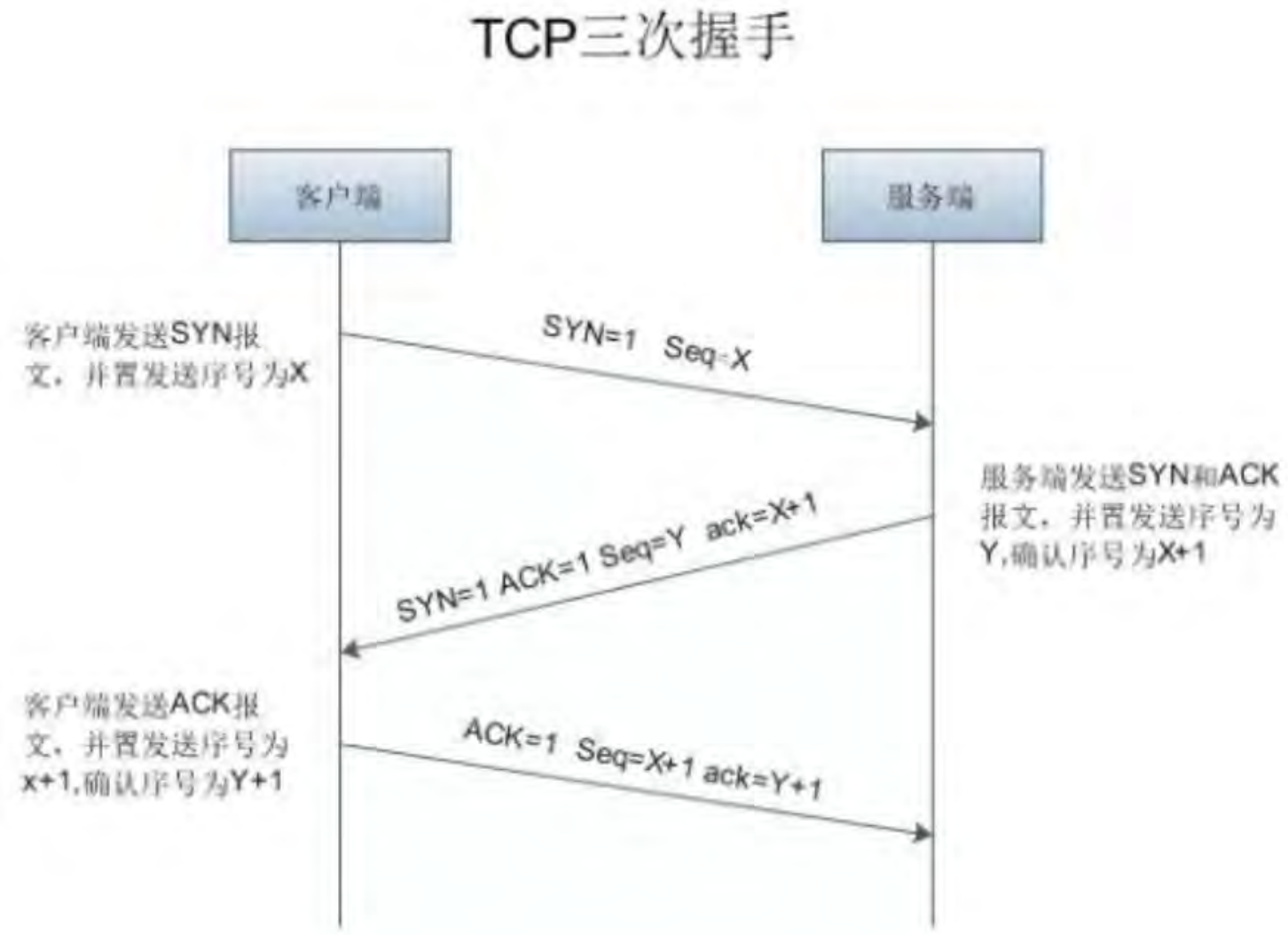
图 3-29　TCP 报文段结构





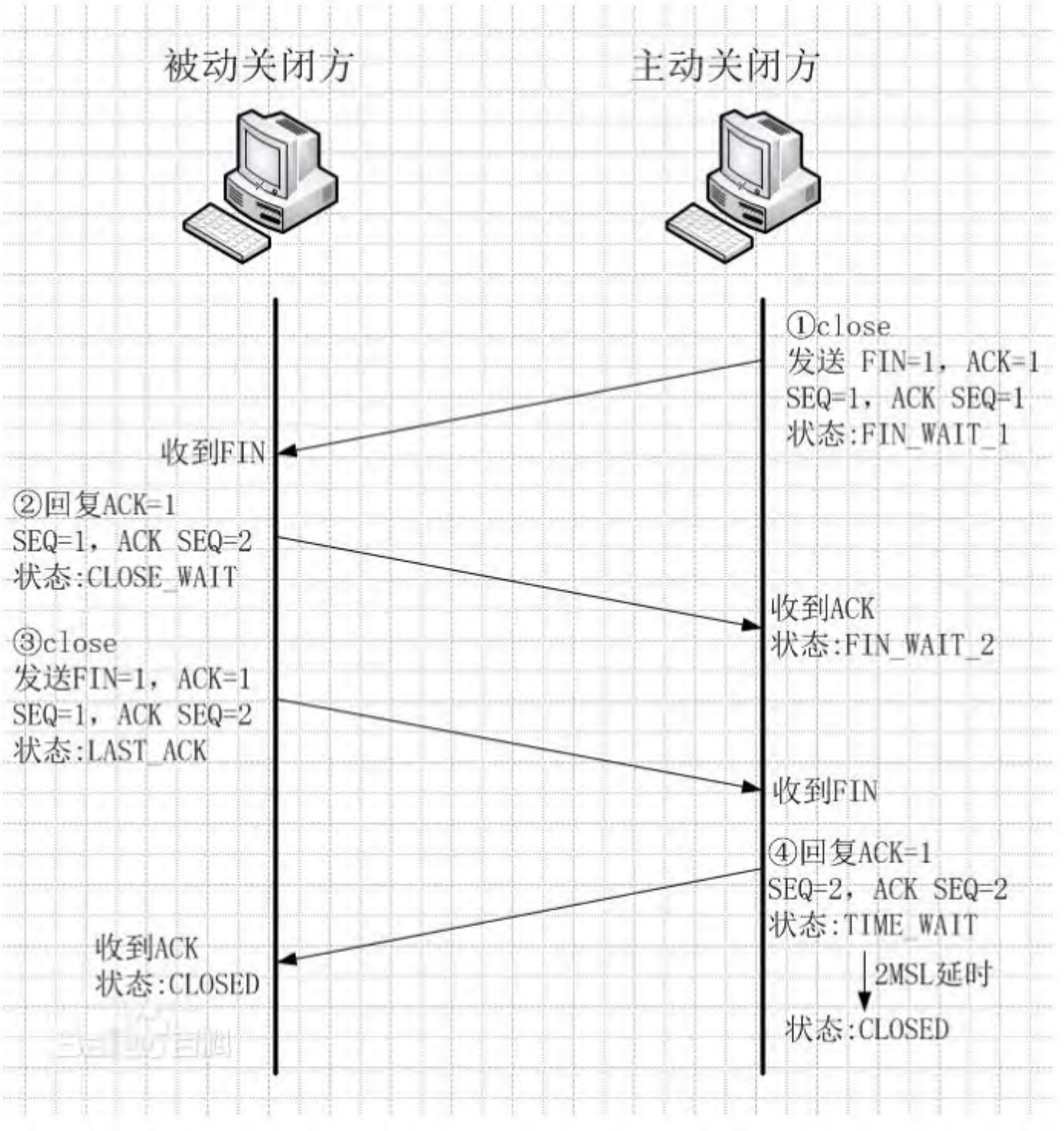
TCP三次握手

TCP建立连接时，会有三次握手过程，如下图所示，Wireshark截获到了三次握手的三个数据包。第四个包才是http的，说明http的确是使用TCP建立连接的。



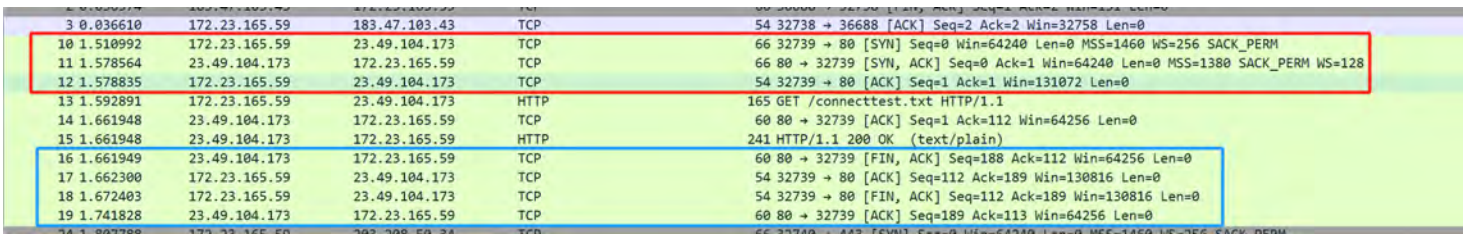
TCP四次挥手

当通信双方完成数据传输，需要进行TCP连接的释放，由于TCP连接是全双工的，因此每个方向都必须单独进行关闭。这个原则是当一方完成它的数据发送任务后就能发送一个FIN来终止这个方向的连接。收到一个 FIN只意味着这一方向上没有数据流动，一个TCP连接在收到一个FIN后仍能发送数据。首先进行关闭的一方将执行主动关闭，而另一方执行被动关闭。因为正常关闭过程需要发送4个TCP帧，因此这个过程也叫作4次挥手。



实验操作

1. 开启 Wireshark 捕获
2. 浏览器访问 www.qq.com
3. 关闭第 2 步访问打开的 QQ 标签页
4. 停止 Wireshark 捕获，捕获结果包括三次握手和四次挥手如下图所示



