

# 计算机网络第一次实验报告

## 一、实验内容与实验步骤

### 1. 实验内容：

- 1) 使用 Wireshark 捕获 HTTP 消息，分析消息头，理解 HTTP 的通信原理；
- 2) 使用 Wireshark 捕获依次从客户端发送 Email 的过程，分析 SMTP 消息，理解 Email 系统中发送邮件的通信原理；
- 3) 使用 Telnet 软件访问 Email 服务器，输入 SMTP 命令与 Email 服务器交互，理解 SMTP 的通信过程和 Base64 编码的概念。

### 2. 实验环境：

Windows 11 操作系统的联网主机，安装 Wireshark。

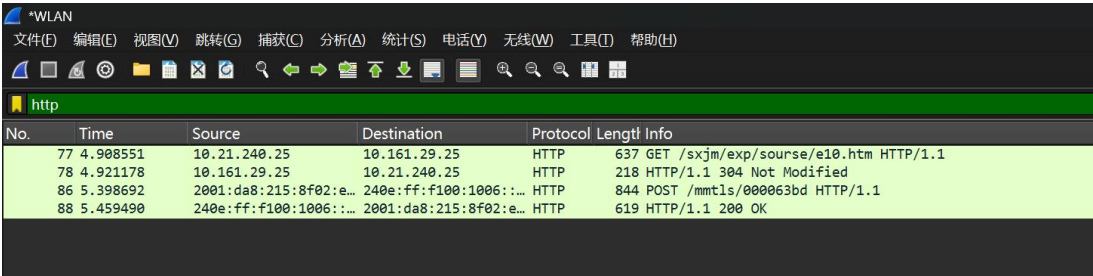
### 3. 实验步骤：

- 1) **准备工作：**下载 Wireshark，了解使用方法和功能，保证计算机连接到网络，启动 Wireshark 选择 WLAN 捕获接口，设置合适过滤器；
- 2) **数据捕获：**
  - **捕获 HTTP 协议数据：**打开浏览器清除 Cookie，选择一个非 HTTPS 的网站访问（本次实验使用 URL 为 <http://10.161.29.25/sxjm/exp/sourse/e10.htm>），全部显示后停止捕获；
  - **捕获 SMTP 协议数据：**下载并安装邮件客户端软件（本次实验使用 Foxmail），配置用户账户，设置发件服务器不选择 SSL，端口为 25，用 Foxmail 发送一封邮件，邮件发送成功后停止捕获。
- 3) **使用 SMTP 命令与邮件服务器交互：**在命令行模式，使用 telnet 程序连接到发件服务器，在新窗口中，输入 SMTP 命令与邮件服务器交互。
- 4) **协议分析：**
  - **HTTP：**选择网页请求的 GET 请求以及返回的响应，记录并分析消息头重各个字段的值和功能；
  - **SMTP：**总结 SMTP 的个命令/响应消息的内容与功能，画出通信过程的消息序列图。

## 二、 HTTP 协议分析

### 1. 数据捕获过程：

清除 Cookie 后访问目标 URL（2 次），在 Wireshark 中设置过滤器为 http，获得结果如下：



No.	Time	Source	Destination	Protocol	Length	Info
77	4.908551	10.21.240.25	10.161.29.25	HTTP	637	GET /sxjm/exp/sourse/e10.htm HTTP/1.1
78	4.921178	10.161.29.25	10.21.240.25	HTTP	218	HTTP/1.1 304 Not Modified
86	5.398692	2001:da8:215:8f02:e...	240e:ff:f100:1006::...	HTTP	844	POST /mmtls/000063bd HTTP/1.1
88	5.459490	240e:ff:f100:1006::...	2001:da8:215:8f02:e...	HTTP	619	HTTP/1.1 200 OK

### 2. 请求消息：

1) 右键追踪 HTTP 流，获得具体消息内容如下图：



```
GET /sxjm/exp/sourse/e10.htm HTTP/1.1
Host: 10.161.29.25
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36 Edg/135.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6
Cache-Control: max-age=0
If-Modified-Since: Sat, 15 Sep 2007 07:00:00 GMT
If-None-Match: "09838866f7c71:0"
Upgrade-Insecure-Requests: 1
```

2) 请求行：分为 3 个部分，分别为请求方法（GET）、URL

（/sxjm/exp/sourse/e10.htm）、协议版本（HTTP/1.1）；

3) 消息头字段分析（由于格式问题，此处不使用表格展示）：

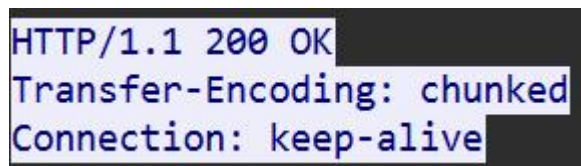
- Host: 10.161.29.25：指定请求的目标主机和端口号；
- User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/135.0.0.0 Safari/537.36 Edg/135.0.0.0：标识发出请求的客户端软件信息（浏览器、操作系统等），此处为浏览器；
- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.7：指定客户端可接受的内容类型（MIME type），当前值表示优先接受网页（HTML、XML）和各种图片格式；
- Accept-Encoding: gzip, deflate：表示客户端支持的内容压缩编码格式，

