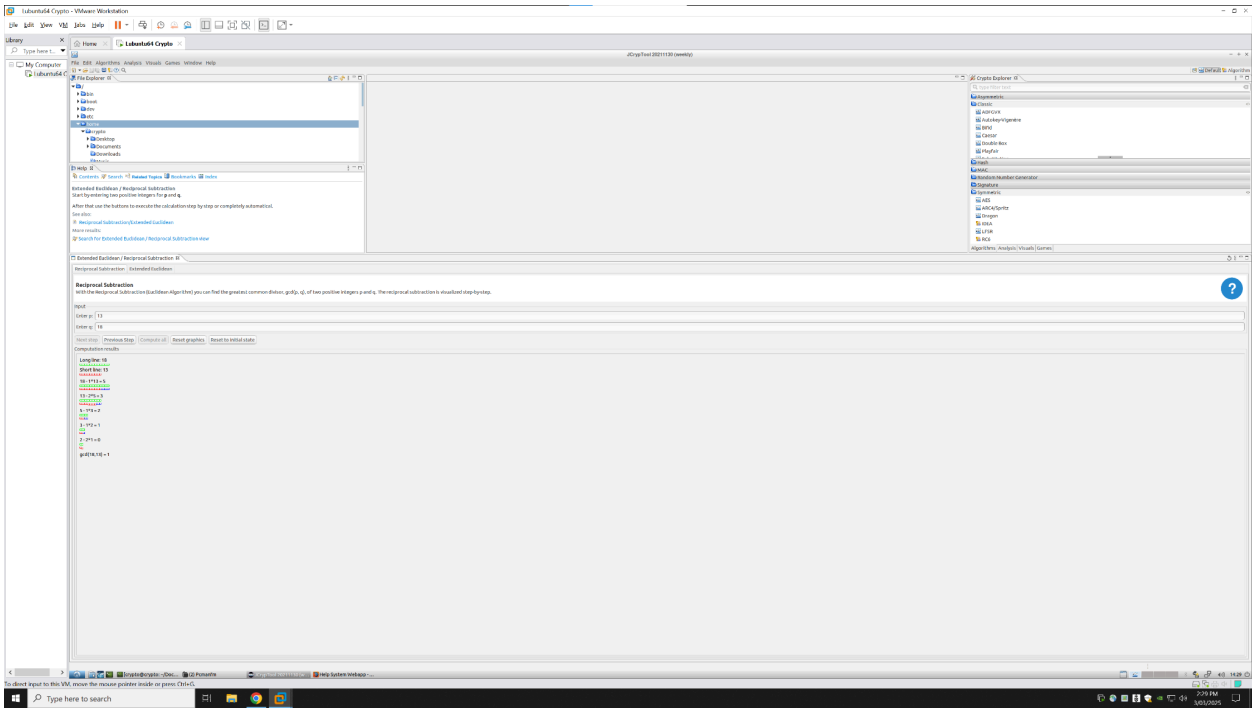
PART 1

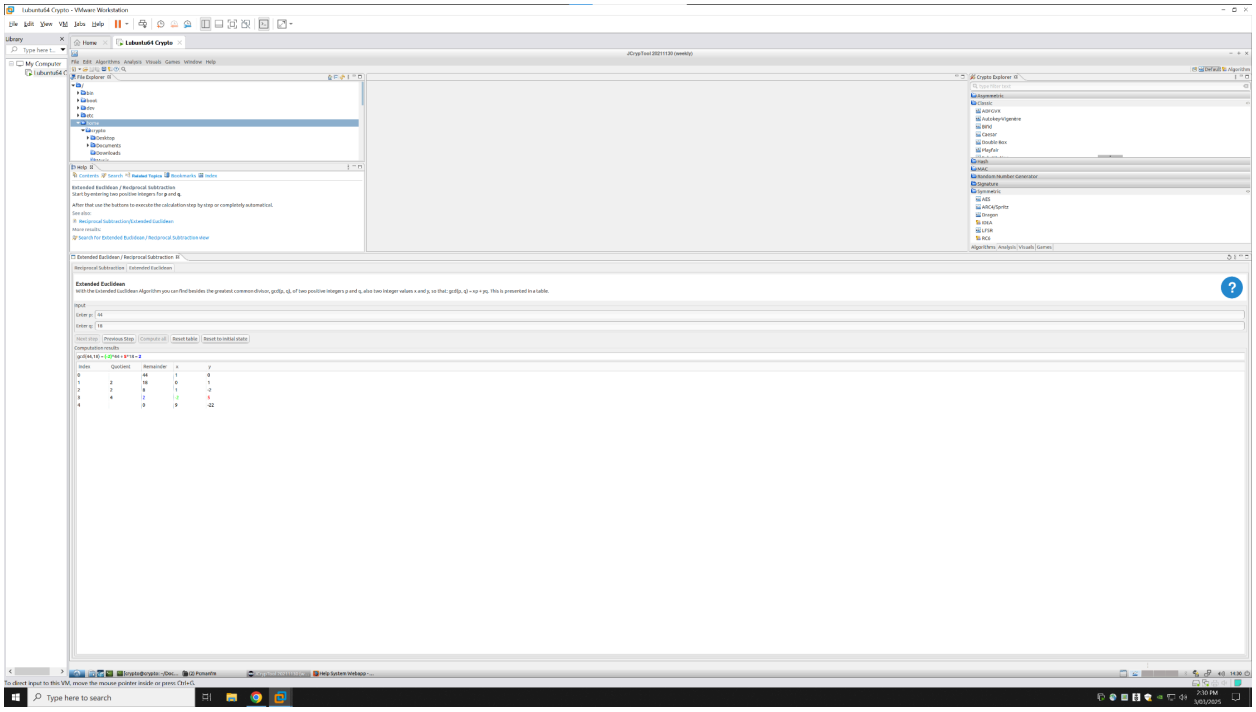
Euclidean Algorithm

- Way to find the greatest common divisor of 2 numbers
- Steps
    1. Open extended Euclidean/reciprocal subtraction under visuals in JCrypTool
    2. Click on reciprocal subtraction tab
    3. Select 2 numbers in the input
    4. Go through whole process
    5. Result: greatest common denominator of two numbers



Extended Euclidean Algorithm

- Finding x and y for ax + by = d = gcd(a,b)
- Steps
    1. Go to the tab “Extended Euclidean”
    2. Fill in p and q used in the Euclidean algorithm
    3. Click next step button