实验任务

task1: 利用Wireshark抓取一个TCP数据包，查看其具体数据结构和实际的数据（要求根据报文结构正确标识每个部分），请将实验结果附在实验报告中。

task2: 根据TCP三次握手的交互图和抓到的TCP报文详细分析三次握手过程，请将实验结果附在实验报告中。

task3: 根据TCP四次挥手的交互图和抓到的TCP报文详细分析四次挥手过程，请将实验结果附在实验报告中。

UDP 协议

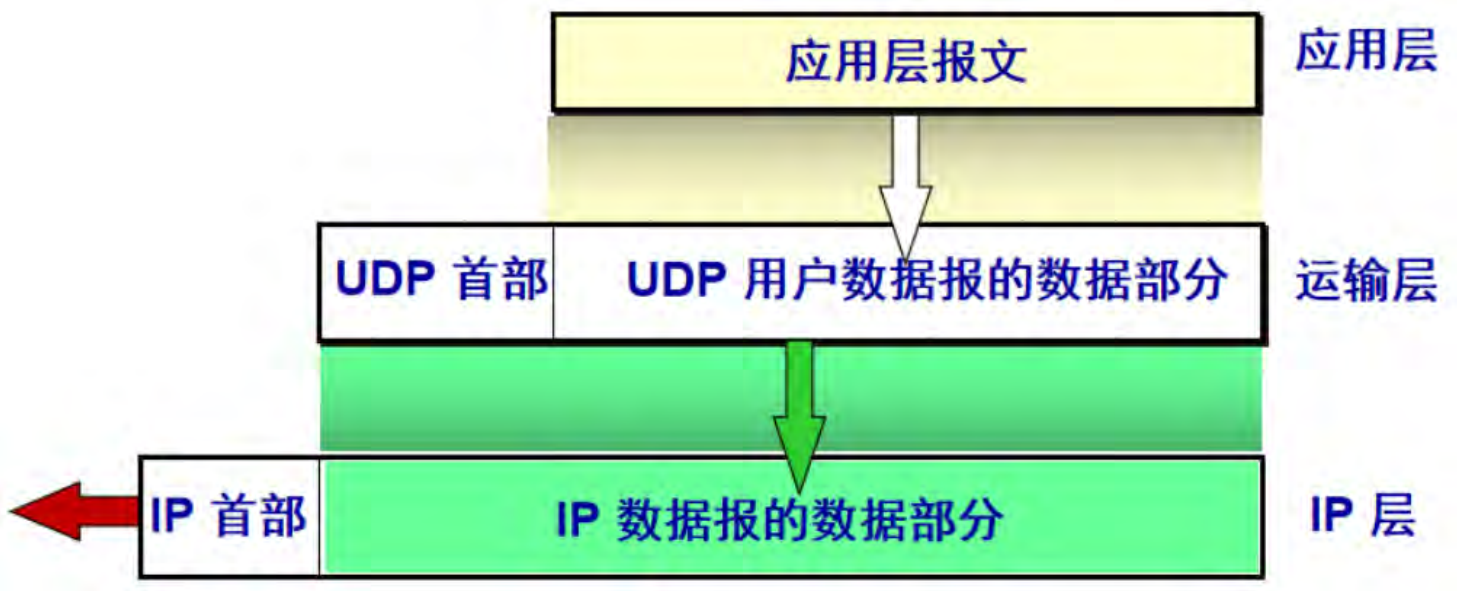
预备知识

用户数据报(UDP)协议是运输层提供的一种最低限度的复用/分解服务，可以在网络层和正确的用户即进程间传输数据。

UDP 是一种不提供不必要服务的轻量级运输协议，除了**复用/分用**功能和简单的**差错检测**之外，几乎就是 IP 协议了，也可以说它仅提供**最小**服务。UDP 是**无连接**的，因此在两个进程通信前**没有握手过程。**

