基于Socket通信的远程监控系统

## 系统概况

该系统基于FTP服务器修改而成，使用TCP协议传送命令，使用UDP协议进行文件发送以及视频流的传输。在监控目标机端运行FileServer.java即可对其进行监控。

Server端使用了线程池，最大支持4个客户端对其进行访问。在指定的根目录下可以通过客户端上传下载文件以及创建目录。

Client端部署在监控机上，使用经典Linux命令如：ls、del、mkdir、cd等命令可以对文件进行操作，同时使用screen命令可以得到被监控机器当前的屏幕截图，使用video命令可以得到被监控机器的屏幕视频流。使用send命令可以向被监控端上传客户端根目录下的文件。

## 二、端口信息

Client下载Server端文件使用2020端口，而Client向Server端上传文件使用20200端口，命令传输在2021端口，视频流传输在5001端口。

## 三、使用方法

Server：在目标机上使用javac编译所有的java文件，然后使用以下命令：

*Java FileServer [rootpath]*

即可在目标机运行服务端程序，监听客户端的链接。

Client端的相关参数硬写入在FileClient文件中，可以根据需求更改目标HOST的地址，Client的根目录随参数传入。使用javac编译后使用以下命令运行。

*Java FileClient [rootpath]*

在云服务器上进行测试后发现网络不是很稳定，故本实验采取localhost进行测试。

相关命令：

*ls： 列出当前目录下的所有文件以及文件夹*

*cd [path]： 进入一个目录*

*.. ： 返回上一级目录*

*mkdir [path]： 创建一个目录*

*del [file]： 删除一个文件*

*get [file]： 下载一个文件到本地*

*screen： 获取目标机器当前屏幕截图*

*video： 启动目标机屏幕监控*

## 四、相关说明

FileClient 客户端

FileServer 目标机服务端

Server 屏幕监控程序服务端

Client 屏幕监控程序客户端

DeleteFileUtil 文件删除模块

CaptureScreen 屏幕截图

## 五、工作流程

**文件操作部分**

TCP协议用于文本传输，而UDP用于服务器和客户端之间的文件传输。第一步是在正在运行的服务器上设置PORT侦听。使用2021作为TCP端口，使用2020作为UDP端口，并在启动服务器时创建用于TCP的serverSocket和用于UDP的datagramSocket。客户端还建立一个TCP套接字，该套接字将连接到服务器。 TCP将确保客户端和服务器之间的通信将完全传输。

客户端通过三向握手连接到服务器。服务器使用线程池来处理来自不同客户端的多个连接。建立连接后，服务器将开始为客户端命令提供服务器。

当客户端发送ls命令时，服务器将检查路径是否为目录，如果不是，则服务器将向客户端发送错误消息或将路径发送至将向客户端发送归档的函数。当执行get cd命令时，服务器将首先检查命令是否通过验证，一旦检查完成，服务器将获取参数。如果参数不是..，服务器将检查目录是否存在，如果path存在，则将当前路径更改为新路径，并将新路径返回到主线程。

如果当前路径是根路径，则该函数不会更改当前路径。当获取get命令时，客户端将首先检查命令验证，然后将其发送到服务器。服务器获取参数将检查路径是否存在目录。如果检查通过，则服务器会将文件长度发送给客户端，并准备从客户端获取数据报包。客户端从服务器接收消息并计算数据包号，然后将datagramPacket发送到服务器，以确保服务器可以获取客户端UDP端口。然后客户端和服务器都准备好传输文件。服务器会将文件读取到缓冲区中，然后将缓冲区拆分为一些数据报包。数据报包将在短时间内一一发送，为了控制传输速度，服务器在将数据包发送到客户端后会休眠1毫秒。当计数器等于数据包编号时，客户端将循环接收数据包并中断循环。当服务器收到再见命令时，服务器将中断侦听循环以释放IO流之类的资源并关闭与客户端的套接字连接。

**图片部分**

截图使用了Robot类，使用Robot.createScreenCapture来截取屏幕图片，然后将图片写入到截图文件当中，在FileServer处调用sendFile将图片文件发送回Client端。

**视频流部分**

FileClient向FileServer发送video命令后，Client端首先启动该一个VideoClient监听来自5001端口的socket连接。FileServer接收到video命令后启动VideoServer，于客户端的5001端口建立socket连接，通过Robot类进行截图，然后使用JPEGCode将图片编码为JPEG格式后加入ZipOutputStream进行压缩传输。上述截图过程将会运行在一个循环当中，每运行一次将暂停50毫秒，每秒截图20次保证20FPS的画面帧率。

## 六、任务分工

程序编写：陈方舟（文件操作）、吴东杰（截图）、商学伟和齐逸（视频流）

操作视频录制：陈方舟

技术文档撰写：吴东杰