目录

[一、课程设计目的 1](#_Toc44879911)

[二、需求分析 2](#_Toc44879912)

[1. 产品介绍 2](#_Toc44879913)

[2. 功能要求 3](#_Toc44879914)

[3. 经济和环境要求 5](#_Toc44879915)

[4. 性能要求 5](#_Toc44879916)

[5. 可靠性要求 6](#_Toc44879917)

[6. 安全保密要求 6](#_Toc44879918)

[7. 数据库要求 7](#_Toc44879919)

[8. 外部接口 7](#_Toc44879920)

[三、详细设计 8](#_Toc44879921)

[1. 系统结构说明 8](#_Toc44879922)

[2. 重要数据的说明 10](#_Toc44879923)

[3. 程序函数清单 10](#_Toc44879924)

[四、系统设计难点与亮点 22](#_Toc44879925)

[五、设计成果 23](#_Toc44879926)

[六、设计心得 24](#_Toc44879927)

[七、团队分工说明 24](#_Toc44879928)

## 一、课程设计目的

本次课程设计的主题是“文件安全传输”，预期实现一个文件能够安全传输共享的产品，在架构上采用C/S和P2P的混合架构来实现服务器与用户、用户之间的文件传输和用户个人信息处理等功能，安全特性的实现取决于我们通过对称密码体制DES对文件内容进行加解密，使用非对称密码体制RSA对DES的密钥进行加密来实现公开信道中的安全传输。

除去产品本身的预期和设计，我们进行本次课程实践的目的还包括其他方面。通过对网络层面需求的设计和编码，巩固计算机网络课程的相关知识的掌握，深入了解C/S、P2P架构的特点和HTTP协议的具体内容，强化编写代码的能力；设计非对称密码RSA、对称密码DES完成传输过程的安全需求，可以及时复习本学期现代密码学的理论知识，并且体会课程知识转化为实际应用的过程。

在本次的大型程序设计实践中，已经掌握的编程语言在限定的变成目标面前体现出一定的劣势，所以本次课程设计，我们选择了特定的语言来实现特定的目标，主要是自学python，熟悉相关语法和特性，并且尝试编写程序。与此同时，设计独具一格的UI界面，从而体现产品独特性，也是我们本次课程设计的目标。

## 二、需求分析

软件系统的需求，与日常生活中的需求有着一定的差别，在这里它通常被分为三个方面：业务需求、用户需求、功能需求，有时也会包含非功能需求。本次课程设计实现文件安全传输，用户范围暂定为有一定软件使用基础的人群，比如大学生和计算机相关行业的工作者，对于软件的部分功能的简单描述可以顺利使用相关功能。

首先给出软件的综合描述，试图从总体架构上给出整个软件的轮廓。同时对功能需求、数据要求、性能需求、外部接口、质量属性进行了详细的描述。便于用户、开发人员进行理解和交流，反映出用户问题的结构，可以作为软件开发工作的基础和依据以及确认测试和验收的依据。

### 1. 产品介绍

如今随着通信技术的迅速发展，甚至5G技术也已经逐步在全国进行普及，同时科技逐渐服务生活，数据超速增长，人与人之间，工作同事之间的文件传送变得越来越频繁，与之相关的产品也如同雨后春笋涌现出来。但是由于网络环境的公开、裸露，人们所传输信息的价值与日俱增，安全也就成为了人们关注的话题。

我们设计该款产品，旨在实现用户间、或者用户与服务器间的安全文件传输。它基于C/S架构实现服务器和用户两种角色分类，然后在用户之间通过P2P架构来设计，分别使用Http协议和socket连接来实现通信。对称密码DES对传输文件内容进行加解密，并且使用非对称密码RSA实现对称密码密钥的安全传输。服务器可以接受用户的注册、登录、查看好友列表状态和好友公钥查询的请求，并且实现这些功能。

### 2. 功能要求

* 提供简洁、易懂的人机交互界面。

该功能要求软件开发者实现下述的几个要求：在同一用户界面中，所有的菜单选择、命令输入、数据显示和其他功能应保持分割的一致性；对所有可能造成损害的动作，坚持要求用户确认。对用户出错采取宽容的态度；人机界面应该能对用户的决定做出及时的响应，最大可能减少击键次数、缩短鼠标移动距离；人机界面应提供上下文敏感的求助系统；合理划分并高效使用显示屏；保证信息的显示方式和数据输入方式的协调一致。