UDP 协议提供一种不可靠数据传输服务，也就是说，当一个进程讲一个报文发送进 UDP 套接字时，UDP 协议**并不保证**该报文将到达接收进程。也正是由于 UDP 不修复错误，因此到达接收进程的报文也可能是乱序到达的。

UDP 是面向报文的，这是因为 UDP 并不会对应用层传递下来的报文进行任何处理，对于报文的边界信息都会保存，向下交付时交付的是完整报文。

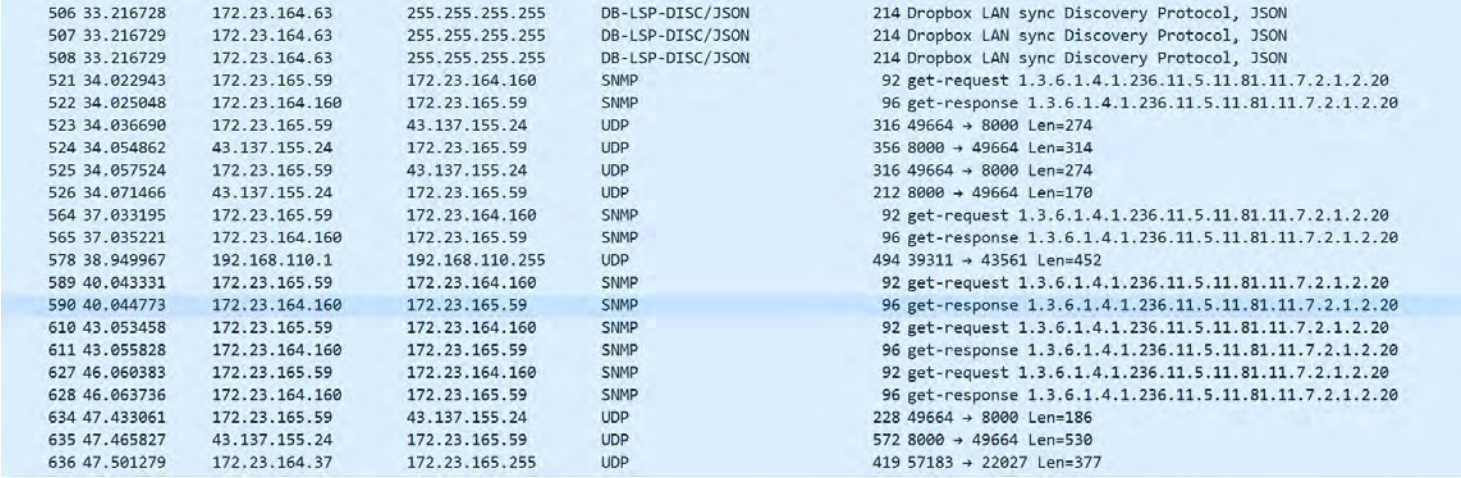


UDP 首部只有 4 个字段：源端口号、目的端口号、长度、校验和，其中每个字段由 2 个字节组成。

实验操作

1. 在 Wireshark 中捕获数据包，然后执行一些会导致主机发送和接收多个 UDP 数据包的操作。**也可以什么也不做，等待一定时间**，仅执行 wireshark 捕获以便获取其他程序发给您的 UDP 数据包。有一种特殊情况：简单网络管理协议（SNMP）在 UDP 内部发送 SNMP 消息，因此可能会在跟踪中找到一些 SNMP 消息（以及 UDP 数据包）。

