

网络安全实验报告

题 目 基于 socket 的扫描器设计

专 业 信息安全

学 号 1190202401

学 生 陈豪

指 导 教 师 王彦

# 一、实验目的

熟悉socket 编程，可以利用socket 编程编写基于linux平台的 C/S程序和基于 windows 平台的扫描器。

**二、实验内容**

1. 熟悉 Linux 编程环境
2. 在Windows机器上安装Linux虚拟机
3. 在 Linux 环境下编写 C/S 程序，熟悉 socket 编程。要求客户端和服务器端能够传送指定文件。该程序在后续实验中仍需使用。客户端与服务器端在不同的机器中。
4. 在 Windows 环境下利用 socket 的 connect 函数进行扫描器的设计，要求有界面，界面能够输入扫描的 ip 范围和端口范围，和需使用的线程数，显示结果。
5. 实验课的时候，检验结果和现场截图，为撰写实验报告做准备。

# 三、实验过程

**（一）Linux 环境下的 C/S 程序**

## 实验基本信息：

实验环境：Ubuntu 20.04 x64 编程语言：C

## 1. 需求分析

需要在两台 linux 虚拟机之间传送文件，所以需要给两台 linux 虚拟机都配置一个可以访问的 ip。

程序功能：

(1)客户端：

a.可以向服务端发送一个本目录下指定的文件，文件名由用户输入；

b.可以从服务端下载一个服务端目录下的文件，文件名由用户输入；

(2)服务端：