* 用户注册及服务器检验。

功能分别要求client端和server端完成相应操作。初次使用本产品时，在用户端，产品提供人机交互界面，UI，来索取用户的相关个人信息，并且生成唯一的用户ID、密码和公私密钥。在此期间，要求服务器能够进行邮箱格式的检验和数据库的检索查重。

* 用户登录及服务器响应。

功能要求用户能够通过向客户端的UI界面填写个人ID和相关个人唯一信息，向服务器发送登录请求，然后服务器处理请求，在已开启的数据库检索，并且校验，然后响应客户端，实现登陆成功或拒绝。

* 通讯录功能、用户在线状态查询及好友在线列表分发

服务器处理用户间的好友请求，建立其好友关系，提供好友间文件传输必备的公钥，也就是建立通讯录。服务器整理所有用户的在线状态，并且对用户提供好友在线状态的列表。

* 用户间文件传输、压缩和加密

服务器提供接收方的公钥，用户间建立连接。通过对称密码DES对传输文件内容进行加解密，并且使用非对称密码RSA实现对称密码密钥的安全传输。在传送之前对所有文件进行压缩，接收以后先进行解压。

#### 软件功能优先级

|  |  |
| --- | --- |
| 功能要求 | 优先级排序 |
| 提供简洁、易懂的人机交互界面 | 3 |
| 用户注册及服务器检验 | ２ |
| 用户登录及服务器响应 | ２ |
| 通讯录功能、用户在线状态查询及好友在线列表分发 | ３ |
| 用户间文件传输、压缩和加密 | １ |

#### 功能关系示意图

如下:

服务器

用户A

用户B

用户注册及服务器检验

用户登录及服务器响应

通讯录功能、用户在线状态查询及好友在线列表分发

用户间文件传输、压缩和加密

UI界面

主要以三部分和两种角色来描述软件的功能关系。

### 3. 经济和环境要求

该产品经济方面需要三名开发人员的三台电脑,时间要求七天。

开发环境要求:python version:3.7或3.8

要求包含库:socket struct json os threading time zipfile glob des CBC PAD\_PKCS5 binascii random string rsa pymssql re tkinter pickle

### 4. 性能要求

该部分对于产品的性能进行需求分析,主要分为下述几个方面。

#### 响应要求

为了能够快捷地提供文件传输和隐私保护服务,系统应该快速地对数据进行加密解密和密钥处理,并实现对数据库的高速访问。

无论是客户端和服务器端,当用户已经登录,进行任何操作时,软件应该及时的进行反应,反应时间在5秒以内。系统能够检测出各种非正常情况,比如说用户端和服务器端的通信中断,无法访问数据库等等,从而避免出现长时间等待甚至是无响应。

#### 容量要求

本次课程设计得到的软件,由于是文件传输类的产品,所以对于容量的要求较高。对于用户和用户之间文件传输的容量要求为4GB。

#### 用户数要求

软件采用了C/S架构,对于用户容量限定为300人。

### 5. 可靠性要求

系统应该保证开启后24H内不宕机,同时登录人数保持为40人时不出现异常,系统能够正常运行,并且正确提示相关内容。

### 6. 安全保密要求

系统有着严格的用户注册和登录的认证检验功能,同时对非好友之间不提供传输文件的功能,好友之间达到一定程度的隐私保护要求。对于公开信道传输的文件经过对称密码体制DES加密,密钥由非对称密码体制RSA来加密传输。

### 7. 数据库要求

1. 能够保证数据的独立性。数据和程序相互独立有利于加快软件开发速度，节省开发费用。
2. 冗余数据少，数据共享程度高。
3. 系统的用户接口简单，用户容易掌握，使用方便。
4. 能够确保系统运行可靠，出现故障时能迅速排除；能够保护数据不受非受权者访问或破坏；能够防止错误数据的产生，一旦产生也能及时发现。
5. 有重新组织数据的能力，能改变数据的存储结构或数据存储位置，以适应用户操作特性的变化，改善由于频繁插入、删除操作造成的数据组织零乱和时空性能变坏的状况。
6. 具有可修改性和可扩充性。
7. 能够充分描述数据间的内在联系。