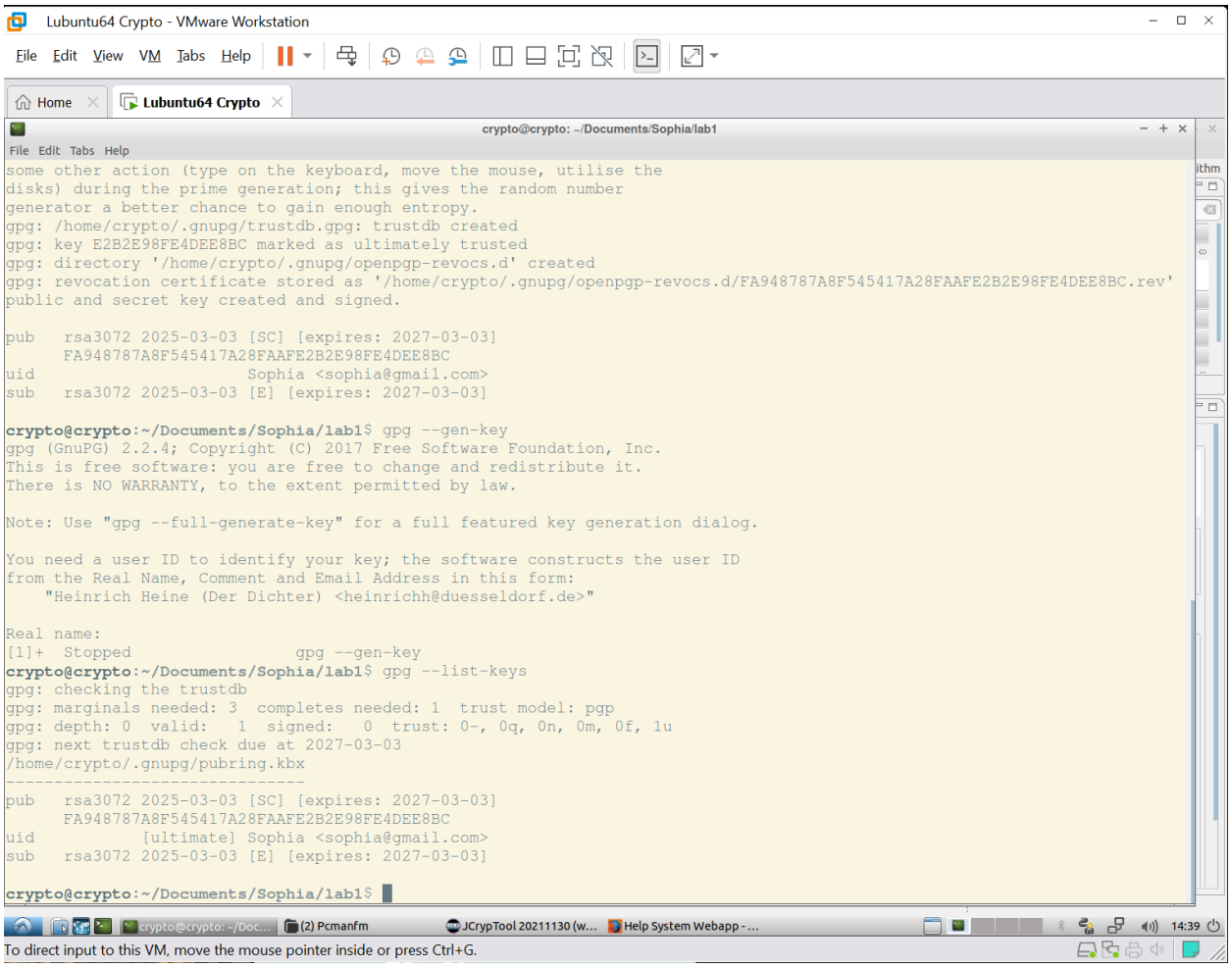
PART 2

Pretty Good Privacy (PGP)

- Encryption program- provides cryptographic privacy and authentication for data communication
    - Used for signing, encrypting and decrypting texts to increase security of e-mail communications
- Phil Zimmermann developed PGP in 1991
- OpenPGP- open standard of PGP encryption software

Generating public/private key pair

- Made email, user
- Also gain a sub key
- PGP- can sign keys, asking a third party to authenticate you by using their private key to sign your public key



Q:

1. Why is it necessary to sign keys?

It's necessary to sign keys to establish trust and to authenticate that you're a legitimate user. Without having your key signed, someone else could falsely claim to be the owner of the generated key

2. Can anyone create a key and pretend to be another person?

Yes, people can easily claim that the generated key belongs to someone else if they aren't asked for proper verification.

