

CS331 Project 1 Write Up – Ben Webb

The purpose of this project was to explore SMTP, particularly learning about how Mail Transfer Agents interact with one another.

The first part of this project challenged us to explore SMTP server action. To do this I logged into gloin.cs.colby.edu and opened the local mail server with the command: `nc localhost 25` which is the command to open the Ubuntu postfix mailserver port, 25.

```
bmwebb20@gloin:~$ nc localhost 25
220 gloin.cs.colby.edu ESMTP Postfix (Ubuntu)
helo gloin.cs.colby.edu
250 gloin.cs.colby.edu
mail from: cs331@gloin.cs.colby.edu
250 2.1.0 Ok
rcpt to: bmwebb20@colby.edu
550 5.1.1 <bmwebb20@colby.edu>: Recipient address rejected: colby.edu
rcpt to: bmwebb20@gloin.cs.colby.edu
250 2.1.5 Ok
data
354 End data with <CR><LF>.<CR><LF>
subject: SMTP test
This is Ben Webb testing his email.
.
250 2.0.0 Ok: queued as 0E8187C007F
quit
221 2.0.0 Bye (Ubuntu)
```

In this terminal interaction, I first type: `Helo gloin.cs.colby.edu` which lets the mail server know I am connecting to send mail. I then add the from address, cs331@gloin.cs.colby.edu. When I try to send an email to my email ending in @colby.edu, the address is rejected because Colby.edu is not local and . I am successful when I send an email to an address that ends in @gloin.cs.colby.edu because the message is delivered as seen below.

```
bmwebb20@gloin:~$ mail
Mail version 8.1.2 01/15/2001. Type ? for help.
"/var/mail/bmwebb20": 2 messages 2 new
>N 1 cs331@gloin.cs.co Mon Oct 7 09:31 14/576 SMTP test
```

Then I tried to send an email from a non-local address.

```
From bmwebb20@colby.edu Tue Oct 8 13:33:13 2019
X-Original-To: bmwebb20@gloin.cs.colby.edu
Date: Tue, 8 Oct 2019 13:32:29 -0400 (EDT)
From: bmwebb20@colby.edu

This is Ben Webb testing his email.
subject: SMTP test
```

This interaction was successful. This suggests that the From: field does not matter when you are running on the localhost, because all addresses will not be evaluated prior to sending. This had me a bit confused because this in combination with the first section goes against some of SMTP server action policy. Upon [looking it up](#), it appears that the rcpt to: field is actually evaluated, and because this MTA is not connected to any external servers, the message will never be delivered.

The second part of this project was to write as short as possible of a program to have an SMTP conversation. I decided to use Python to do this. The entire code is 13 lines long.

```

1 import socket
2 s, data = socket.socket(socket.AF_INET, socket.SOCK_STREAM), ''
3 s.connect(('localhost', 25))
4 while True:
5     print(s.recv(1024).decode())
6     if data.lower() == 'data':
7         data = input('> ')
8         s.send((data + '\r\n.\r\n').encode())
9         continue
10    elif data.lower() == 'quit': exit()
11    data = input('> ')
12    s.send((data + '\r\n').encode())

```

The program runs by being logged into gloin and typing: `python3 smtpclient.py`. The first line imports socket. The second and third lines create the socket and message object and then connects to the server. It then enters a while loop. There are only two special cases, data, which needs a special end character and quit, which is how the program ends. Below is an example of the code in use. User entry is denoted by the '>'.

```

bmwebb20@gloin:~/Senior Year/CS331/coding things$ python3 smtpclient.py
220 gloin.cs.colby.edu ESMTP Postfix (Ubuntu)

> HELO gloin.cs.colby.edu
250 gloin.cs.colby.edu

> mail from: cs331@gloin.cs.colby.edu
250 2.1.0 Ok

> rcpt to: bmwebb20@gloin.cs.colby.edu
250 2.1.5 Ok

> data
354 End data with <CR><LF>.<CR><LF>

> I wanted to email myself, because I, Ben Webb am a narcissist
250 2.0.0 Ok: queued as B75937C007F

> quit
221 2.0.0 Bye

You have new mail in /var/mail/bmwebb20
bmwebb20@gloin:~/Senior Year/CS331/coding things$