### 8. 外部接口

该部分表明软件和外部设备或者用户之间的交互与连接设计。

#### 用户界面

用户界面是程序中用户能看见并与之交互作用的部分,设计一个好的用户界面是非常重要的,本设计将为用户提供美观,大方,直观,操作简单的用户界面。使用python进行UI设计。

## 三、详细设计

软件设计的基本目标是用比较抽象概括的方式确定目标系统如何完成预定的任务，即软件设计是确定系统的物理模型。

### 1. 系统结构说明

#### 模块分类

在客户端程序中，有着以下模块：

* 非对称加密模块：RSA

RSA是目前使用最广泛的公钥密码体制之一。算法的安全性基于RSA问题的困难性，也就是基于大整数因子分解的困难性上。RSA算法的保密强度随其密钥的长度增加而增强。但是，密钥越长，其加解密所耗用的时间也越长。因此，要根据所保护信息的敏感程度与攻击者破解所要花费的代价值不值得以及系统所要求的反应时间来综合考虑。由于进行的都是大数计算，使得RSA最快的情况也比DES慢上好几倍，无论是软件还是硬件实现。速度一直是RSA的缺陷。一般来说只用于少量数据加密。RSA的速度比对应同样安全级别的对称密码算法要慢1000倍左右。

算法主要包含密钥对生成，公钥加密，私钥解密三个部分。

该模块也就是基于RSA公钥密码体制来设计的。

* 对称加密模块：DES

DES算法属于对称密码体制，它相对于RSA的特点在于加密密钥和解密密钥是相同的。它的有效密钥长度为56位，剩余几位为奇偶校验位来补足64位，分组长度为64位，也就是说对数据进行加解密的单位是64位，明文和密文的长度相同。它的主要组成部分为密钥生成算法、加密算法和解密算法。

* 压缩模块：ZIP

该模块对特定路径下的文件进行压缩，包含压缩和解压两个过程。并且该模块采用ZIP格式。

* P2PServer模块

P2P指的是“Peer to peer”，是一种网络架构模式，在这个模块中，包含着P2P被动方上传和P2P被动方接受两个函数。

* P2PClient模块

与上一个名为P2PServer的模块相对应，在这个模块中包含着相对应的两个函数，分别为：P2P主动方下载和P2P主动方发送。

* C/S：Client2Center线程
* P2P：Server线程

在服务器端程序中，有着以下模块：

* 注册模块

该模块的主要功能可以通过姓名来轻松得到，它实现用户按照客户端的提示，填写要求的相关信息，然后向服务器端发送请求，服务器进行检验以后，再返回客户端请求的过程。

* 查询模块

这个模块包含了大多数的查询操作，它可以通过调用来查询与用户相关的许多信息。其中包括：查询用户数量、查询用户ID、查询用户名、查询所有好友状态和查询自己用户的在线状态。

* 登录模块

这是一个专注于用户登录系统，从而使用系统功能的模块。登录过程其实包括着用户和服务器之间的通信过程，它首先是用户在客户端提供的UI界面填写个人信息，然后客户端打包信息，传输给服务器，然后服务器在数据库中进行用户是否存在、ID和密码填写是否正确的校验。并且根据校验结果，对用户的客户端程序做出相应。

* 好友处理模块

这一个模块其实可以被称为“通信录管理模块”，本次的课程设计之中加入了好友的元素，我们可以通过本模块的引用，来实现好友的添加、删除。

* 下线模块

顾名思义，这是一个处理如何在本系统优雅而且正确地下线的模块。

### 2. 重要数据的说明

在“ client.py文件中

* client = threading.Thread(target=c2s)中的client数据结构，表示客户端线程
* server = threading.Thread(target=p2p)中的server数据结构，表示服务端线程，这两个数据结构是多线程开发的体现。

