

文件传输系统实验报告

林鹏 2019E8017761010

一、 实验目的

在当下的大数据和人工智能时代，数据的重要性已经愈来愈重要。但是文件作为数据在计算机系统中的重要载体，在网络传输的过程中容易被监听嗅探，导致隐私泄露问题，因此文件传输安全的保障也是数据安全中重要的一部分。

本实验我们将学习 socket 编程，从传输层开始动手设计一个简单的文件安全传输系统。本实验涉及到简单的应用层协议设计，socket C/S 程序实现，SSL 安全传输协议，多线程技术，以及网络通信流量的嗅探与分析。

二、 完成的工作

1) 证书的签发

- ✓ 使用 OpenSSL 生成 CA 根证书
- ✓ 使用 OpenSSL 生成服务器私钥及待签名证书，并使用 CA 根证书对其签名
- ✓ 使用 OpenSSL 生成客户端私钥及待签名证书，并使用 CA 根证书对其签名

2) 自定义传输协议

- ✓ 客户端包头
- ✓ 服务端包头

3) 文件传输系统客户端

- ✓ 用户注册、登录
- ✓ 文件浏览、下载
- ✓ 文件上传
- ✓ 可选 使用/不使用 SSL 加密传输
- ✓ GUI 图像界面

4) 文件传输系统服务端

- ✓ 响应用户注册、登录
- ✓ 响应用户上传、下载
- ✓ 多线程
- ✓ 记录用户操作日志

5) Wireshark 抓包分析

- ✓ SSL 加密
- ✓ 不使用 SSL 加密

三、 程序运行环境

注：本系统服务器没有部署在云端，需在本地启动执行

操作系统：Win10

编程语言：Python3.6

依赖 Python 库：tkinter、pymysql、ssl、socket 等（大多均为 Python 内置）

mysql 中应先建立数据库：filetransfer，新建表 user，含有三个字段：id、username、password

启动方法：

1) 启动服务器：

python server_ssl.py

python server_no_ssl.py

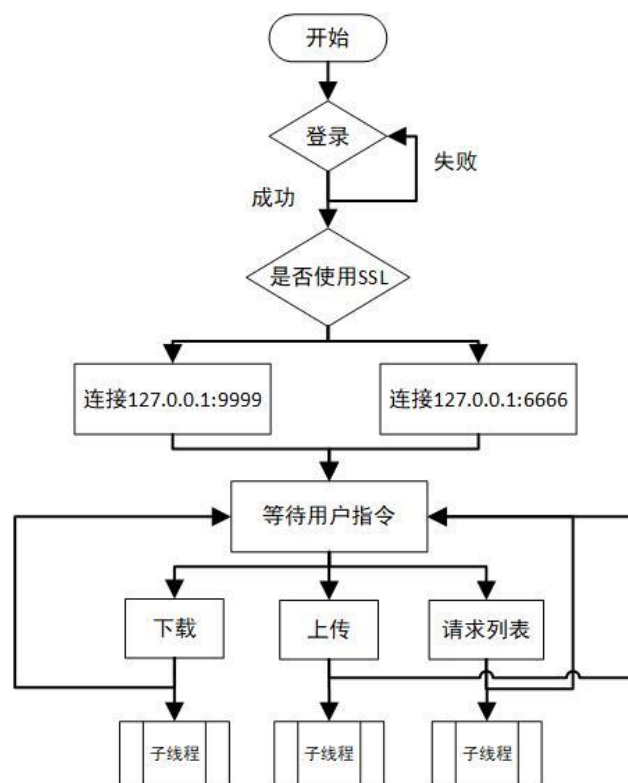
2) 启动客户端：

python main.py

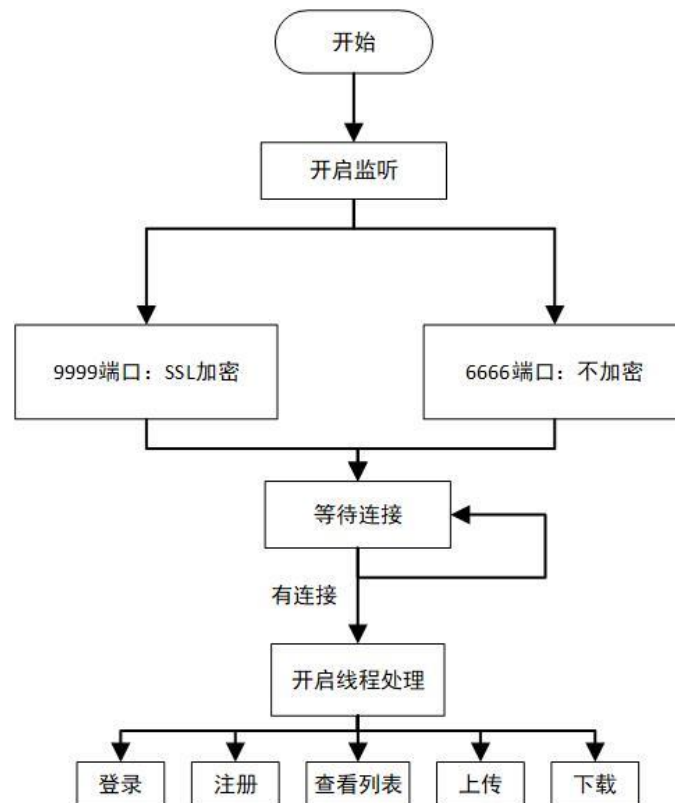
文件夹说明请参考 README.md 文件

四、 系统框架

1) 客户端流程图



2) 服务器流程图



3) 自定义传输协议

在服务器与客户端的每次交流都添加了自定义报头, 客户端主动向服务器请求数据, 服务器被动返回, 因此两者的数据包头会不同。本系统使用了 python 的 struct 结构体实现了报头二进制流的传输。

