P2P Download Protocol

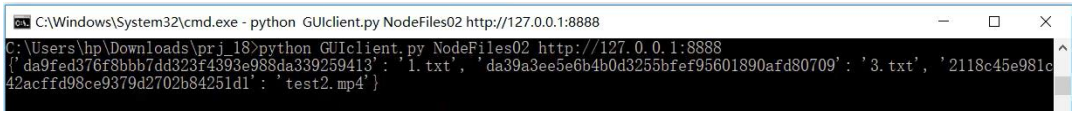
成员	张嘉琪-11611907；罗炎-11611902；蒲雨佳-11612429

Project 环境搭建：

1） 操作系统： Window 10
2） Python 版本： 3.7
3） CPU 服务器：2.60GHZ*2.8-core total
4） 运行 Python 方式：IDE PyCharm

测试场景以及测试结果：

首先，我们在一台主机上打开 CMD，启动节点服务。将共享文件匹配唯一 hash 码。



其次，将文件 hash 码和自己的 IP 地址上传到 tracker。

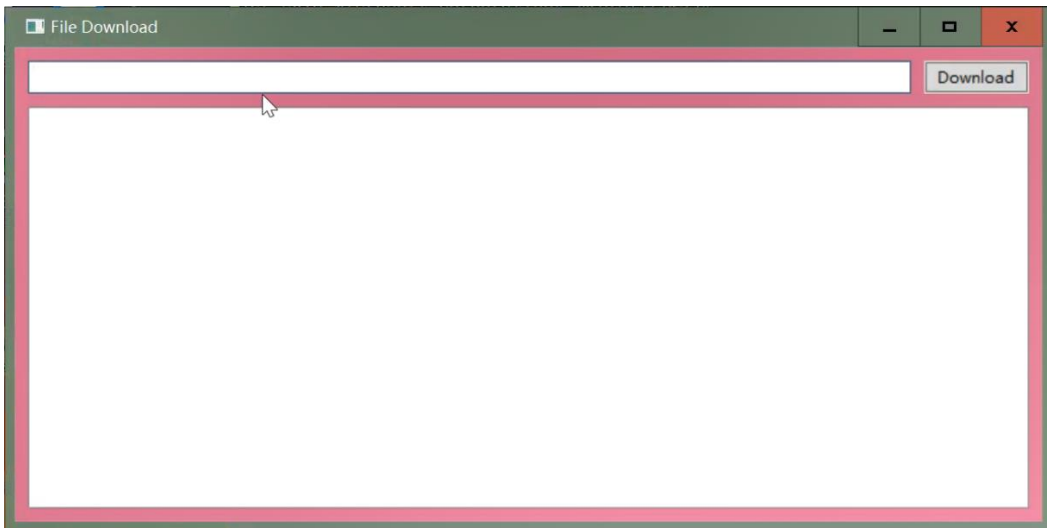
接下来，用户端拿着 hash 码搜索拥有文件的 ip 地址

得到

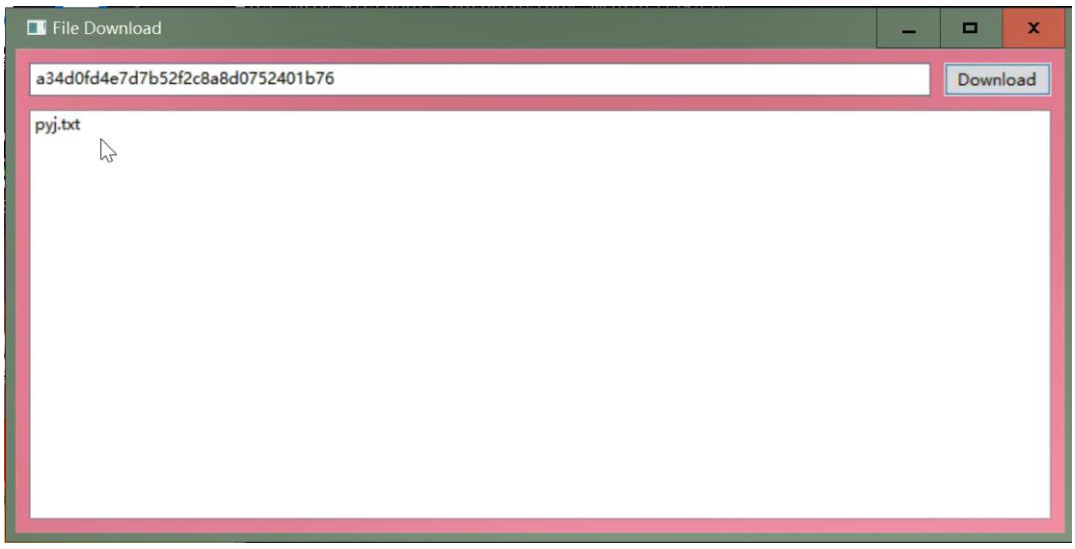
然后，将这个 IP 地址输入代码中

```
class Client(wx.App):
    def __init__(self, dir_name, url):
        self.dir_name = dir_name
        self.secret = random_string(SECRET_LENGTH)
        node = ListableNode(url, dir_name, self.secret)  # 实例化ListableNode节点对象
        thread = Thread(target=node._start)
        thread.setDaemon(1)
        thread.start()
        sleep(HEAD_START)
        self.server = ServerProxy(url)
        url_file = ["http://127.0.0.1:8888"]
        for line in url_file:
            print(line)
            self.server.hello(line.strip())
        super(Client, self).__init__()
```

