上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

高压数字测量系统课程设计

中 期 报 告

姓名 谢弘洋

学号 515021910641

同组姓名 孟诗涵 陈晓彤

2018 年 11 月 15 日

一 进度概述

在半个学期的时间里，小组成员利用课余时间，查阅资料、实践探索，在共同努力下使得项目进度稳步推进，目前已经完成的内容如下：

1. 搭建云服务器平台，安装及配置服务器框架、资源和端口。
2. 云服务器程序的编写，实现了数据接收和进行简单处理的功能。
3. 4G 模块的调试和使用，能够通过串口实现和服务器的双向数据收发。
4. 在 PC 端编写 Python 程序，实现串口数据发送和接收功能，为下一步大量数据点的发送和接收做好准备。

下面将对各个任务的具体内容以及小组成员所作工作进行简要叙述。

二 云服务器搭建

1. 云服务器的购买与登录

本小组选择腾讯云作为项目初期使用的云服务器平台，进入腾讯云官方网站（https://cloud.tencent.com/），绑定微信登录后，完成在校学生认证后即可以 10 元/月的价格购买指定的服务器套餐，并拥有两次续费机会（每次最长 12 个月）。在购买界面可以选择服务器操作系统，本小组选择了 CentOS 系统，作为 Linux 发行版之一，在网上能找到许多相关的资料和服务器搭建教程，方便学习与实践。

图 1: 控制台安全组菜单

云服务器的正常使用需要首先为其配置安全组，在腾讯云服务器控制台界面进入左侧的安全组菜单，点击新建安全组，由于当前的云服务器仅作为学习探索使用，并没有重要资料，可以直接使用提供的“放通全部端口”的模板。完成安全组创建后，在云主机列表中，选择“更多”操作中的“安全组”-“配置

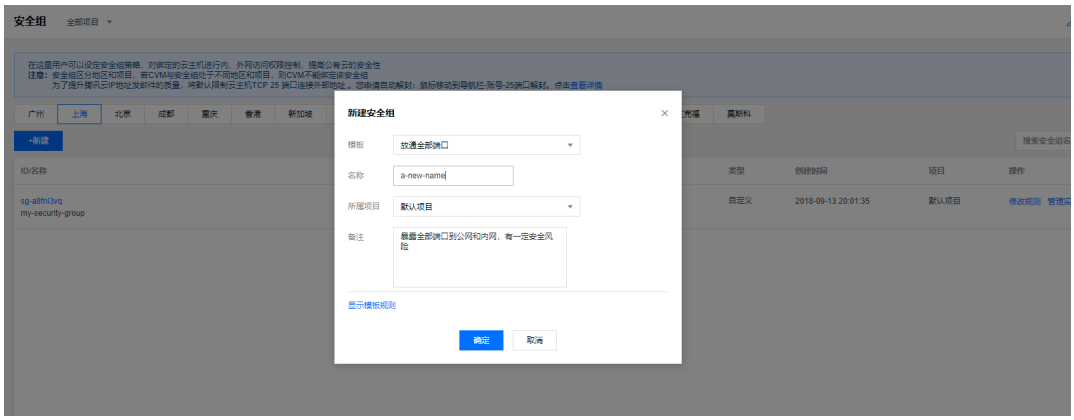

图 2: 新建安全组

图 3: 配置安全组

安全组”，选择刚刚创建的安全组；也可以直接选择“安全组”-“一键放通”完成安全组的配置。完成安全组的配置后，即可以选择“登录”操作，初始用户名“root”（最终权限用户），填入购买服务器后收到的密码即可完成云服务器登录。由于初始密码往往复杂且无规律，可以从云主机列表的“更多”操作下选择“密码/密钥”-“重置密码”完成密码重置。