a.可以监听来自客户端的连接请求；

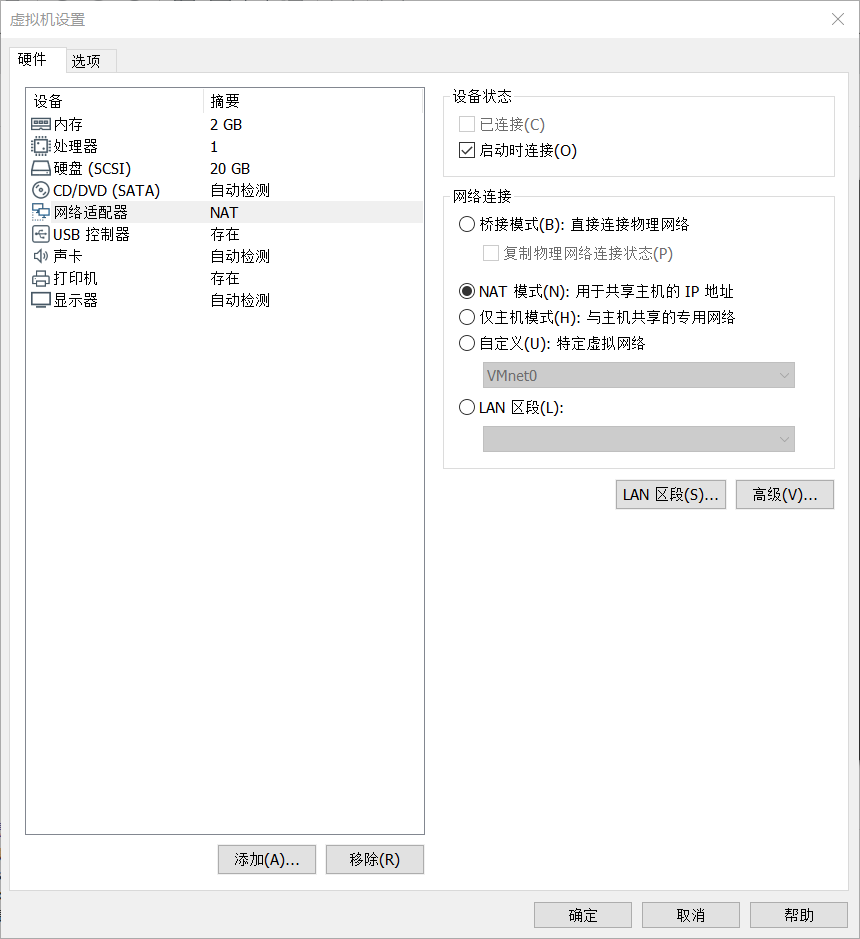
b.可以接收客户端传送的文件；

c.可以向客户端传送一个指定的文件，文件由客户端给出。

(3)传送文件要求：任何二进制文件。

## 2. 环境配置

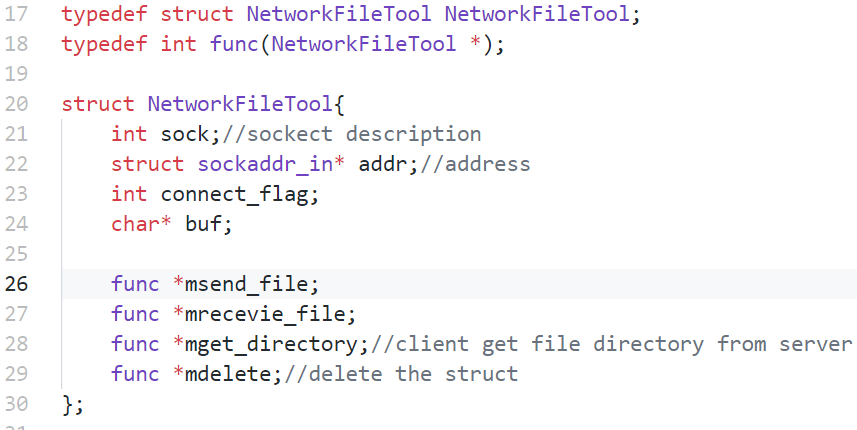
配置虚拟机的网卡即可：



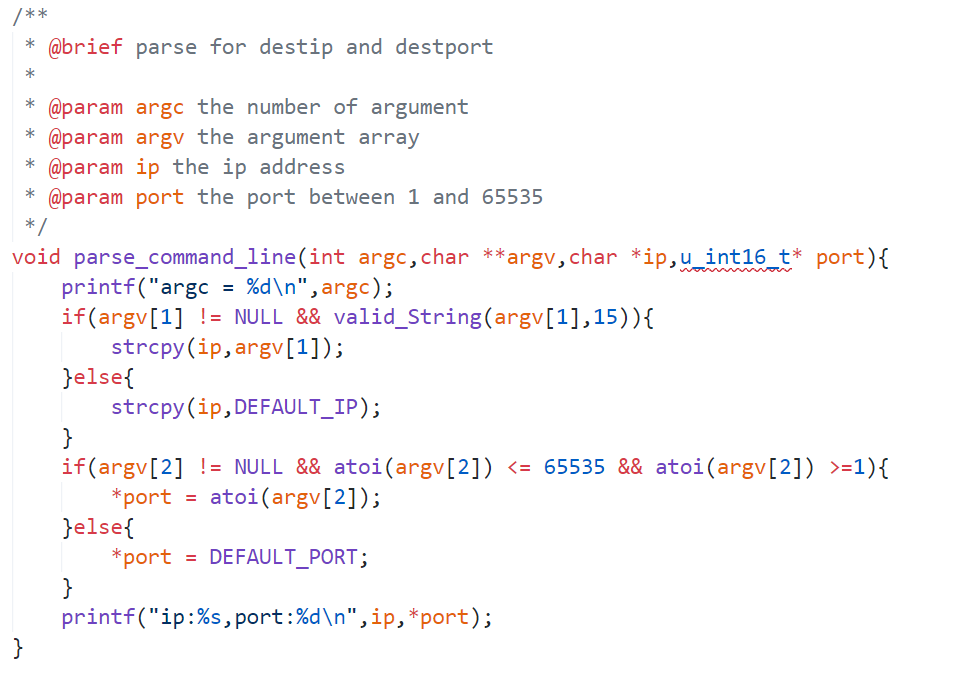


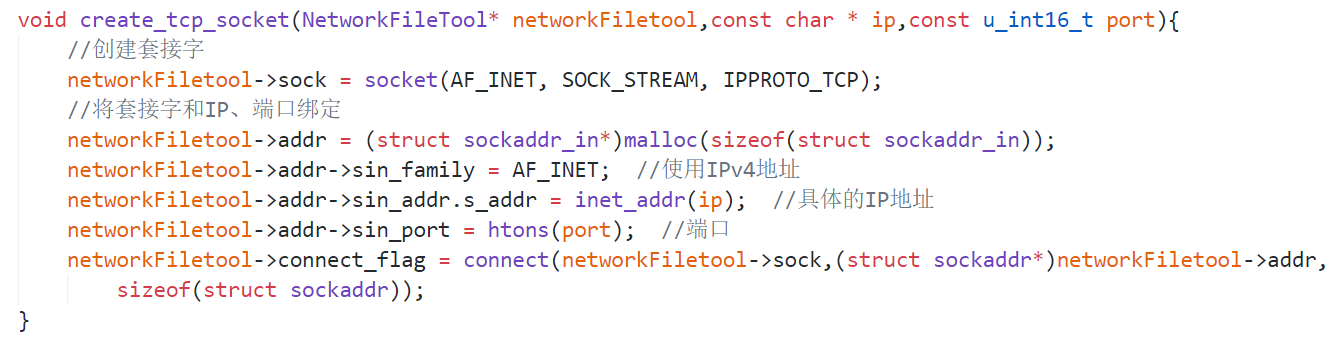
## 3. 客户端编写

新建了一个结构体定义如下：

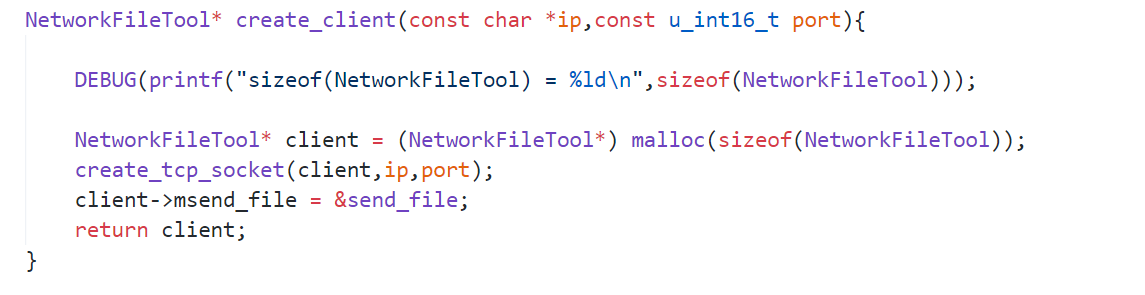


实现中msend\_file用于发文件，剩下的三个函数指针并未实现。





编写了parse\_command\_line和create\_tcp\_socket函数用于从命令行获取ip和端口并创建相应的套接字。



create\_client函数用于创建并初始化结构体。

主要的处理函数如下：



首先从提示选择是下载还是上传，接着要求客户输入文件名。在上传的时候，由于fread函数不会清空缓存区会导致最后一次读取时如果没读完将把上一次读取的内容一起上传，因此这里先使用了fseek和ftell获取文件大小。实际还可以选择每次上传一次后就使用memset清空缓存区。下载同服务器下载相同原理。