在“server.py”文件中

* sql数据结构，是以字符串形式存储的sql命令

### 3. 程序函数清单

下面会将“client.py”文件和“server.py”文件的程序函数做出整理，并且进行一定发研究。

#### 函数名——所在文件名

在“client.py”文件中，程序函数如下：

|  |  |
| --- | --- |
| 获取本地IP | socket.gethostname() |
|  | socket.gethostbyname(hostname) |
| RSA密钥对生成 | create\_keys(ID) |
|  | rsa.newkeys(1024) |
|  | pubkey.save\_pkcs1() |
|  | privkey.save\_pkcs1() |
| RSA公钥加密 | encrypt(text, PK) |
|  | rsa.PublicKey.load\_pkcs1(PK) |
|  | text.encode('utf8') |
|  | rsa.encrypt(original\_text, pubkey) |
| RSA私钥解密 | decrypt(crypt\_text, ID) |
|  | privatefile.read() |
|  | rsa.PrivateKey.load\_pkcs1(p) |
|  | rsa.decrypt(crypt\_text, privkey).decode() |
| DES密钥生成 | newKey() |
|  | ''.join(random.sample(string.ascii\_letters + string.digits, 8)) |
| DES文件加密 | desenc\_file(filename, KEY) |
|  | des\_encrypt(a, KEY) |
| DES文件解密 | desdec\_file(filename, KEY) |
|  | des\_descrypt(a, KEY) |
| 字符串加密 | des\_encrypt(s, KEY) |
|  | des(secret\_key, CBC, iv, pad=None, padmode=PAD\_PKCS5) |
|  | k.encrypt(s, padmode=PAD\_PKCS5) |
|  | binascii.b2a\_hex(en) |
| 字符串解密 | des\_descrypt(s, KEY) |
|  | des(secret\_key, CBC, iv, pad=None, padmode=PAD\_PKCS5) |
|  | k.decrypt(binascii.a2b\_hex(s), padmode=PAD\_PKCS5) |
| 压缩 | zip(filename) |
|  | zipfile.ZipFile(target, 'w', zipfile.ZIP\_DEFLATED) |
| 解压缩 | un\_zip(zip\_path) |
|  | glob.glob(zip\_path) |
|  | zipfile.ZipFile(dir\_zip, 'r') |
|  | f.extract(file, path=unzip\_file\_path) |
| P2P被动方上传 | sget(cmds, conn) |
|  | json.dumps(header\_dic) |
|  | header\_json.encode('utf-8') |
| P2P被动方接收 | sput(pc, ID) |
|  | pc.recv(4) |
|  | struct.unpack('i', obj)[0] |
|  | un\_zip('%s/%s' % (share\_dir, file\_name)) |
|  | decrypt(m, ID) |
|  | desdec\_file(file\_name, dk) |
| P2P主动方下载 | cget(pc) |
|  | pc.recv(4) |
|  | struct.unpack('i', obj)[0] |
|  | pc.recv(header\_size) |
|  | header\_bytes.decode('utf-8') |
|  | json.loads(header\_json) |
|  | header\_dic['file\_size'] |
|  | header\_dic['filename'] |
| P2P主动方发送 | cput(cmds, conn, PK) |
|  | newKey() |
|  | desenc\_file(filename, dk) |
|  | encrypt(dk, PK) |
|  | zip(filename) |
|  | json.dumps(header\_dic) |
|  | header\_json.encode('utf-8') |
| Client2Center线程 | c2s() |
|  | socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) |
|  | pc.connect(('10.38.28.219', 8080)) |
|  | pc.recv(1024).decode('utf-8').strip() |
|  | pc.recv(1024).strip().decode('utf-8') |
|  | time.sleep(0.2) |
| P2P：Server线程 | p2p() |
|  | socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) |
|  | phone.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1) |
|  | phone.bind((hipaddr, 6000)) |
|  | phone.listen(5) |