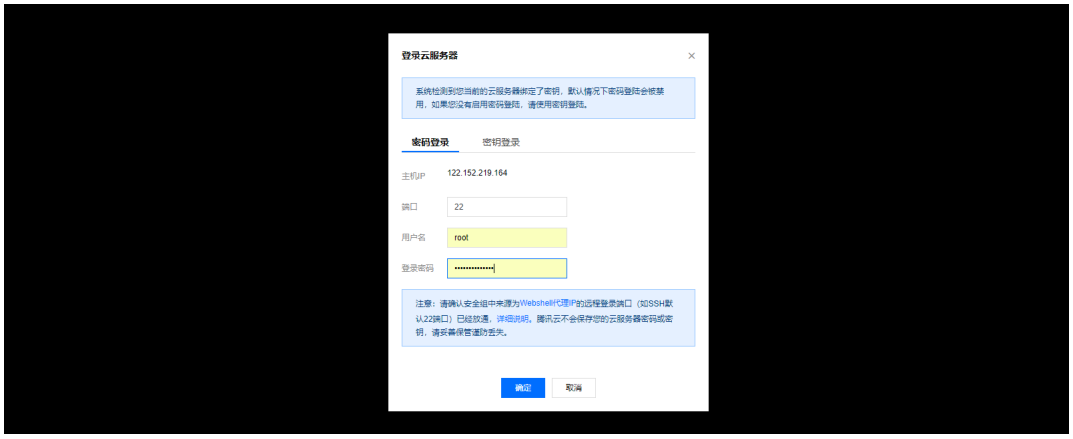

图 4: 登录云服务器

2. 使用 SSH 密钥登录云服务器

上述登录云服务器的操作以用户名和密码的方式，通过腾讯云提供的 Web-Shell 完成，而如果配置了 SSH 密钥，不仅可以更为快捷地完成登录操作，也能够通过各种远程登录工具（如 PuTTY）实现云服务器的登录和相关操作。

首先完成 SSH 密钥的创建和绑定，进入控制台左侧的 SSH 密钥菜单，新建密钥，选择“创建新密钥对”，并下载创建的密钥。之后，首先需要将正在运行的云主机关机，之后同样从“更多”操作中选择“密码/密钥”-“加载密钥”，选择之前创建的密钥名称，完成绑定。此时，再次选择登录操作，选择“密钥登录”，在“密钥文件”一栏选择之前下载的密钥文件，即可完成登录。

而使用 PuTTY 登录云服务器首先需要下载 PuTTY（https://www.putty.org/）。完成安装后，首先运行 PuTTYgen，选择“Conversions”-“Import Key”，加入之前下载的密钥文件后，点击“Save private Key”，保存私钥。之后，打开 PuTTY，配置远程会话模板。在“Connection”-“Data”中的“Auto-login usename”一栏填入默认用户名“root”；在“Connection”-“SSH”-“Auth”中选择加入之前保存的私钥文件；最后，在“Session”中的“Host Name”中填入服务器的公网 IP 地址，端口选择 22，并保存当前会话，之后即可以直接按照本次配置进行云服务器的登录。点击“Open”即可登录云服务器。

3. 云服务器端资源安装

3.1 Python 安装

由于最新版本的 Python 和服务器操作系统的尚存在不少兼容性问题，因此选择安装较为成熟的 Python3.6.4 版本。首先进入希望的安装目录：

```
cd ..
cd usr/local
```

之后下载 Python 安装包：

```
wget https://www.python.org/ftp/python/3.6.4/Python-3.6.4.tgz
```

解压安装包：

```
tar -zxvf Python-3.6.4.tgz
```

安装到指定位置（由第二条语句配置）：

```
cd Python-3.6.4
```

```
./configure --prefix=/usr/local/python3
make && make install
```

配置软连接（相当于环境变脸配置，由于 CentOS 已经安装了 Python-2.7，因此需要将 python 命令关联到 Python-3.6.4 上）：

```
ln -s /usr/local/python3 /usr/bin/python
```

此时，在任意位置输入“python”命令，既可以进入 Python 环境。

3.2 Python 虚拟环境和 Python 库安装

由于使用 Python 进行 Web 应用的开发需要安装许多的第三方库，为了防止不同项目安装的不同版本的库之间相互干扰，需要在 Python 虚拟环境来对不同项目进行隔离。Python-3 以上版本通过自带的 Pyvenv 提供了虚拟环境功能，使得虚拟环境的创建变得较为简单。首先进入存放项目的目录，例如：

```
cd pythonweb
```

之后使用 pyvenv 创建虚拟环境：

```
pyvenv flasktest
```

该命令即在 pythonweb 目录下新建了 flasktest 目录，该目录即为即将开发的项目，并已经包含了虚拟环境。

虚拟环境的启动由对应目录中的 activate 程序完成：

```
source bin/activate
```

启动虚拟环境后，命令行最前面将出现虚拟环境的名称，表示虚拟环境成功开启。如果需要关闭虚拟环境，只需要在任意位置输入 deactivate 命令即可。

```
deactivate
```

开启虚拟环境既可以在其中安装所需要的 Python 库而不会干扰其他项目，例如使用 pip 安装 flask 库：

```
pip install flask
```

三 服务器端程序编写

1. TCP 通讯程序

若 4G 模块直接通过 TCP 方式与服务器进行通讯，则需要在云服务器端编写相应的 TCP 通讯程序。Python 提供了 socket 库来完成 TCP 通讯。对应的 Python 程序如下所示：

```
import socket, threading

#创建socket实例
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
#对本机在公网的IP地址, 5000端口进行监听
s.bind(('0.0.0.0',5000))
#最大连接数
s.listen(5)

def tcplink():
  #创建TCP连接实例
    sock, addr = s.accept()
    sock.send(b'Welcome!')
    #循环接收信息
    while True:
        msg = sock.recv(1024)
        if msg.decode('utf-8') == 'exit':
            break
        sock.send(('Hello,%s'%msg.decode('utf-8')).encode('utf-8'))

#启动TCP连接线程
t = threading.Thread(target = tcplink, args = ())
t.start()
```