当前值表示两种主流压缩方式；

- Accept-Language:  
zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6: 指定客户端能接受的语言种类及优先级，当前值表示浏览器偏好语言顺序为简体中文、中文、英文、英国英语、美国英语；
- Cache-Control: max-age=0: 控制缓存行为。当前值表示资源最大允许缓存时间为 0 秒，即客户端不希望使用缓存的内容，而是请求最新的资源；
- If-Modified-Since: Sat, 15 Sep 2007 07:00:00 GMT: 客户端缓存了资源的时间，如果服务器的资源没有在此时间之后更新，就返回 304（未修改）；
- If-None-Match: "09838866f7c71:0": 配合 ETag（实体标签）使用。如果服务器的资源 ETag 值和这个一样，说明未更改，则返回 304；
- Upgrade-Insecure-Requests: 1: 表示客户端愿意将 HTTP 请求升级为 HTTPS 请求。

### 3. 响应消息 (200):

- 1) 此处以请求小图标返回的响应信息为例，消息内容如下：



```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
Connection: keep-alive
```

- 2) 状态行：协议版本 + 状态码 + 描述；
- 3) 消息头：
  - Transfer-Encoding: chunked: 表示服务器使用了“分块传输编码”来返回数据内容；
  - Connection: keep-alive: 指示当前连接不会在本次请求后立即关闭，允许复用（即持久连接）。

### 4. 响应消息 (304):

- 1) 右键追踪 HTTP 流，获得具体消息内容如下图：

```
HTTP/1.1 304 Not Modified
Accept-Ranges: bytes
ETag: "09838866f7c71:0"
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Tue, 08 Apr 2025 05:28:05 GMT
```

- 2) 状态行：协议版本 + 状态码 + 描述；
- 3) 消息头：
  - Accept-Ranges: bytes：表明服务器支持“范围请求”，当前值说明服务器支持按字节范围分段获取文件；
  - ETag: "09838866f7c71:0"：实体标签（Entity Tag），是服务器生成的资源唯一标识符，用于缓存管理，当前值是这个资源的版本标识符；
  - Server: Microsoft-IIS/10.0：指出响应来自哪个 Web 服务器软件，当前值表示服务器使用的是 Microsoft-IIS；
  - X-Powered-By: ASP.NET：指明后端技术栈，当前值表示使用了 ASP.NET 框架来处理请求；
  - Date: Tue, 08 Apr 2025 05:28:05 GMT：服务器生成响应的时间（UTC 格式）。

### 三、 SMTP 协议分析

#### 1. 数据捕获过程

- 1) 配置客户端（本次实验使用 Foxmail）用户的发件服务器不选择 SSL、端口为 35，如下：



The screenshot shows the 'Outgoing Mail Server' configuration window in Foxmail. The 'Server' field is set to 'smtp.qq.com'. The 'SSL' checkbox is unchecked. The 'Port' field is set to '25'. Below these fields, there is a line of text that is partially obscured: '如果服务器支持，启用 STARTTLS 加密传输'.

- 2) 使用客户端发送邮件，待发送完成后停止捕获，获得如下消息：



- LOGIN: 用户名 + 密码 (Base64 编码);
  - PLAIN: 用户名 + 空格 + 密码, 明文形式;
  - XOAUTH 和 XOAUTH2: OAuth 授权机制 (更安全)。
- 250-AUTH=LOGIN: AUTH 机制的另一个声明形式
  - 250-MAILCOMPRESS: 表示支持邮件压缩传输;
  - 250-SMTPUTF8: 表示支持 UTF-8 编码的发件人、收件人地址和邮件内容;
  - 250 8BITMIME: 结束行, 同时表示支持 8 位 MIME 编码的邮件体。
- 4) (Client) "AUTH XOAUTH2
- dXNlcj0xOTgyNjA3NDMwQHFXLmNvbQFhdXR0PUJlYXJlciBBQVRKSik0RU  
 FBQUJBQUFBQUFEUXYrZjd0NkNZZU4rVmtYLzBaeUFBQUFCTk1SMk5OUX  
 JjVENmOFFUV2NjTWg1cEtZeJdpSVdTR3czUWIHMlNnK25EREJSWmVkeG  
 U3R2FIU2VSM0VpM2VPbDZYQU1kdzROR2MvUDRnQkQxMno0NWtld1Jt  
 VDVyazE3Zlp5TEFmYzlwOVU0TGJXaFU2ejNDYXByczZWaWg1SXljMzdBV  
 E93Q2VqTWxqWk1rbzc3ODBUblI2QTJUdURyelpaeXQzQW01MzcvQ0Fpb  
 EwwVlFXynZHSXNLbmQzNWpmTwEB": 客户端发起 XOAUTH2 登录, AUTH  
 XOAUTH2 表示使用 OAuth2 授权机制进行身份验证, 后面的 Base64 编码  
 字符串解码后是用户凭证;
- 5) (Server) "334
- eyJzdGF0dXMiOiI0MDAiLCJtc2ciOiJsb2dpbiBmYWlsliwic2NoZW1lcyl6IkJIY  
 XJlcilnNjb3BlIjoiaHR0cHM6Ly9tYWlsLnFmNvbS8ifQ==": 服务器返回认  
 证结果, 334 状态码表示“继续交互”, 后续 Base64 编码字符串解码后表示  
 认证失败。
- 6) 认证成功后的消息一般为“235 2.7.0 Authentication successful”, 其中 235  
 为状态码表示认证成功。