在“server.py”文件中，程序函数如下：

|  |  |
| --- | --- |
| 连接数据库 | pymssql.connect('(local)', 'sa', 'weimo\_siji', 'FSTS') |
|  | conn.cursor() |
| 查询用户数量 | numberofUsers() |
|  | cur.execute(sql) |
|  | cur.fetchone() |
| 加入新用户 | db\_insert\_user(username, password, email, pk) |
|  | numberofUsers() |
|  | cur.execute(sql) |
|  | conn.commit() |
| 查询用户ID | db\_IDsearch(email) |
|  | cur.execute(sql) |
|  | cur.fetchone() |
| 查询用户名 | db\_usernamesearch(ID) |
|  | cur.execute(sql) |
|  | cur.fetchone() |
| 验证密码 | db\_check\_password(ID, pw\_input) |
|  | cur.execute(sql) |
|  | cur.fetchone() |
|  | pw\_in\_db[0].strip() |
| 新用户注册 | user\_register(socket, email) |
|  | numberofUsers() |
|  | socket.send(("ID:%d" % newID).encode("utf-8")) |
|  | time.sleep(0.1) |
|  | socket.send("Your username：".encode("utf-8")) |
|  | socket.recv(1024).decode("utf-8").strip() |
|  | db\_insert\_user(username, password, useremail, PK) |
| 添加好友 | befriend(ID1, ID2) |
|  | cur.execute(sql)  cur.fetchone() |
|  | conn.commit() |
| 解除好友 | deletefriend(ID1, ID2) |
|  | cur.execute(sql) |
| 查看所有好友状态 | checkfriends(socket, ID) |
| 发起传输请求 | fileTransportRequest(ID1, ID2) |
| 验证邮箱格式 | validateEmail(email) |
| 登录 | login(socket, IP, port) |
|  | db\_IDsearch(email) |
|  | user\_register(socket, email) |
|  | db\_usernamesearch(ID) |
| 在线状态检查 | ifonline(ID) |
| 下线操作 | offline(ID) |
| 服务器主程序 | servermian(newSocket: socket.socket, addr) |
|  | str(addr).split() |
|  | login(newSocket, IP, port) |

#### 函数功能

|  |  |
| --- | --- |
| 函数名称 | 功能 |
| socket.gethostname() | Socket  获取本地主机名 |
| socket.gethostbyname(hostname) | 获取IP列表和Host别名列表 |
| create\_keys(ID) | 生成RSA公钥和私钥 |
| rsa.newkeys(1024) | 生成公钥和私钥 |
| pubkey.save\_pkcs1() | 提取公钥 |
| privkey.save\_pkcs1() | 提取私钥 |
| encrypt(text, PK) | 使用RSA的公钥对文本加密 |
| rsa.PublicKey.load\_pkcs1(PK) | 公钥提取 |
| text.encode('utf8') | 公钥加密 |
| rsa.encrypt(original\_text, pubkey) | RSA加密 |
| decrypt(crypt\_text, ID) | 私钥解密 |
| privatefile.read() | 私钥文件读取 |
| rsa.PrivateKey.load\_pkcs1(p) | 提取私钥 |
| rsa.decrypt(crypt\_text, privkey).decode() | 解密 |
| newKey() | DES密钥生成 |
| ''.join(random.sample(string.ascii\_letters + string.digits, 8)) | 密钥合成 |
| desenc\_file(filename, KEY) | DES文件加密 |
| des\_encrypt(a, KEY) | 同上 |
| des(secret\_key, CBC, iv, pad=None, padmode=PAD\_PKCS5) | 字符串加密 |
| k.encrypt(s, padmode=PAD\_PKCS5) | 加密 |
| binascii.b2a\_hex(en) | 转换 |
| des\_descrypt(s, KEY) | 字符串解密 |
| des(secret\_key, CBC, iv, pad=None, padmode=PAD\_PKCS5) | 定义DES |
| k.decrypt(binascii.a2b\_hex(s), padmode=PAD\_PKCS5) | 解密 |
| zip(filename) | 压缩文件 |
| zipfile.ZipFile(target, 'w', zipfile.ZIP\_DEFLATED) | 打开指定压缩文件 |
| un\_zip(zip\_path) | 解压缩文件 |
| glob.glob(zip\_path) | 路径生成 |
| zipfile.ZipFile(dir\_zip, 'r') | 读取压缩文件内容 |
| f.extract(file, path=unzip\_file\_path) | 提取压缩内容到指定文件 |
| sget(cmds, conn) | P2P被动方上传 |
| json.dumps(header\_dic) | 将字典形式的数据转为字符串 |
| header\_json.encode('utf-8') | 编码规则选定 |
| sput(pc, ID) | 被动方接收 |
| pc.recv(4) | 接收数据 |
| struct.unpack('i', obj)[0] | 解包得到一个元组 |
| un\_zip('%s/%s' % (share\_dir, file\_name)) | 文件解压 |
| decrypt(m, ID) | 密钥获取 |
| cget(pc) | P2P主动方下载 |
| json.loads(header\_json) | 将json格式数据转换为字典 |
| cput(cmds, conn, PK) | P2P主动方发送 |
| c2s() | Client2Center线程 |
| p2p() | P2P：Server线程 |
| phone.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1) | 回收重用端口 |