## 4. 服务端编写



服务器程序上传方面会让客户端发来文件名字和大小，以方便进行文件的大小校对。

**（二）Windows 环境下的扫描器程序**

## 实验基本信息：

实验环境：Windows10 x64

vscode

编程语言：python

## 1. 需求分析

实验指导中要求编写界面，由于不会使用C语言编写界面，所以使用python加qtpython编写界面。

程序功能：

1. 用户可以输入需要扫描的 ip 范围、端口范围和想使用的线程数，输入框中只能输入合法的字符；
2. 如果用户输入的错误，将不予执行。
3. 当所有输入都正确无误后，按下开始扫描，程序开始扫描用户指定的 ip 和端口；
4. 关于扫描的线程分配：

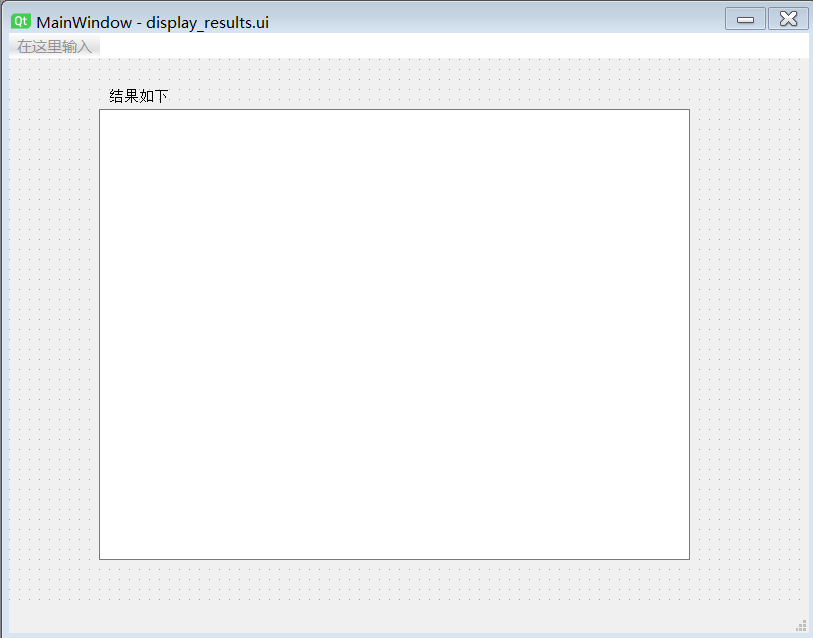
采用的方案是总端口除以总线程数，然后给每个线程分配端口范围。

## 2. 界面编写

界面使用qtpython自带的designer软件只需拖一拖就可以实现实现界面了。



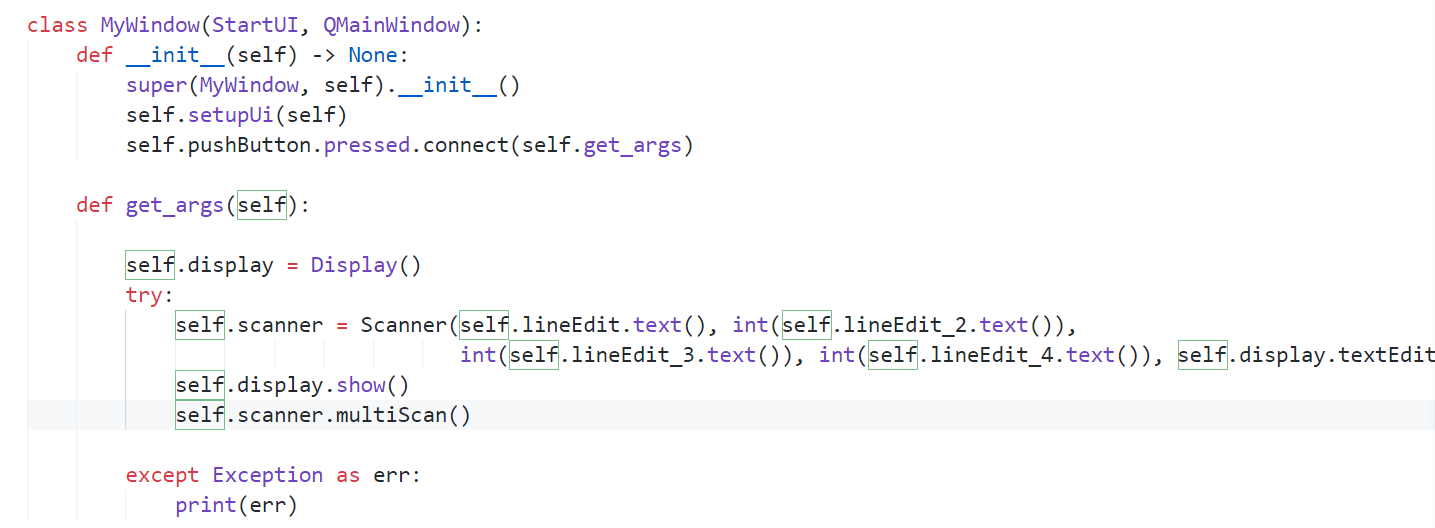
主界面



输出界面

## 3. 控件逻辑编写

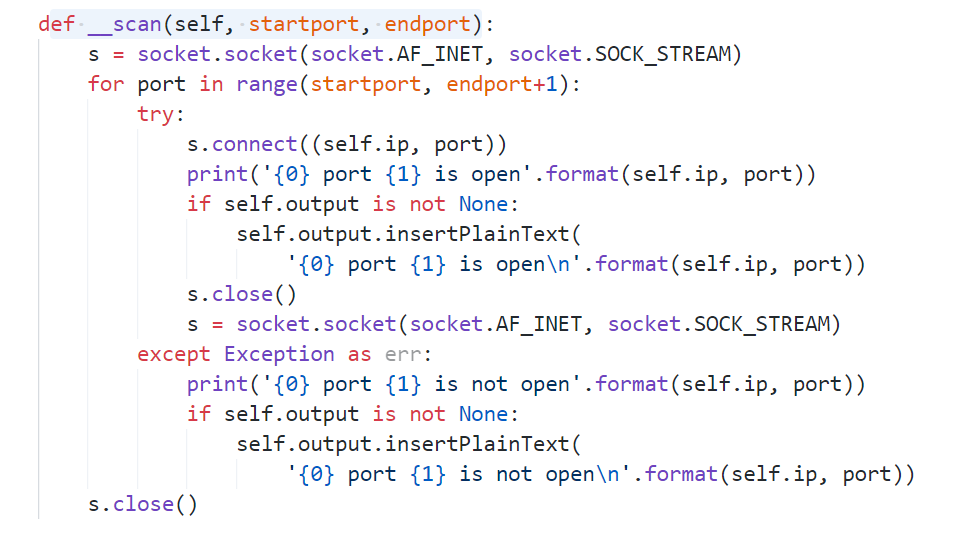
为主界面的开始扫描按钮添加了点击事件