客户端的报头内容如下: (1024 字节)

Command 包括: Download、Upload、Update、Login 和 Register

Filename 在 download 指令下是要下载的文件名, 在 upload 模式下是本地上传文件的路径。

Filesize 是文件的大小

Time 是数据请求的时间

User 和 password 是用户名和密码, 每次请求数据都会验证一次, 模拟 Cookie 模式。

```
header = {  
    'Command': 'Download',  
    'fileName': filename,  
    'fileSize': '',
```

```
        'time': time.strftime("%Y-%m-%d %H:%M:%S", time.localtime()),
        'user': self.username,
        'password': self.password,
    }
```

服务器的报头内容如下：(128 字节)

由于服务器是被动地回复客户端，所以报头内容不需要太多，故使用 128 字节。

Feedback 指示要回应的指令

Stat 指示响应的状态（如注册、登录等）

Filesize 是文件的大小

User 是当前用户

```
header = {
    'Feedback': 'Login',
    'stat': 'Success',
    'fileSize': os.stat(listResult).st_size,
    'user': username
}
```


五、 效果展示

1) 登录界面

可选择使用或不使用 SSL 加密



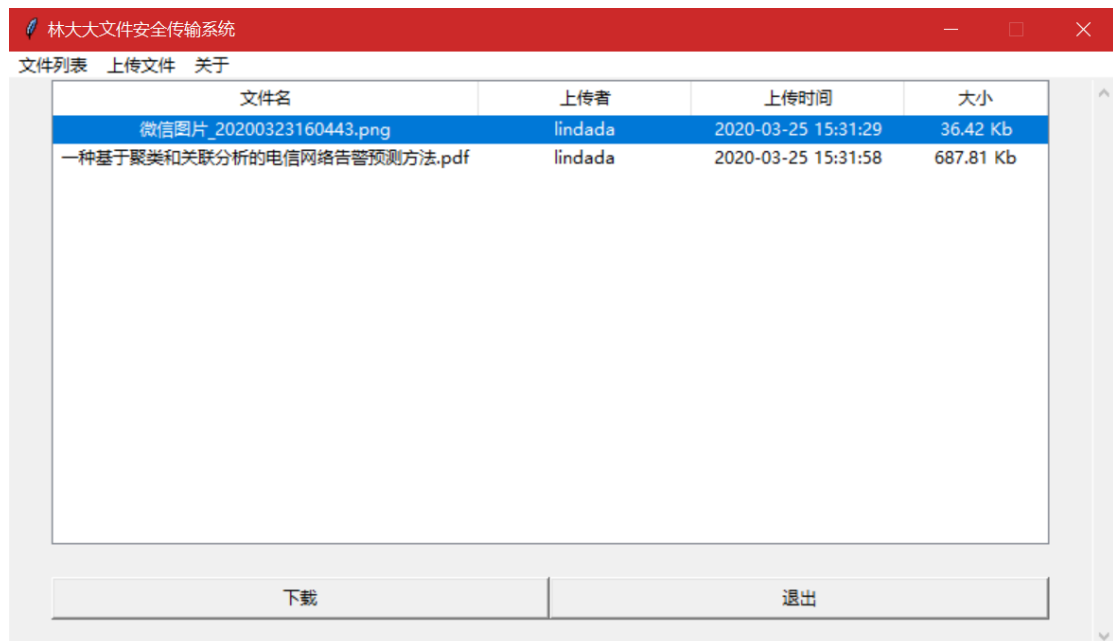
2) 注册界面



The registration window is titled "林大大文件安全传输系统" (Lin University Large File Secure Transfer System) and "注册" (Registration). It contains three input fields for "用户名:" (Username), "请输入密码:" (Please enter password), and "请再次输入密码:" (Please enter password again). A "确认注册" (Confirm Registration) button is located at the bottom.

