COMP-4476 - Assignment #4
Chris Campbell

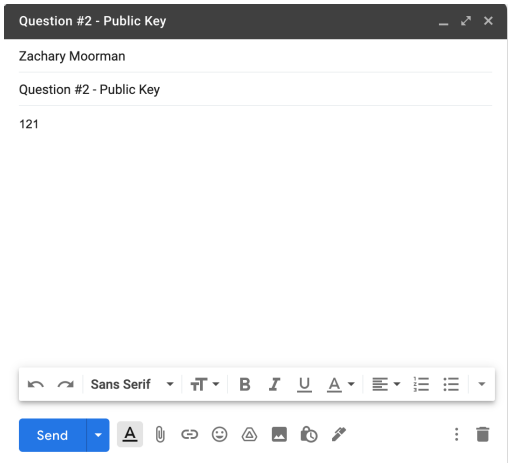
Question #1:

I would design a system that uses SSH for client-server architecture. I would also combine that with a one time password model for authentication. This way the design remains efficient and secure. Since passwords can only be used once there is no need to worry about an attacker figuring out your passwords.

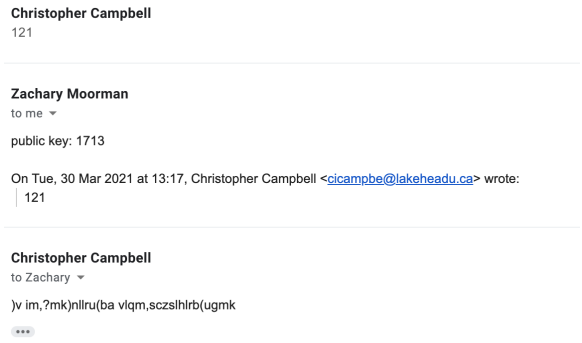
Question #2:

Please see code for question #2

Output photos:
Step 1: Send my public key to Zach



Step 2: Retrieve public key from Zach, create and send cipher text using common key to Zach



```
→  Assignment4 git:(master) ✗  /usr/bin/env /Libra
ication Support/Code/User/workspaceStorage/a555bf3
Send this number to your partner:
121
Enter public key from partner:
1713
```

```
ication Support/Code/User/workspaceStorage/a555bf3c
Send this number to your partner:
121
Enter public key from partner:
1713
)v im,?mk)nllru(ba vlqm,sczslhlrb(ugmk
Enter ciphertext from partner to decrypt:
```

Step 3: Retrieve ciphertext from Zach and decrypt using common key

Question #2 - Public Key Inbox ×

Christopher Campbell
121

Zachary Moorman
public key: 1713 On Tue, 30 Mar 2021 at 13:17, Christopher Campbell <cicampbe

Christopher Campbell
)v im,?mk)nllru(ba vlqm,sczslhlrb(ugmk

Zachary Moorman
to me

decrypted:
hello zach, its nice to work with you.

Here's my encrypted message:
)v im,glzqdnzrbm(sqdldyva(.c?vrqgci
...

```
→  Assignment4 git:(master) ✗  /usr/bin/env /L
ication Support/Code/User/workspaceStorage/a55
Send this number to your partner:
121
Enter public key from partner:
1713
)v im,?mk)nllru(ba vlqm,sczslhlrb(ugmk
Enter ciphertext from partner to decrypt:
)v im,glzqdnzrbm(sqdldyva(.c?vrqgci
hello chris, this is a test message
→  Assignment4 git:(master) ✗
```

Question #3

An efficient and secure way to create a common key between three parties can happen is four steps.

Step 1: Alice creates A with her secret key and sends A to Bob

- Alice sends A = **α**^a mod p to Bob

Step 2: Bob takes A and packages his secret key b, and also sends B to Carl

- Bob creates A^b
- Bob sends both A^b and B to carl
- A^b = **α**^ab mod p / B = **α**^b mod p

Step 3: Carl takes B and packages his secret key c, creates the common key A^bc, and sends C and B^c to Alice

- Carl creates B^c = **α**^bc mod p
- Carl creates his common key A^bc = **α**^abc mod p
- Carls sends both B^c and C = **α**^c mod p to Alice

Step 4: Alice creates common key A^bc, creates and sends C^a to Bob

- Alice creates common key **α**^abc mod p
- Alice creates C^a = **α**^ac mod p
- Alice sends C^a to Bob

Bob then takes C^a and creates common key **α**^abc mod p.

In four steps each Alice, Bob, and Carl have the common key without creating any security risk.