当点击的时候就会创建输出界面并且开始多线程扫描。

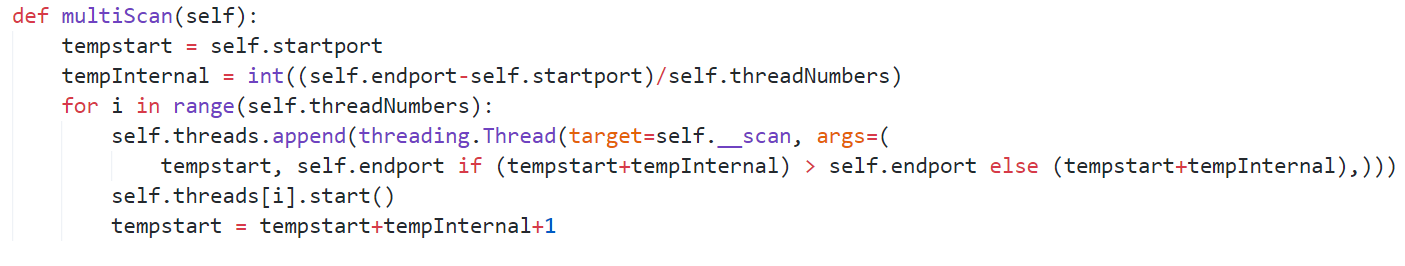
## 4. 具体功能编写

1. 扫描主方法



对一定范围内的端口扫描，如果connect上了就关闭socket并重建一个，并打印xx端口is open否则打印xx端口is not open。

1. 扫描线程方法



根据端口总数除以线程数进行扫描分配。

(3) 输入检查

    def \_\_check\_ip(self):

        compile\_ip = re.compile(

            '^(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|[1-9])\.(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|\d)\.(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|\d)\.(1\d{2}|2[0-4]\d|25[0-5]|[1-9]\d|\d)$')

        if compile\_ip.match(self.ip):

            return True

        else:

