

### Assignment 1 Mail Client (10points)

I used python 3 to implement the mail client, relying heavily on the smtplib library to handle the creation of the smtp ports and uses TCP. Calling the SMTP\_SSL method from smtplib will create a smtp socket with the smtp server being contacted and its port. Since the gmail smtp server needs some form of encryption and for the port I chose to connect through, I needed to use SSL encryption. For this I imported the ssl library and using the create\_default\_context method creates an object to handle the SSL encryption. Passing this object into the socket creation will encrypt the data passing through the socket automatically. Using the “with” statement to create the socket, the socket will automatically be closed after the “with” statement completes.

For simplicity’s sake the email’s and the body text is hardcoded and running the client displays the to email, from email, and the body before sending the message.

```
MINGW64/e/Programing/Computer Networks/Computer-Networks-Midterm-Project
robbo@RobPC MINGW64 /e/Programing/Computer Networks/Computer-Networks-Midterm-Project (master)
$ python Mail\ Client\smtp_mail_client.py
Sending this email:

To: robbob345@gmail.com
From: compnettest345@gmail.com
Subject: This was sent using python

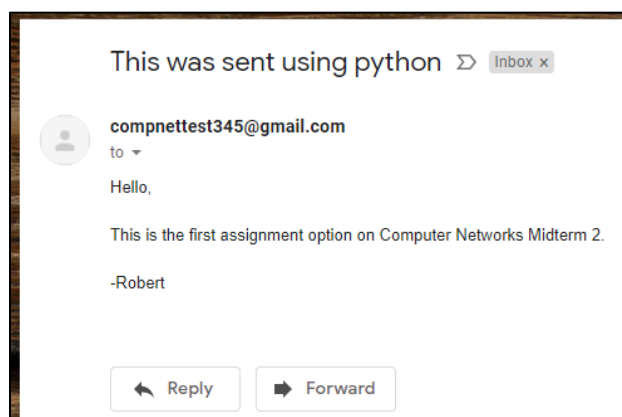
Hello,

This is the first assignment option on Computer Networks Midterm 2.

-Robert

robbo@RobPC MINGW64 /e/Programing/Computer Networks/Computer-Networks-Midterm-Project (master)
$
```

Sent email:



### Assignment 3: ICMP Ping(10points)

Again, I used python 3 to implement the ICMP ping tool, which utilized the pythonping library’s ping method. The ping method handles sending the ICMP packets and receives the responses and

stores the data in a return object. Using the return object, I track the maximum RTT, minimum RTT, and number of missed packets, as well as calculating the percentage packet loss, average RTT, and the standard deviation of the RTTs. I imported the statistics library to use the stdev method to calculate the standard deviation, as well as importing the time library to space out the time between sending the ICMP packets.

The tool asks for the user to input an IP address to ping and the number of packets to send. Then returns each response and after all responses, displays the stats.

Output of the ping tool:

```
MINGW64/e/Programing/Computer Networks/Computer-Networks-Midterm-Project
robbo@RobPC MINGW64 /e/Programing/Computer Networks/Computer-Networks-Midterm-Project (master)
$ python ICMP\ Ping\icmp_ping_tool.py
Address to ping: 8.8.8.8
Number of packets: 5

Reply from 8.8.8.8, 9 bytes in 23.63ms
Reply from 8.8.8.8, 9 bytes in 21.35ms
Reply from 8.8.8.8, 9 bytes in 28.52ms
Reply from 8.8.8.8, 9 bytes in 19.36ms
Reply from 8.8.8.8, 9 bytes in 20.56ms

Packet Loss: 0.00%
Min RTT: 19.36
Max RTT: 28.52
Avg RTT: 22.684
Std Deviation of RTT: 3.6157198453420034

robbo@RobPC MINGW64 /e/Programing/Computer Networks/Computer-Networks-Midterm-Project (master)
$
```