```

The final part of this project was to implement another program that performed the same actions as the python script, but using the linux tool expect. Expect essentially allows for there to be a response. Expect works by pairing phrases entered in the command line with answers. The code looks like this:

```

1 #!/usr/bin/expect -f
2 set timeout -1
3 spawn nc localhost 25
4 expect "220 gloin.cs.colby.edu ESMTP Postfix (Ubuntu)\r"
5 send -- "HELO gloin.cs.colby.edu\r"
6 expect "250 gloin.cs.colby.edu\r"
7 send -- "mail from: cs331@gloin.cs.colby.edu\r"
8 expect "250 2.1.0 Ok\r"
9 send -- "rcpt to: bmwebb20@gloin.cs.colby.edu\r"
10 expect "250 2.1.5 Ok\r"
11 send -- "data\r"
12 expect "354 End data with <CR><LF>.<CR><LF>\r"
13 send -- "subject: eMailHeader\r"
14 send -- "body of the email\r"
15 send -- ".\r"
16 expect "250 2.0.0 Ok: queued as *\r"
17 send -- "quit\r"
18 expect eof
19

```

It first connects to the localhost, and then automatically has a conversation with the mailserver. Below is an example of it in use.

```

[bmwebb20@gloin:~/Senior Year/CS331/coding things$ ./smtp.exp
spawn nc localhost 25
220 gloin.cs.colby.edu ESMTP Postfix (Ubuntu)
HELO gloin.cs.colby.edu
250 gloin.cs.colby.edu
mail from: cs331@gloin.cs.colby.edu
250 2.1.0 Ok
rcpt to: bmwebb20@gloin.cs.colby.edu
250 2.1.5 Ok
data
354 End data with <CR><LF>.<CR><LF>
subject: eMailHeader
body of the email
.
250 2.0.0 Ok: queued as 315E37C007F
quit
221 2.0.0 Bye

```

Extensions

For my first extension I decided to play with using SSL Authentication to send an email on the gmail server. This was done in the file gmailclient.py and only imports socket and ssl for network programming. This process is very similar to the python file used for having an SMTP conversation but is much less compressed and there is an extra authentication step. The authentication is done using the command: "AUTH PLAIN \0username\0password\r\n". Below is an example of the code working:

```

Benjamins-MacBook-Pro-5:coding things Ben$ python gmailclient.py
Username: bmwebb20@colby.edu
Password:
Connected: 220 smtp.gmail.com ESMTP l23sm15370413qta.53 - gsmtp

Login: 235 2.7.0 Accepted

Mail From Response: 250 2.1.0 OK l23sm15370413qta.53 - gsmtp

Email Address of Recipient: bmwebb20@colby.edu
Recipient Response: 250 2.1.5 OK l23sm15370413qta.53 - gsmtp

Subject: Extension Email

Message:
    This is part of my extension for project 2.

Transmitting Message 354 Go ahead l23sm15370413qta.53 - gsmtp

Message Sent: 250 2.0.0 OK 1570557698 l23sm15370413qta.53 - gsmtp

Quit: 221 2.0.0 closing connection l23sm15370413qta.53 - gsmtp

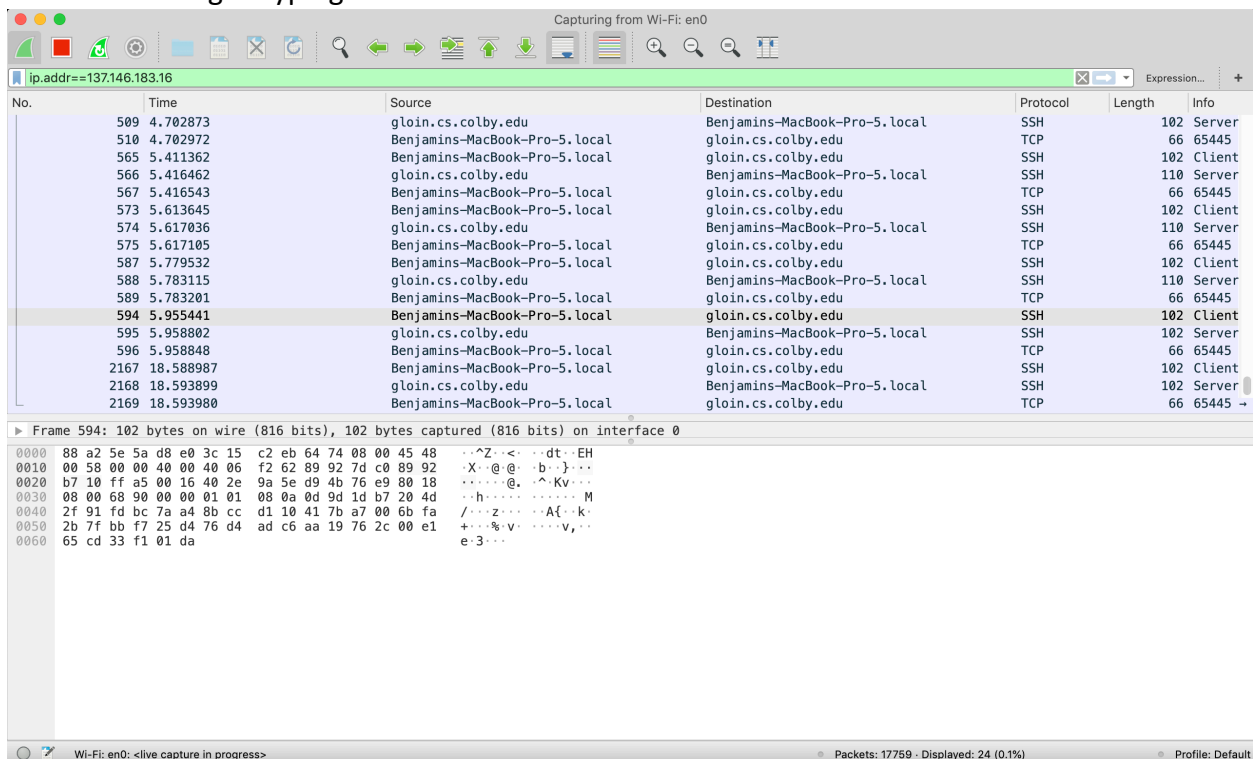
Benjamins-MacBook-Pro-5:coding things Ben$