            return False

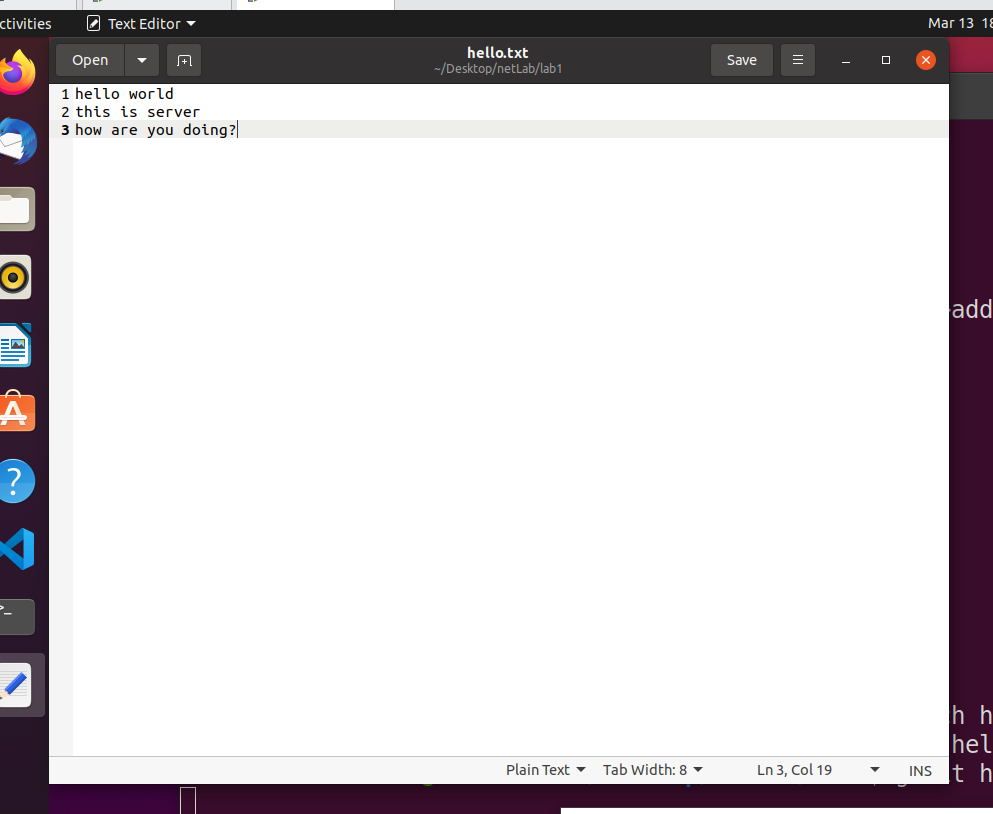
根据正则表达式检查ip，对于端口的话就简单检查是否在1和65535之内。