3. Can you think of a way to make sure that a given key really belongs to the person listed on the key?

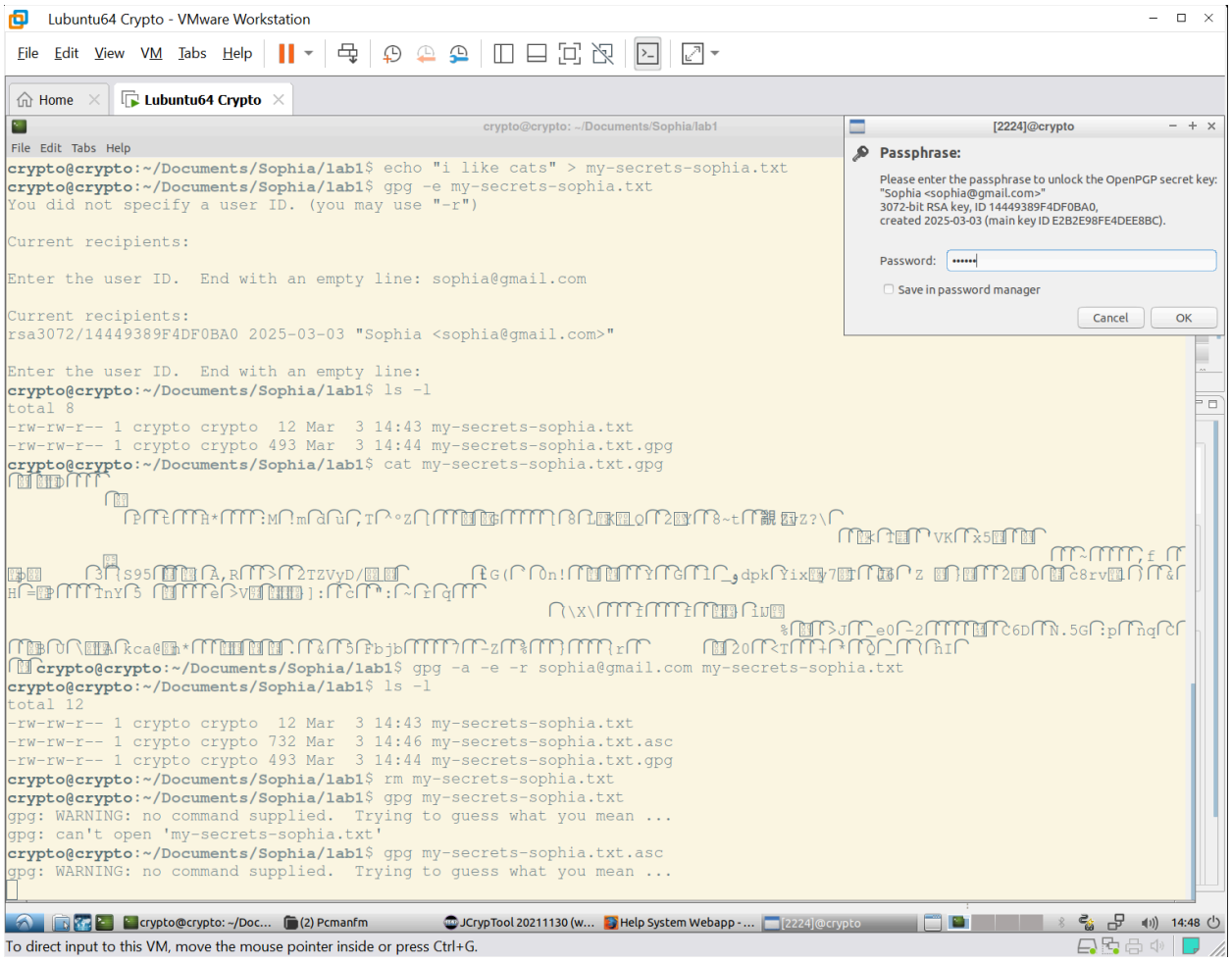
To ensure that the key legitimately belongs to the owner, digital signatures and certificate authorities can be used to prove their identity.

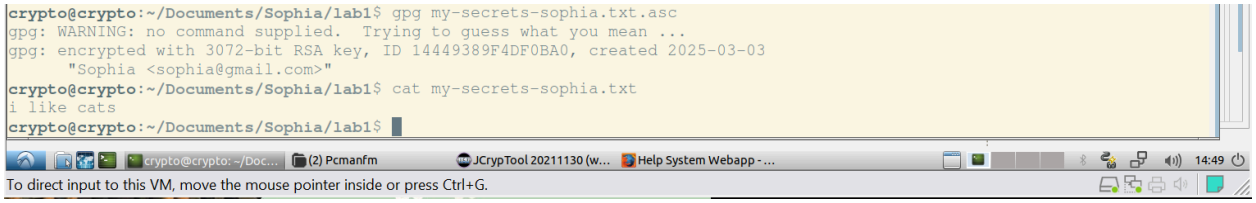
4. What do you think are the benefits of signing keys?

Signing keys is beneficial for the security of the cryptographic system since it ensures the user's authenticity and ensures that the key hasn't been altered.