```

I used the package getpass to allow for the password to be typed without echoing in the terminal. I also print out all of the status messages to show how the google server responds. Below is the actual email as seen in the gmail webpage that was delivered to myself.



I also began to wonder if the authentication required would also produce an encrypted email, so I set wireshark to track gloins IP (137.146.183.16). What was interesting was the number of communication packets even to just type one character into the terminal. I paused for a couple seconds and began typing.



You can also see all of the SSH communications and the associated TCP level communication. But the TCP packet is never seen when coming from gloin to my computer. This is also really cool because you can see that a single key press is encapsulated to 66 bytes as TCP and 102 at SSH. Then I assume the TCP side on gloin would be decapsulated to 66 bytes.

I then wanted to look at an SMTP interaction but from my side instead of SSH so I could actually see the going ons of the message. Below is the activity captured as I was logging in via my python gmail client program.

No.	Time	Source	Destination	Protocol	Length	Info
21673	229.577291	ns8.colby.edu	137.146.125.192	DNS	123	Standard query response 0x71a2 No such name PTR 178
21674	229.578089	ns8.colby.edu	137.146.125.192	DNS	150	Standard query response 0x6ef4 No such name PTR 9.1
21841	232.856023	137.146.125.192	gmail-smtp-msa.l.google.com	TLSv1.3	124	Application Data
21844	232.892447	gmail-smtp-msa.l.google.com	137.146.125.192	TCP	66	urdf(465) → 65490 [ACK] Seq=4032 Ack=867 Win=65280 L
21845	232.896326	gmail-smtp-msa.l.google.com	137.146.125.192	TLSv1.3	312	Application Data
21846	232.896375	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	65490 → urdf(465) [ACK] Seq=4096 Ack=933 Win=65280 L
21847	232.896523	137.146.125.192	gmail-smtp-msa.l.google.com	TLSv1.3	141	Application Data
21848	232.931610	gmail-smtp-msa.l.google.com	137.146.125.192	TCP	66	urdf(465) → 65490 [ACK] Seq=4251 Ack=962 Win=65280 L
21860	233.370790	gmail-smtp-msa.l.google.com	137.146.125.192	TLSv1.3	188	Application Data
21861	233.370844	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	65490 → urdf(465) [ACK] Seq=4250 Ack=961 Win=65280 L
21862	233.370981	137.146.125.192	gmail-smtp-msa.l.google.com	TLSv1.3	121	Application Data
21863	233.399550	gmail-smtp-msa.l.google.com	137.146.125.192	TCP	66	urdf(465) → 65490 [ACK] Seq=4251 Ack=961 Win=65280 L
21864	233.399555	gmail-smtp-msa.l.google.com	137.146.125.192	TLSv1.3	129	Application Data
21865	233.399639	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	65490 → urdf(465) [ACK] Seq=4251 Ack=962 Win=65280 L
21938	234.573331	137.146.125.192	ns8.colby.edu	DNS	88	Standard query 0xd9e PTR 201.124.146.137.in-addr.arpa
21939	234.578153	ns8.colby.edu	137.146.125.192	DNS	150	Standard query response 0x8bdf PTR 86.120.146.137.in-addr.arpa
21960	235.573555	137.146.125.192	ns8.colby.edu	DNS	87	Standard query 0x8bdf PTR 86.120.146.137.in-addr.arpa