2. 停止数据包捕获后，设置数据包筛选器，以便 Wireshark 仅显示在主机上发送和接收的 UDP 数据包。 选择其中一个 UDP 数据包并在详细信息窗口中展开 UDP 字段。



506	33.216728	172.23.164.63	255.255.255.255	DB-LSP-DISC/JSON	214 Dropbox LAN sync Discovery Protocol, JSON
507	33.216729	172.23.164.63	255.255.255.255	DB-LSP-DISC/JSON	214 Dropbox LAN sync Discovery Protocol, JSON
508	33.216729	172.23.164.63	255.255.255.255	DB-LSP-DISC/JSON	214 Dropbox LAN sync Discovery Protocol, JSON
521	34.022943	172.23.165.59	172.23.164.160	SNMP	92 get-request 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
522	34.025048	172.23.164.160	172.23.165.59	SNMP	96 get-response 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
523	34.036690	172.23.165.59	43.137.155.24	UDP	316 49664 → 8000 Len=274
524	34.054862	43.137.155.24	172.23.165.59	UDP	356 8000 → 49664 Len=314
525	34.057524	172.23.165.59	43.137.155.24	UDP	316 49664 → 8000 Len=274
526	34.071466	43.137.155.24	172.23.165.59	UDP	212 8000 → 49664 Len=170
564	37.033195	172.23.165.59	172.23.164.160	SNMP	92 get-request 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
565	37.035221	172.23.164.160	172.23.165.59	SNMP	96 get-response 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
578	38.949967	192.168.110.1	192.168.110.255	UDP	494 39311 → 43561 Len=452
589	40.043331	172.23.165.59	172.23.164.160	SNMP	92 get-request 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
590	40.044773	172.23.164.160	172.23.165.59	SNMP	96 get-response 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
610	43.053458	172.23.165.59	172.23.164.160	SNMP	92 get-request 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
611	43.055828	172.23.164.160	172.23.165.59	SNMP	96 get-response 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
627	46.060383	172.23.165.59	172.23.164.160	SNMP	92 get-request 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
628	46.063736	172.23.164.160	172.23.165.59	SNMP	96 get-response 1.3.6.1.4.1.236.11.5.11.81.11.7.2.1.2.20
634	47.433061	172.23.165.59	43.137.155.24	UDP	228 49664 → 8000 Len=186
635	47.465827	43.137.155.24	172.23.165.59	UDP	572 8000 → 49664 Len=530
636	47.501279	172.23.164.37	172.23.165.255	UDP	419 57183 → 22027 Len=377

实验任务

task1: 从跟踪中选择一个 UDP 数据包。从此数据包中，识别并确定 UDP 首部字段，请为这些字段命名并将实验结果附在实验报告中。

task2: UDP首部中的长度字段指的是什么，以及为什么需要这样设计？使用捕获的 UDP 数据包进行验证，请将实验结果附在实验报告中。

task3: UDP 有效负载中可包含的最大字节数是多少？请将实验结果附在实验报告中。

首先先认识下**有效负载**：

> 有效负载是被传输数据中的一部分，而这部分才是数据传输的最基本的目的，和有效负载一同被传送的数据还有：数据头或称作元数据，有时候也被称为开销数据，这些数据用来辅助数据传输。
> ——百度百科

task4: 观察发送 UDP 数据包后接收响应的 UDP 数据包，这是对发送的 UDP 数据包的回复，请描述两个数据包中端口号之间的关系。(提示：对于响应 UDP 目的地应该为发送 UDP 包的地址。）请将实验结果附在实验报告中。

五、实验报告

- 将三次协议分析的实验任务汇总完成一个实验报告
- 实验报告中的上机实践名称为**DNS、TCP、UDP协议分析实验**，上机时间编号、组号、上机实践时间不用填写，提交文件名称格式：学号+姓名+第四次实验。