### 3. Telnet 远程连接, 使用 SMTP 命令与邮件服务器进行交互:

- 1) 首先安装 telnet, Windows 在控制面板中找到程序, 启用 Telnet Client;
- 2) 使用命令“telnet smtp.126.com 25”连接;
- 3) 进行如下交互 (仅解释发送内容, 服务器返回响应解释同上):

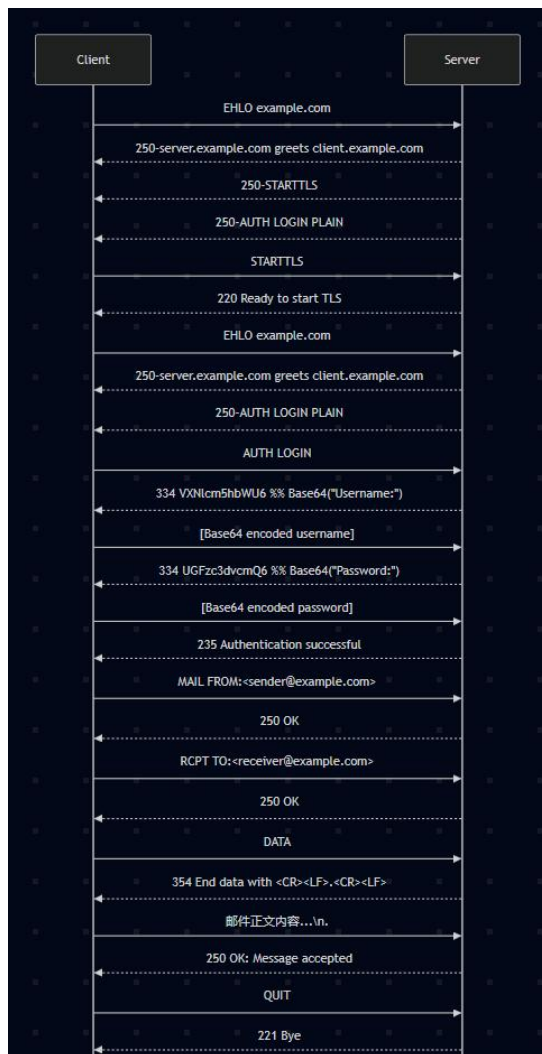
```

220 newxmesmtplогicsvrszb21-0.qq.com XMail Esmtp QQ Mail Server.
EHLO ddd
502 Invalid input from 106.39.130.31 to newxmesmtplогicsvrszb21-0.qq.com
EHLO ddd
250-newxmesmtplогicsvrszb21-0.qq.com
250-PIPELINING
250-SIZE 73400320
250-STARTTLS
250-AUTH LOGIN PLAIN XOAUTH XOAUTH2
250-AUTH=LOGIN
250-MAILCOMPRESS
250-SMTPUTF8
250 8BITMIME
AUTH LOGIN
502 Invalid input from 106.39.130.31 to newxmesmtplогicsvrszb21-0.qq.com
AUTH LOGIN
334 VXNlcm5hbWU6
U21vb3RoV2F0ZXI=
334 UGFzc3dvcmQ6
cWN1aWVsZGx0amN5YmhkaQ==

```

- EHLO: 必须发送的第一条命令;
- AUTH LOGIN: Client 向 Server 说明身份认证方式, 该命令要求所有信息都必须使用 Base64 编码;
- 后续进行交互即可。

#### 4. 消息序列图:



#### 四、 实验结论与心得

在 HTTP 协议分析实验中，我明确了请求/响应消息头的字段功能（如 Content-Type、User-Agent 等）；在 SMTP 协议分析实验中，我通过命令（如 HELO 等）和状态码（如 250 OK、354 Start mail input）的解析，掌握了邮件传输的全流程，并成功绘制了消息序列图。

本次实验通过捕获和分析 HTTP 与 SMTP 协议的实际通信过程，我深入理解了两种协议的功能及交互机制。