Worcester Polytechnic Institute
Department of Computer Science
CS4516– Computer Networks

Project 1 – SMTP Client

Goal: This assignment is to help you understand the SMTP protocol. You will also gain experience in implementing a standard protocol using Python.

Instructions:
Your task is to develop a simple mail client that sends email to any recipient. Your client will need to connect to a mail server, dialogue with the mail server using the SMTP protocol, and send an email message to the mail server. Python provides a module, called `smtplib`, which has built in methods to send mail using SMTP protocol. However, we will not be using this module in this lab, because it hides the details of SMTP and socket programming.

The skeleton code is provided later in this assignment. You are to complete the skeleton code. The places where you need to fill in code are marked with **`#Fill in start`** and **`#Fill in end`**. Each place may require one or more lines of code.

In some cases, the receiving mail server might classify your e-mail as junk. ***Make sure you check the junk/spam folder when you look for the e-mail sent from your client***.

Detailed Instructions:

In this assignment, you'll need a server machine (e.g. the SMTP server of Gmail), and a client machine (e.g. your local host, on which you run the program you wrote). You do not need to do anything on the server machine because you'll use the existing mail server. You only need to run the client program you wrote on the client machine. Your client program will connect with your Google mail server, got authenticated for your gmail account, and send out an email from your gmail to any recipient. Figure 1 shows an example of running the SMTP client program.

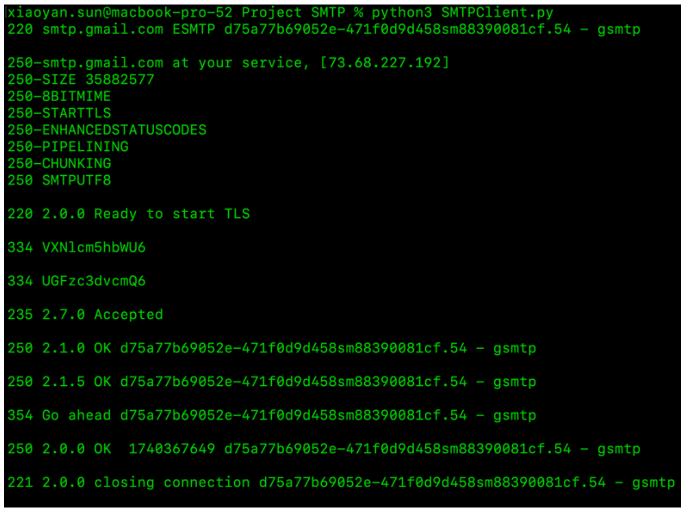
Mail servers like Google mail (**address: smtp.gmail.com, port: 587**) requires your client to add a Transport Layer Security (TLS) or Secure Sockets Layer (SSL) for authentication and security reasons, before you send MAIL FROM command. Add TLS/SSL commands to implement your client using Google mail server at above address and port.

To pass authentication of your gmail account, please follow the instruction below to create an App password for your gmail:
https://support.google.com/accounts/answer/185833?visit_id=638759601307026124-2089972828&p=InvalidSecondFactor&rd=1
You will need to **enable two factor authentication first** before you can create an App password.

Please be noted that this password is different from your regular gmail password. It can be used for the untrusted applications, such as the program you write for this assignment. Do NOT disclose your App password to anyone else. You still need to protect this password as your regular password. I would suggest:

1) Create a new gmail account for this assignment so you do not accidentally disclose your App password;
2) When you submit your code, remove your sender gmail and password. **SAs will use their own**

sender gmail and password to run your code.



```
xiaoyan.sun@macbook-pro-52 Project SMTP % python3 SMTPClient.py
220 smtp.gmail.com ESMTP d75a77b69052e-471f0d9d458sm88390081cf.54 - gsmtp

250-smtp.gmail.com at your service, [73.68.227.192]
250-SIZE 35882577
250-8BITMIME
250-STARTTLS
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-CHUNKING
250 SMTPUTF8

220 2.0.0 Ready to start TLS

334 VXNlcm5hbWU6

334 UGFzc3dvcmQ6

235 2.7.0 Accepted

250 2.1.0 OK d75a77b69052e-471f0d9d458sm88390081cf.54 - gsmtp

250 2.1.5 OK d75a77b69052e-471f0d9d458sm88390081cf.54 - gsmtp

354 Go ahead d75a77b69052e-471f0d9d458sm88390081cf.54 - gsmtp

250 2.0.0 OK  1740367649 d75a77b69052e-471f0d9d458sm88390081cf.54 - gsmtp

221 2.0.0 closing connection d75a77b69052e-471f0d9d458sm88390081cf.54 - gsmtp
```

Figure 1. Snapshot of running SMTPClient

Deliverables:

1. The client code.
2. A README (txt) file that spells out exactly how to run the code.
3. Two screenshots: one showing running the client code in terminal; another one showing the test email received by your recipient email (WPI email preferred).

Code plagiarism is absolutely **NOT** allowed. If needed, you may be asked for a **demonstration** of running your program in front of the instructor/SAs and answer their questions about your code. In this case, your grade will be based on both the report and your performance during demonstration.

Grading:

Grade breakup (%) assuming all documents are present:

1. Client code = 90 points
2. Readme file = 5 points
3. Screenshots = 5 points

The grade will be a **ZERO** if the code does not run or gives a run-time error.

To get the FULL CREDIT for the code:

1. The code should be able to send an email successfully;
2. The recipient email should receive the test email successfully.

Notes:

1. If you cannot install Python on your own machine, you can connect to ***linux.wpi.edu*** **server** on campus network to do the assignment. If you are off campus and ***you try to run the programs on the server***, you need to use VPN to connect back to campus network and use ssh to connect to the server. You don't need VPN or ssh if you run all your programs on your local machine.
   Please see more instruction here:
   https://hub.wpi.edu/article/786/connect-to-linux-cluster-using-a-terminal-program
2. When you connect to linux.wpi.edu, you may get hostnames other than `linux.wpi.edu` (as given the examples above). Please use the command `hostname` to find out where you are connected.

3. If you need to kill a background process after you have started it, you can use the UNIX *kill* command. Use the UNIX *ps* command to find the process id of your server.
4. Make sure you close unused sockets that you use in your program.
5. If you abort your program, the socket may still hang around and the next time you try and bind a new socket to the port ID you previously used (but never closed), you may get an error.
6. On UNIX systems, you can run the command "netstat" to see which port numbers are currently assigned.