下一部分的函数清单如下：

|  |  |
| --- | --- |
| 函数名称 | 功能 |
| pymssql.connect('(local)', 'sa', 'weimo\_siji', 'FSTS') | 连接 |
| conn.cursor() | 数据库连接操作 |
| numberofUsers() | 获取用户数目 |
| cur.execute(sql) | 数据库执行sql语句 |
| cur.fetchone() | 数据库中获取单条数据 |
| db\_insert\_user(username, password, email, pk) | 数据库中新建用户、存入数据 |
| conn.commit() | 提交sql命令 |
| db\_IDsearch(email) | 在数据库中，根据邮箱地址查找ID |
| db\_check\_password(ID, pw\_input) | 校验用户ID和输入的密码是否匹配 |
| pw\_in\_db[0].strip() | 移除字符串头尾指定字符 |
| user\_register(socket, email) | 实现用户注册 |
| befriend(ID1, ID2) | 相互添加好友 |
| deletefriend(ID1, ID2) | 删除好友 |
| checkfriends(socket, ID) | 查看所有好友的状态 |
| fileTransportRequest(ID1, ID2) | 好友间的文件传输请求发送 |
| validateEmail(email) | 验证邮箱格式是否有效 |
| login(socket, IP, port) | 登录函数 |
| user\_register(socket, email) | 用户注册函数 |
| db\_usernamesearch(ID) | 通过用户ID查询用户姓名 |
| ifonline(ID) | 在线状态检查 |
| offline(ID) | 下线操作 |

#### 参数说明

1. 在RSA的密钥对生成中，为了安全起见，我们选取密钥长度为1024比特。
2. 编码过程中，涉及到编码规则的选取，我们选择UTF-8国际编码标准。

#### 算法描述

由于流程图过于大，所以该部分请查看下一页。

|  |
| --- |
|  |

## 四、系统设计难点与亮点

本系统实现了文件传输以及用户登录注册等功能，主要的设计难点是文件传输的时候客户端（client）与服务器端（server）的交互，重点是利用socket技术进行监听（server），以及两端进行通信交互的命令与条件跳转，由于尚未系统性学习过socket包的使用，在实际调用的时候会出现一些错误，如，OSError: [WinError 10038] 在一个非套接字上尝试了一个操作。而这些错误有的会影响程序运行，有的却在实际完成文件传输后报错，令人不知是喜是忧；另外在文件传输的关键就是IP地址的获取，首先，尝试百度直接搜索IP地址，尝试后失败，查询资料得知是负责本块地区的分管路由器的地址，并不会直接联系到笔记本电脑端；其次，尝试window+R，cmd命令行查询，得到本机的地址，但仍不可以成为通信传输的地址，可以说是本地区网内的地址，类似局域网，跨地区异地仍旧难以实现，最后我们想到了我们日常使用的北邮vpn，这可以使我们在同一“网络”下，以一种便捷的方式实现跨区域的IP与端口连接实现通信；