Frame 21860: 188 bytes on wire (864 bits), 188 bytes captured (864 bits) on interface 0

```

0000  3c 15 c2 eb 64 74 88 a2 5e 5a d8 e0 08 00 45 00  <...dt...^Z...E...
0010  00 5e 81 1f 00 00 6d 06 d4 f9 ad c2 42 6c 89 92  <...m...BL...
0020  7d c0 01 d1 ff d2 51 89 bc 16 a4 50 39 c3 80 18  >...Q...PG...
0030  00 ff 73 fd 00 00 01 01 08 0a 64 89 86 3b 0d ac  <...f...d...;...
0040  08 8e 17 03 03 00 25 11 2a 8e b7 b8 d7 11 2a bc  <...q...*...*...
0050  ec 14 60 46 b5 07 a5 6e e0 5d 87 b4 2d 75 73 01  <...F...n...]-us...
0060  03 76 b6 3f f6 50 7d 9a e5 35 36 5c              <v...P...>56\

```

I then wanted to better answer the problem set so I specifically captured what happen when I sent my message:

No.	Time	Source	Destination	Protocol	Length	Info
26551	235.573555	gmail-smtp-msa.l.google.com	137.146.125.192	TLSv1.3	141	Application Data
26552	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	65490 → urdf(465) [ACK] Seq=933 Ack=4171 Win=130944
26553	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TLSv1.3	94	Application Data
26554	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	urdf(465) → 65490 [ACK] Seq=4171 Ack=961 Win=65280 L
26555	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TLSv1.3	145	Application Data
26556	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	65490 → urdf(465) [ACK] Seq=961 Ack=4250 Win=130944
26557	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	urdf(465) → 65490 [FIN, ACK] Seq=4250 Ack=961 Win=65280 L
26558	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	65490 → urdf(465) [ACK] Seq=961 Ack=4251 Win=131072
26559	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	65490 → urdf(465) [FIN, ACK] Seq=961 Ack=4251 Win=131072
26560	235.573555	137.146.125.192	gmail-smtp-msa.l.google.com	TCP	66	urdf(465) → 65490 [ACK] Seq=4251 Ack=962 Win=65280 L
26561	235.573555	137.146.125.192	137.146.127.255	NBNS	92	Name query NB WORKGROUP<id>
26562	235.573555	137.146.125.192	ns8.colby.edu	DNS	88	Standard query 0xda9e PTR 201.124.146.137.in-addr.arpa
26563	235.573555	137.146.125.192	ns8.colby.edu	DNS	88	Standard query 0x5bdf PTR 183.141.146.137.in-addr.arpa
26564	235.573555	137.146.125.192	ns8.colby.edu	DNS	150	Standard query response 0x5bdf No such name PTR 183
26565	235.573555	137.146.125.192	ns8.colby.edu	DNS	150	Standard query response 0xda9e No such name PTR 201
26566	235.573555	137.146.125.192	137.146.127.255	NBNS	92	Name query NB WORKGROUP<id>
26567	235.573555	137.146.125.192	ns8.colby.edu	DNS	83	Standard query 0xe14f PTR 7.64.31.68.in-addr.arpa
26568	235.573555	137.146.125.192	ns8.colby.edu	DNS	162	Standard query response 0xe14f No such name PTR 7.6
26569	235.573555	137.146.125.192	137.146.127.255	NBNS	92	Name query NB WORKGROUP<id>
26570	235.573555	137.146.125.192	ns8.colby.edu	DNS	88	Standard query 0xbf86 PTR 187.136.146.137.in-addr.arpa
26571	235.573555	ns8.colby.edu	137.146.125.192	DNS	150	Standard query response 0xbf86 No such name PTR 187
26572	235.573555	137.146.125.192	ns8.colby.edu	DNS	87	Standard query 0x8bdf PTR 86.120.146.137.in-addr.arpa

Frame 26551: 141 bytes on wire (1128 bits), 141 bytes captured (1128 bits) on interface 0

```

0000  3c 15 c2 eb 64 74 88 a2 5e 5a d8 e0 08 00 45 00  <...dt...^Z...E...

```

There is a continuous TLS, TCP rotation when sending the message, except the interaction ends with a set of TCP packets outside of TLSv1.3. TLS1.3 is an up to date version of TLS.