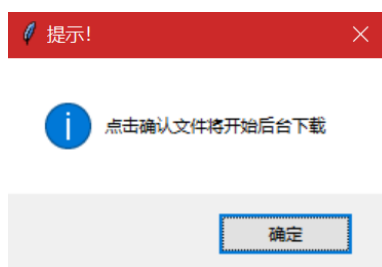
3) 下载

下载路径为项目文件夹下的 ClientDownload，下载采用多线程，点击确认后后台下载，不影响当前页面操作。



The file list window is titled "林大大文件安全传输系统" (Lin University Large File Secure Transfer System) and "文件列表" (File List). It displays a table of files with columns: 文件名 (File Name), 上传者 (Uploader), 上传时间 (Upload Time), and 大小 (Size). The table contains two files: "微信图片_20200323160443.png" and "一种基于聚类和相关分析的电信网络告警预测方法.pdf". Below the table are "下载" (Download) and "退出" (Exit) buttons.

文件名	上传者	上传时间	大小
微信图片_20200323160443.png	lindada	2020-03-25 15:31:29	36.42 Kb
一种基于聚类和相关分析的电信网络告警预测方法.pdf	lindada	2020-03-25 15:31:58	687.81 Kb



4) 上传

上传同样采用多线程，不会影响当前界面的操作。上传的文件路径是项目目录下的 ServerRec 文件夹。



5) 服务器日志

日志保存在项目目录下的 Serverlog.txt，记录了用户的登录、注册、上传、下载操作及具体的时间和操作的状态，如图：



六、 抓包分析

使用 Wireshark 对应用产生的流量进行抓包分析，结果如下：

1) 不使用 SSL 加密

过滤条件为：ip.addr == 127.0.0.1 and tcp.port == 9999

前三个数据包如下：第一个数据包为客户端（端口 50623）向服务器（端口 9999）发送，其中 SYN 被置位了，说明这是个 TCP 连接请求包。第二个包为服务器向客户端发送，其中 SYN 被置位，ACK 为 1，说明是确认了客户端的连接请求。第三个包为客户端向服务器发送，ACK 为 1。至此，TCP 三次握手建立完成。

31	15.397804	127.0.0.1	127.0.0.1	TCP	56	50623 → 9999 [SYN] Seq=0 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
32	15.397856	127.0.0.1	127.0.0.1	TCP	56	9999 → 50623 [SYN, ACK] Seq=0 Ack=1 Win=65535 Len=0 MSS=65495 WS=256 SACK_PERM=1
33	15.397917	127.0.0.1	127.0.0.1	TCP	44	50623 → 9999 [ACK] Seq=1 Ack=1 Win=2619648 Len=0

从第四个包开始，是服务器与客户端的数据传输包，可以看到，每次发送后都会收到对方的反馈包，ACK 为自己的 Seq+len+1。这是 TCP 的协议决定的，以此实现可靠的传输。这里也能看出，客户端的数据包头为 1024 字节，服务器的数据包头为 128 字节，与我们的协议设置一致。

TCP	1068	50623 → 9999 [PSH, ACK] Seq=1 Ack=1 Win=2619648 Len=1024
TCP	44	9999 → 50623 [ACK] Seq=1 Ack=1025 Win=2619648 Len=0
TCP	172	9999 → 50623 [PSH, ACK] Seq=1 Ack=1025 Win=2619648 Len=128
TCP	44	50623 → 9999 [ACK] Seq=1025 Ack=129 Win=2619648 Len=0
TCP	1068	9999 → 50623 [PSH, ACK] Seq=129 Ack=1025 Win=2619648 Len=1024
TCP	44	50623 → 9999 [ACK] Seq=1025 Ack=1153 Win=2618624 Len=0
TCP	621	9999 → 50623 [PSH, ACK] Seq=1153 Ack=1025 Win=2619648 Len=577
TCP	44	50623 → 9999 [ACK] Seq=1025 Ack=1730 Win=2617856 Len=0