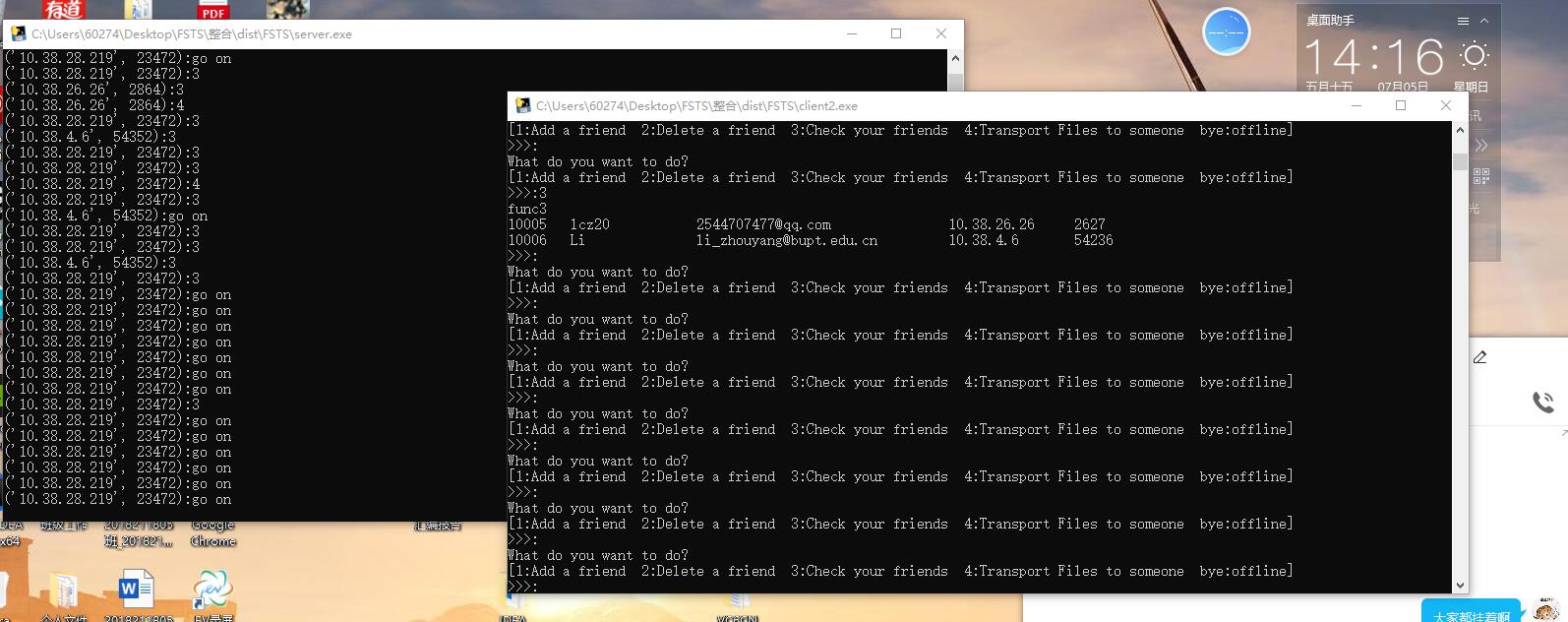
另一个难点就是对python语言掌握的不太充足，比如，实现文件加解密的过程需要导入特定的包，比如crypto，pyDES，pycryptodome等等，由于不同的版本问题所需要的包不太相同，对应的函数在不同的包中也有所差异，这点值得我们更仔细更扎实地研究与学习python语言的使用；