# 四、实验结果

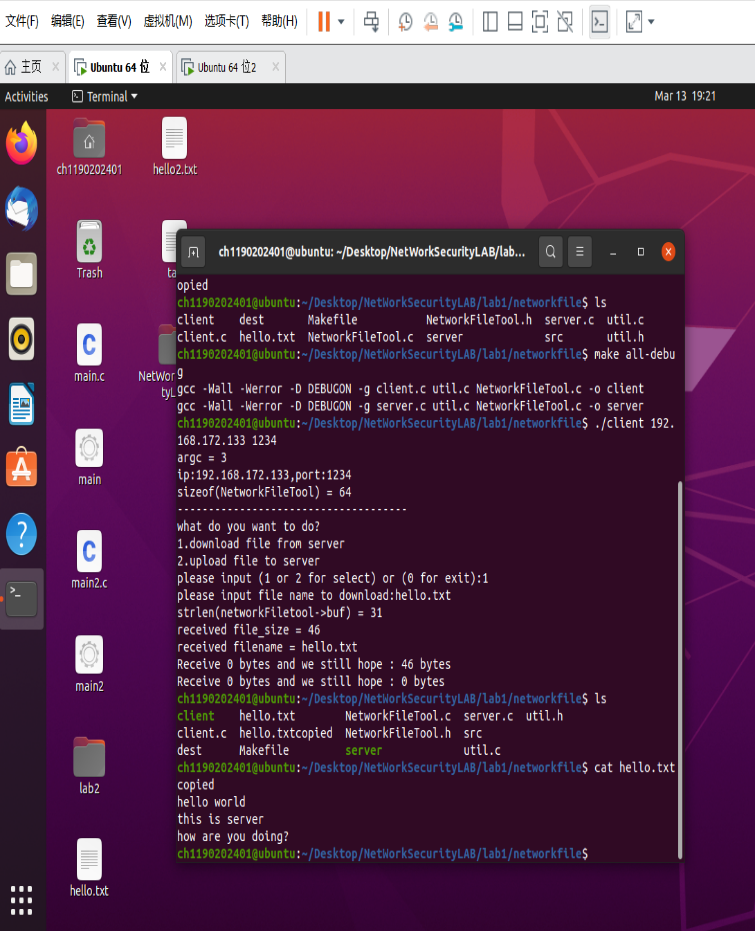
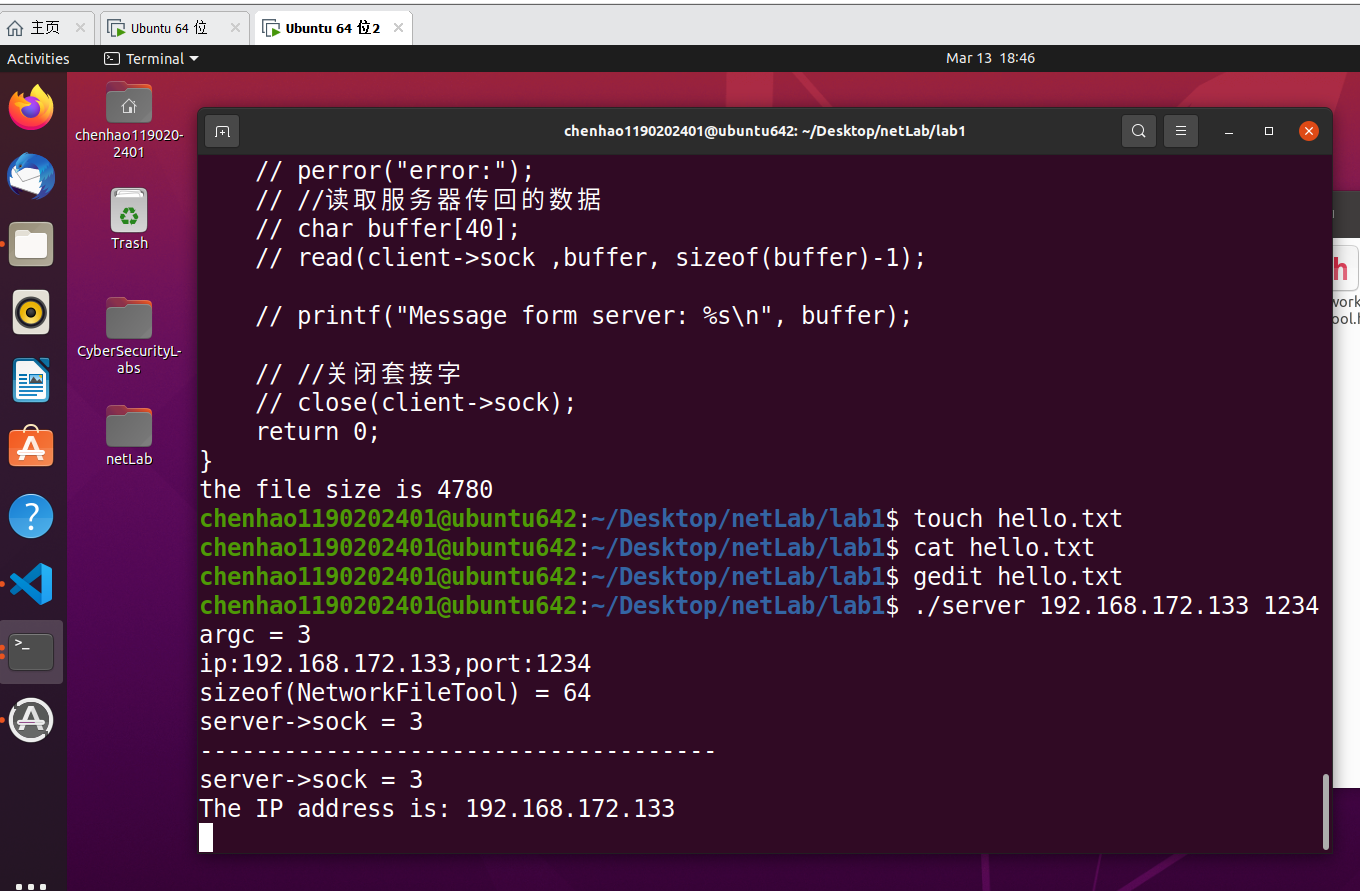
1. **Linux 环境下的 C/S 程序**