点开第四个包的内容（客户端登录请求的数据包头），可以看到信息都以明文的方式传输，非常不安全。如果被人嗅探，则很容易泄露账号及密码。

0010	7f 00 00 01	7f 00 00 01	c5 bf 27 0f 5c e4 34 69 \.4i
0020	52 a6 47 93 50 18 27 f9	8f 36 00 00 7b 22 43 6f	R·G·P·'·'·6·{"Co	
0030	6d 6d 61 6e 64 22 3a 20	22 4c 6f 67 69 6e 22 2c	mmand": "Login"	
0040	20 22 66 69 6c 65 4e 61	6d 65 22 3a 20 22 22 2c	"fileNa me": "",	
0050	20 22 66 69 6c 65 53 69	7a 65 22 3a 20 22 22 2c	"fileSi ze": "",	
0060	20 22 74 69 6d 65 22 3a	20 22 32 30 32 30 2d 30	"time": "2020-0	
0070	33 2d 31 39 20 32 31 3a	31 38 3a 30 36 22 2c 20	3-19 21: 18:06".	
0080	22 75 73 65 72 22 3a 20	22 6c 69 6e 64 61 64 61	"user": "lindada	
0090	22 2c 20 22 70 61 73 73	77 6f 72 64 22 3a 20 22	"pass word": "	
00a0	6c 69 6e 64 61 64 61 22	7d 00 00 00 00 00 00 00	lindada"}.....	

2) 使用 SSL 加密

前三个包依然是 TCP 三次握手，在握手完成之后客户端跟服务器开始了 SSL 加密协议商定过程，可以看到双方互相交换了证书、Key 等信息。

TCP	56	50531 → 9999	[SYN]	Seq=0	Win=65535	Len=0	MSS=65495	WS=256	SACK_PERM=1
TCP	56	9999 → 50531	[SYN, ACK]	Seq=0	Ack=1	Win=65535	Len=0	MSS=65495	WS=256
TCP	44	50531 → 9999	[ACK]	Seq=1	Ack=1	Win=2619648	Len=0		
TLSv1...	561	Client Hello							
TCP	44	9999 → 50531	[ACK]	Seq=1	Ack=518	Win=2619648	Len=0		
TLSv1...	2348	Server Hello, Certificate, Server Key Exchange, Certificate Request, Ser...							
TCP	44	50531 → 9999	[ACK]	Seq=518	Ack=2305	Win=2617344	Len=0		
TLSv1...	2283	Certificate, Client Key Exchange, Certificate Verify, Change Cipher Spec...							
TCP	44	9999 → 50531	[ACK]	Seq=2305	Ack=2757	Win=2617344	Len=0		
TLSv1...	1166	New Session Ticket, Change Cipher Spec, Encrypted Handshake Message							
TCP	44	50531 → 9999	[ACK]	Seq=2757	Ack=3427	Win=2616320	Len=0		

点开一个 Application Data 包，发现内容都是乱码，说明已经被加密，嗅探者无法破译内容。证明我们的 SSL 加密是成功的。

0000	02 00 00 00 45 00	04 45	f8 9a 40 00 80 06 00 00E.E..@.....
0010	7f 00 00 01 7f 00 00 01	c5 63 27 0f 44 09 88 39c'.D.9	
0020	12 a3 3a 2a 50 18 27 ec	73 2a 00 00 17 03 03 04	..: *P.' s*.....	
0030	18 95 82 dc 40 70 bc 95	48 28 89 60 0a 75 57 43	...@p..H(.`uWC	
0040	9d 5e c4 75 cf 53 45 0d	a7 57 ff 94 37 20 6d f4	..^u.SE..W..7 m.	
0050	b4 24 bd 9d 1f a5 9a ac	c8 ee 25 30 00 50 e8 b2	.\$.....%0.P..	
0060	68 21 90 f1 9d df e2 35	4b 3c 4b 13 cc 63 b3 81	h!.....5 K<K..c..	
0070	e7 7f d3 c2 db 3e b8 7e	f0 69 7d 22 78 d3 9d c2>~.i}"x...	
0080	4c e9 69 c8 3e 60 d2 04	9c 38 32 02 20 9f f3 63	L.i.>`..82..c	
0090	f4 47 d4 d4 73 36 df e4	06 51 80 e0 f0 ad 03 ee	.G.s6..Q.....	

七、实验收获与体会

这次的实验让我学到了很多，让我对 SSL 加密传输和 TCP 协议有了更深的了解。以前都是在书上看到的 TCP 协议知识，这次通过自己抓包，验证了 TCP 的三次握手、ACK 号等等，让我对计算机网络知识更加感兴趣了。此外，这次实验自己设计协议，自己申请 SSL 证书、实现 GUI 等都是以前没有试过的，完成之后成就感非常大，对相关领域的理解又加深了一点。

此次实验遇到的问题主要有 GUI 的设计、SSL 证书的申请等。因为事前不是很了解，所以开发时比较难处理，后来经过大量查阅资料、逛博客、学习别人代码，终于把困难一一解决，成功实现了所有功能。这次的实验锻炼了我思考和解决问题的能力，也增加了自己的经验，非常充实。