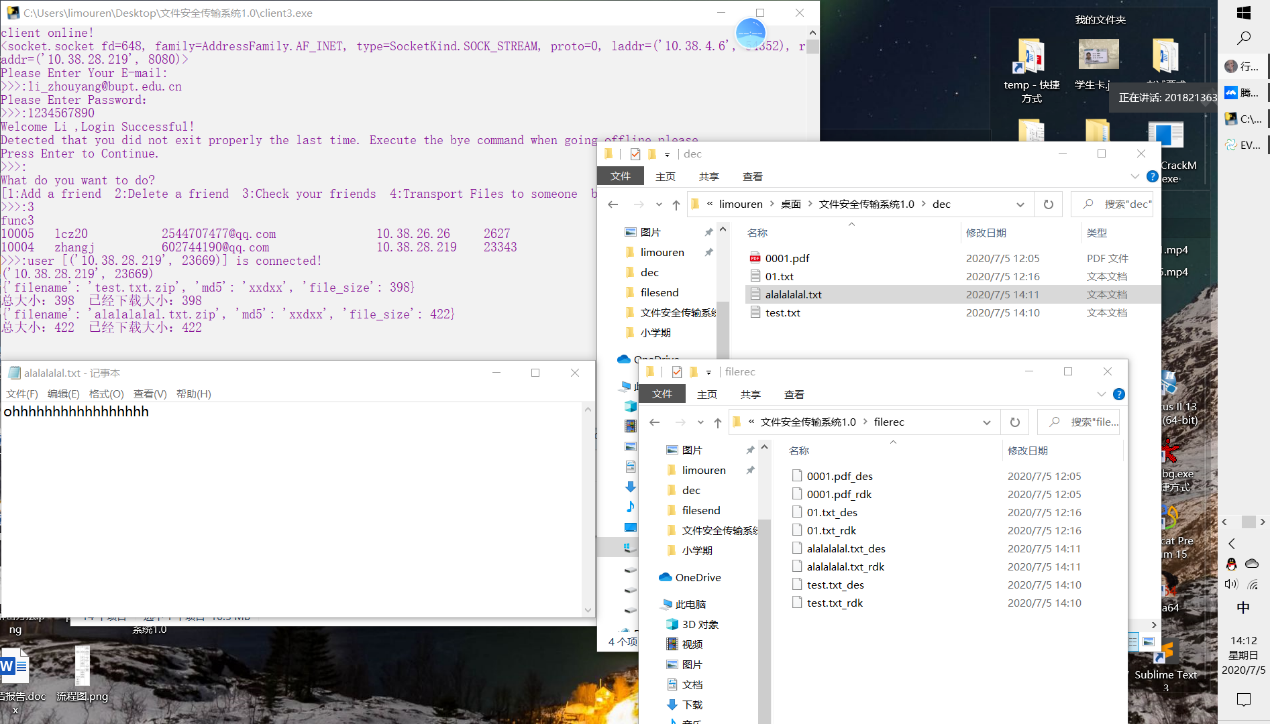
最后，最遗憾的是没有完成GUI界面，分析原因在于认真完成代码后却不知tk包的特点，比如，tk不能连续调用，tk只能存在一个根窗口，其余窗口可以通过Toplevel()实现等等；与其他组交流后得知PyQt5或是wx包均可以完成同样的功能。

本系统的亮点有实现了异地真正的文件传输，以及标准完成了所要求的基本功能。尤其是可以实时查看好友的在线状态，即好友若在线可以显示其IP、端口号与公钥，否则会在对应位置显示None字符；在添加好友的功能中需要提供仅好友自身才知道的ID，避免了单向添加对方未知这一事件的尴尬与风险。特别要说的是，我们的系统同时实现了P2P以及C/S的功能模块，方便了文件传输的选择，正式传输前会显示对方的IP与端口号，确保连接成功以及接收方正确，在传输的时候通过输入简单的指令即可完成传输。

## 五、设计成果

服务器端：（支持多人连接并支持相互传输文件）



客户端：（与服务器端连接交互，实现上传与下载文件）

详细代码请见文件夹附件，谢谢！

## 六、设计心得

最后，终于实现了安全文件传输系统。纵然过程有许多坎坷，但是没有汗水的付出怎么会得来成功运行传输的喜悦呢。通过这次小学期实践课的学习与操作，我们深刻体会到一个软件从一个想法一个要求，到一点点软件需求分析，再到各功能模块的编程，最后汇总连接成一个完整的系统的真实过程。我们选择的是Windows10环境下的Python3.7及以上版本的语言编程，还没接触时得知Python是一个面向对象的功能强大的语言，后面实际操作体会到这三个Python语言的初学者在Python上下的功夫是最多的，调用函数，行缩进对齐的格式要求等等都是需要认真研究克服的难点。今日的完成，只是一个初入大型程序编程的开始，在今后的时间更需要不断扎实掌握的语言，例如，Python等等，多实践多发现实际运用的问题才会对其更加熟悉，了解其特点。

## 七、团队分工说明

|  |  |  |
| --- | --- | --- |
| 罗晨钊 | 张靖（组长） | 李周阳 |
| 系统前端GUI窗口界面的实现，实验报告后四部分的撰写，文件传输系统的测试工作 | 系统后端服务器与客户端的设计，文件压缩及传输加解密的实现，文件传输系统的测试工作，实验报告的制图 | 文件压缩及传输加解密实现，实验报告前三部分的撰写，文件传输系统的测试工作 |