首先在测试下载，在服务器端准备文件hello.txt

其中的内容如下：



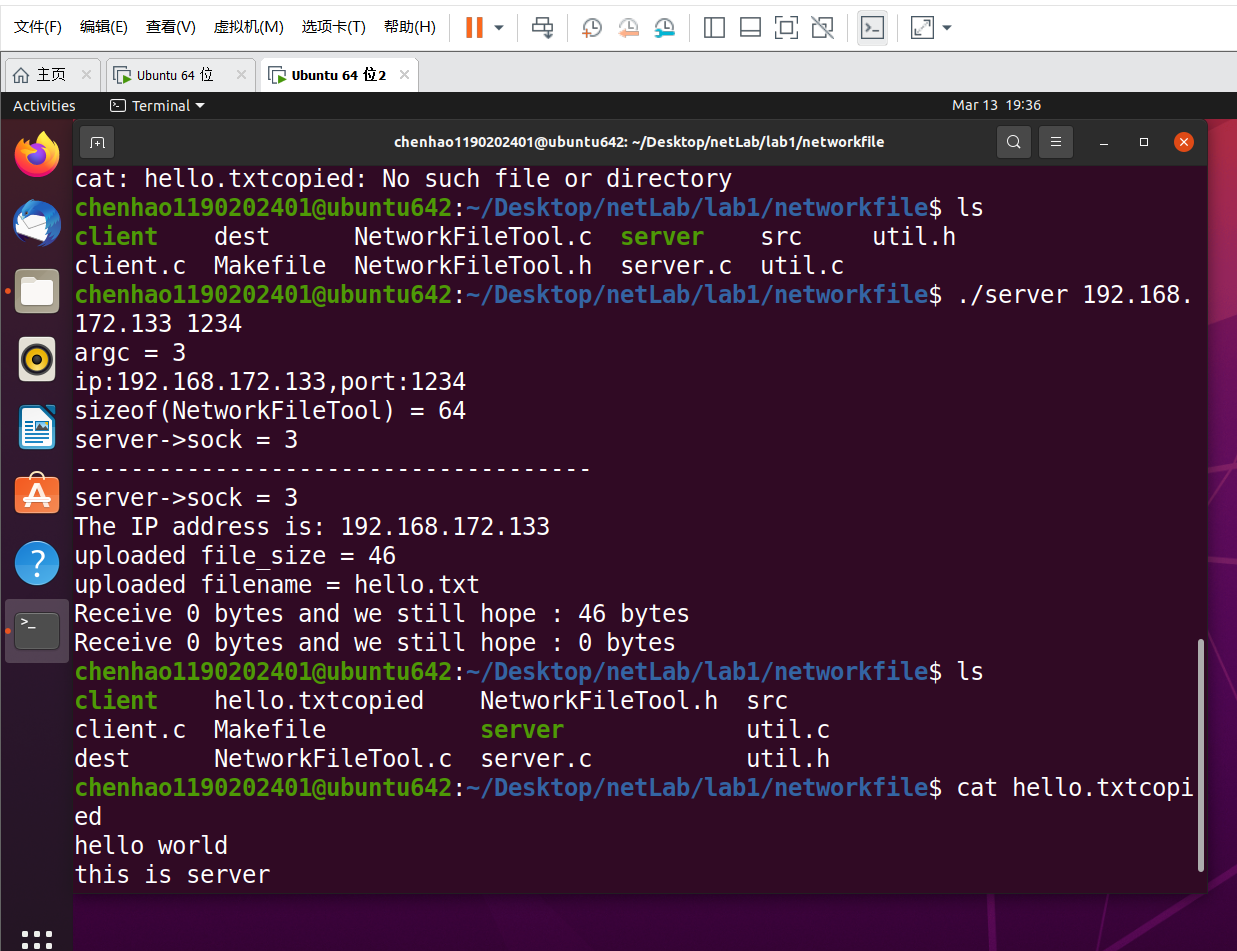
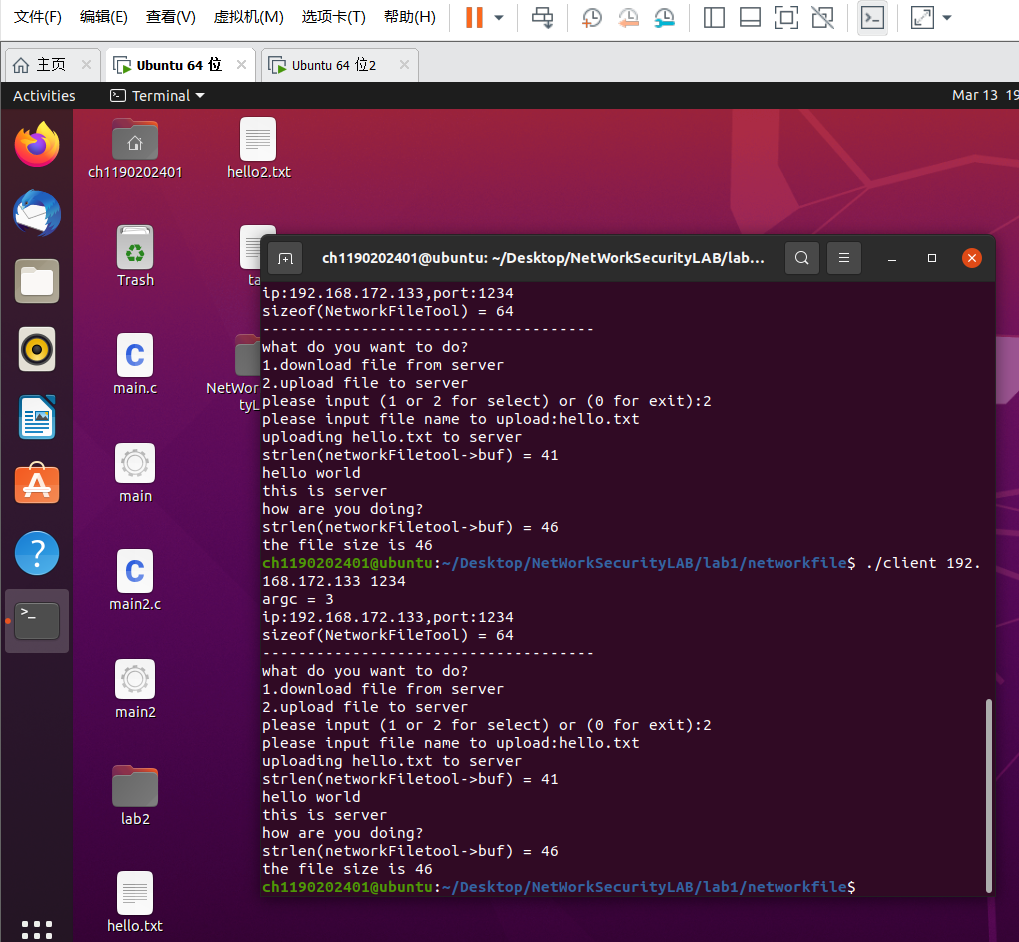
然后开始测试下载