接着，在用户端打开下载器



然后，在搜索框输入 hash 码，点击下载



显示列表中就出现了我们想要下载传输的文件，pyj.txt。双击这个文件打开，验证我们传输成功了。

代码思路：

P2P 对等网络中，各台计算机无主从之分。一台计算机即可作为客户端从其他计算机获取信息，也可以作为服务器设定共享资源为其他计算机所使用。Peer2 分享一个文件并把这个文件生成的唯一 Hash 码分享给对等方 Peer1. 当 Peer1 要进行资源下载时，拿着 Hash 码向 Tracer 获取拥有此资源的对等方列表。列表包括对等方的 IP 地址。Peer1 拿到 IP 地址向对等方 Peer2 发送文件请求。两台计算机此时都应该启动节点服务，启动节点服务后的计算机才能够处理收到的文件请求，当 Harsh 码验证正确时文件才能够顺利传输。因此两台主机能够顺利传输的步骤是：

1）两台主机在 CMD 命令行中启动节点服务
2）Peer2 将文件匹配生成唯一 Hash 码
3）Peer2 通知 Tracker 自己拥有文件资源并愿意共享
4）Peer1 通过某些途径了解到文件对应 Hash 码
5）Peer1 利用 Hash 码向 Tracker 请求拥有资源的同伴地址
6）Peer1 在 CMD 命令行中打开文件下载器界面
7）输入 Hash 码，验证成功，下载相应的文件

文件下载器的界面代码的思路：

1）导入 wx 包
2）创建应用程序对象
3）创建下载器窗体，定义窗口高度
4）添加下载列表控件
5）设置列表控件内容为文件列表
6）绑定列表点击事件的处理方式
7）更新文件列表

经验总结：

1）这次实验中最大的收获就是大家都切实理解到 P2P 传输的好处和实现思路。但在这次实验中，为了实现 hash 码的问题，我们做了很久的研究。Hash 码是标识文件的手段，而要保证相同的文件共用同一 Hash 码，就要求生成 Hash 码时使用的内容是一样的。一开始，我们希望 Tracker 在用户请求做种时，利用文件名和文件路径为用户提供 Hash 码。然而对于不同的用户来说，同一个内容的文件会由于分享文件夹的路径不同而对应到不同的 Hash 码上。因此，我们在用户启用节点服务时，利用文件的内容

生成 Hash 码来保证文件与其的一对一关系。

2）另外，在实现 Tracker 功能时，我们原先希望利用 python asyncio 对用户发来的包进行解析，再返回 http 包完成和用户之间的交互。然而利用 python socket 来发送用户的消息时发现 asyncio 完成的 Tracker 不作出应答。后来抓包发现，socket 包发送的是 UDP 或 TCP 报文，而 asyncio 只能收到 Application 层协议的报文内容。后来利用 Django 框架实现网页上的交互。

个人总结：

罗炎： 负责 GUI 部分，使用的是 Python 中的 wxPython。学习使用 wx 进行一些基本的 GUI 设计，创建一个完整的功能健全的简单 GUI 用户界面。并负责写报告和 PPT。

张嘉琪：负责 Tracker 和数据传输部分，主要实现了通过 Tacker 做种和请求文件持有用户地址列表两个功能，通过对文件 hash 码对应用户地址列表的字典的维护，使用户可以公开资源并找寻目标下载地址。

蒲雨佳：负责数据传输部分，基于 XML-RPC 远程调用实现文件传输下载。实现上传文件获得唯一 hash 码功能，在服务器判断 hash 码是否匹配。