2. HTTP 通讯程序

由于 4G 模块提供了 HTTPD 模式，因此也可以通过 HTTP 方式实现与服务器端的通讯。使用 flask 来完成 HTTP 的解析任务，因此在 flask 框架下只需要编写相应的数据提取和响应程序即可。

```
from flask import Flask,request
from flask_script import Manager

app = Flask(__name__)
manager = Manager(app)
```

```
@app.route('/')
def index():
    return '<h1>Hello World!</h1>'

#对带有/test的url进行响应
@app.route('/test',methods=['GET','POST'])
def test():
  #请求为GET方式，从url中的args提取数据，关键字为"data"
    if request.method  == 'GET':
        msg = request.args.get('data')
    #请求为POST方式，从请求体中获取参数，关键字为"data"
    else:
        msg = request.form.get('data')
    #将请求中附带的数据返回作为相应
    return msg

if __name__ == '__main__':
    manager.run()
```

四 4G 模块的使用

使用有人物联网公司的 4G 透明传输模块 USR-LTE-7S4 作为项目中的 4G 模块，该模块支持 TCP 透传和 HTTPD 两种工作模式，支持域名 DNS 解析、套接字分发、心跳数据包等功能。

该 4G 模块能够通过一系列“AT+ 指令”实现对模块工作模式和相应参数的查询和设置，通过连接的串口对模块发送对应的语句即可实现，而在 PC 端，有人物联网公司提供了配套的 USR-G78X 软件，能够直接在 UI 界面上实现对模块的模式和参数设置，能够大大简化模块测试使用阶段的工作，加快和云服务器的通讯功能验证进程。

通过 USB 转串口模块将 4G 模块与 PC 连接后，打开 USR-G78X 软件，选择对应的端口号，选择波特率为 115200，“检验/数据/停止”分别为“None/8/1”，选择“流控”为“None”，之后打开串口。点击“进入配置状态”后，在软件左侧界面完成相应的设置，点击“设置并保存所有参数”，软件自动将对应的 AT 指令发送给模块。设置完成后，点击模块重启，即可完成工作模式和参数的设置。

1. 网络透传模式

在网络透传模式下，设置服务器地址为云服务器的公网 IP，端口设置为服务器端 TCP 通讯程序中设置的监听端口，连接类型设置为 TCP、长连接，连接超时设置 5 秒。依次点击“设置并保存所有参数”、“模块重启”后，即可以在右下方发送窗口中输入发送的内容，点击发送后可以在右侧中间看到收到的服务器发回的内容。同样，也可以在完成网络透传模式的设置后，点击“关闭串口”，使用其他的 PC 端串口助手实现和服务器端的 TCP 通讯。

2. HTTPD 模式

选择 HTTPD 模式，选择 HTTP 请求方式为“GET”，HTTP 请求的 URL 为“/test[3F]”，服务器地址为云服务器的公网 IP，端口设置为云服务器端 flask 框架内设置的端口。依此点击“设置并保存所有参数”、“模块重启”。之后，在软件右下方发送窗口中输入“data=...”，点击发送即可在右侧中部看到服务器端发回的“...”数据。

同样，在完成了 HTTPD 模式的设置后，也可与关闭软件，使用其他的 PC 端串口助手实现和服务器端的 HTTPD 通讯。

五 PC 端串口程序编写

通过 4G 模块配套提供的软件或者是其他 PC 端串口助手工具，只能实现手动输入的内容的发送，而如果要进行 2048 个波形数据点的发送、服务器端处理、结果返回的测试，显然从正弦数据点的生成以及数据量的角度都无法通过手动输入的方式实现，因此考虑编写 Python 程序来自动实现测试数据点的生成、通过串口发送以及接收返回结果的功能。

另外，经过反复测试，TCP 透传的方式比 HTTPD 的通讯模式相比，具有更好的稳定性和可靠性，处理和响应的延迟时间也更小，也不会出现得不到服务器端接收不到请求的情况，同时，对于之后 FPGA 的编程，由于 TCP 透传模式只要通过串口给出数据即可，而 HTTPD 模式还需要给出相应的请求头、请求体，因此 TCP 透传模式下的 FPGA 编程也更容易实现，因此首先考虑实现 TCP 透传模式下的波形数据测试。

对于该 Python 程序而言，首先需要实现通过 PC 端串口进行数据的收发，Python 中的 serial 库提供了串口打开、串口设置和串口数据收发的一系列功能，编写程序如下所示。