图中左边为服务端，右边为客户端。

接下来测试上传

这一次在客户端上传hello.txt文件，服务器接收并写入到hello.txtcopied文件中



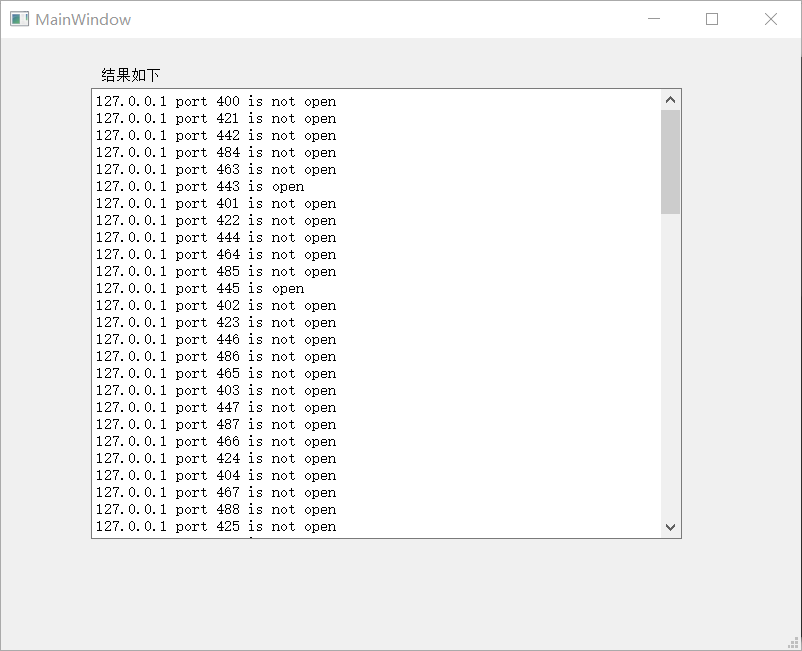
图左为客户端，图右为服务器

1. **Windows 环境下的扫描器程序**

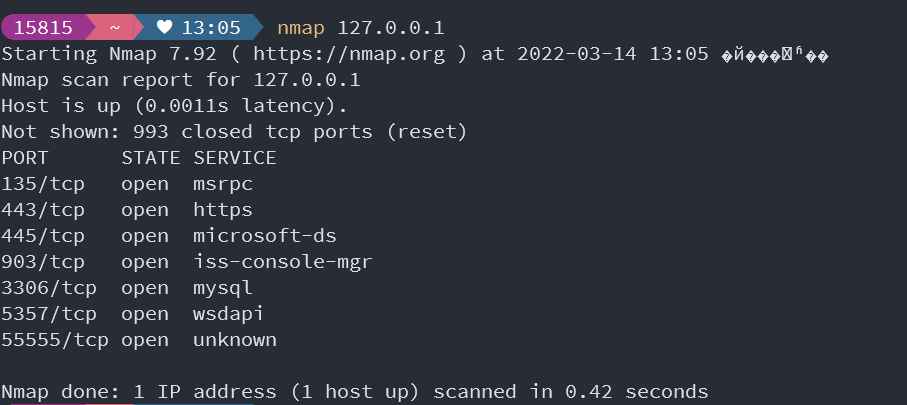
使用该程序扫描本地部分端口



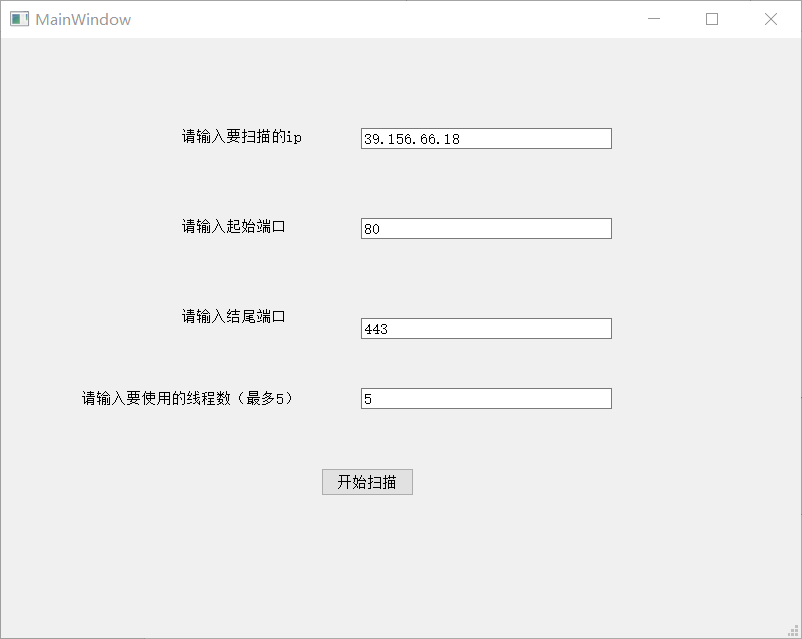
部分结果如下：



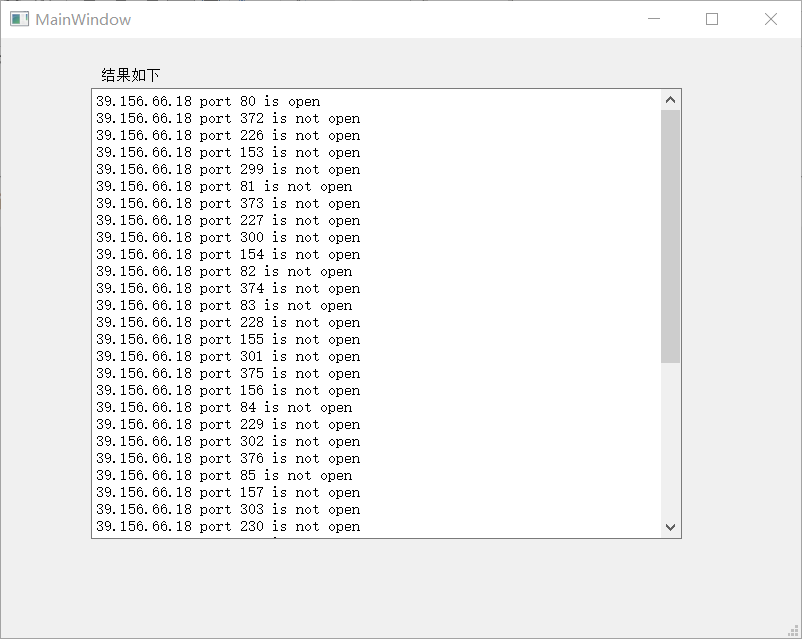
与使用nmap扫描的结果对比如下：



扫描百度端口如下：



部分结果如下：



使用nmap做对比：