Encrypt with GPG using public key

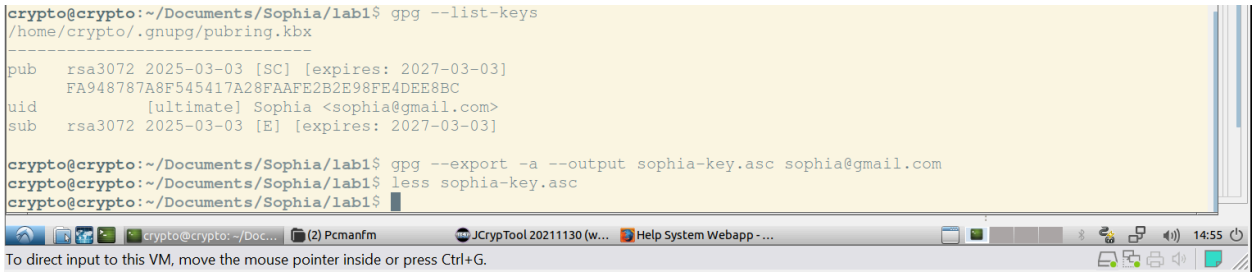
- GPG automatically figures out who the file is encrypted for by checking if you have the private key
- It will overwrite the encrypted file with the decrypted code

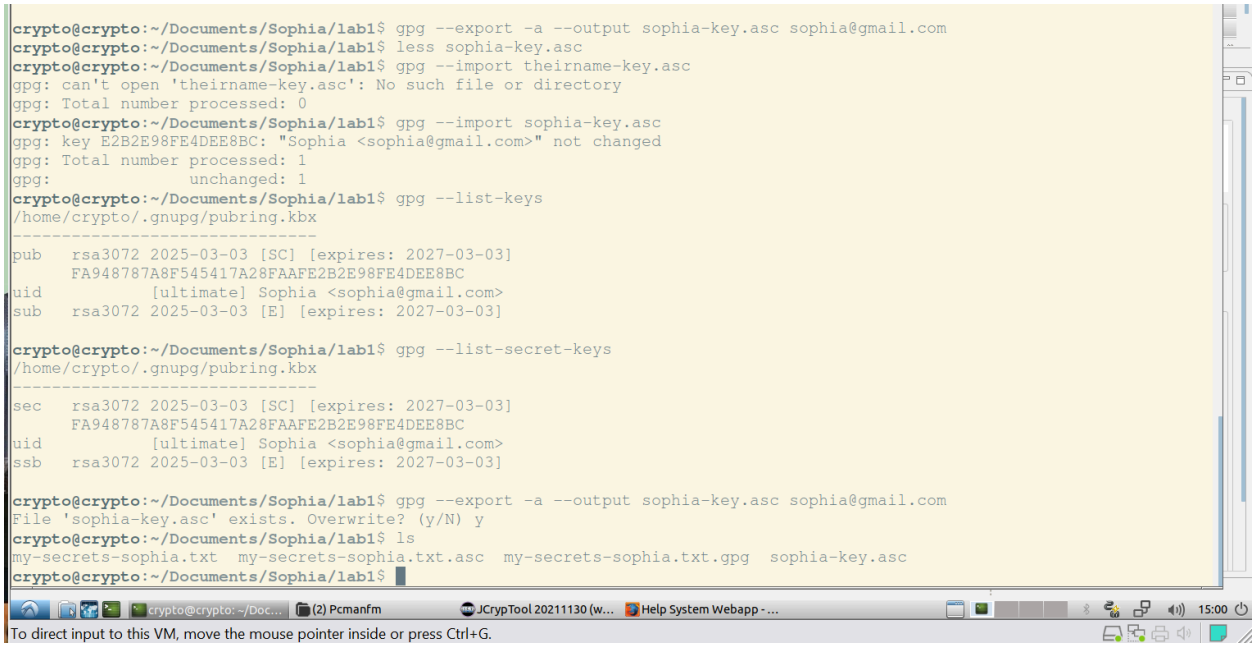


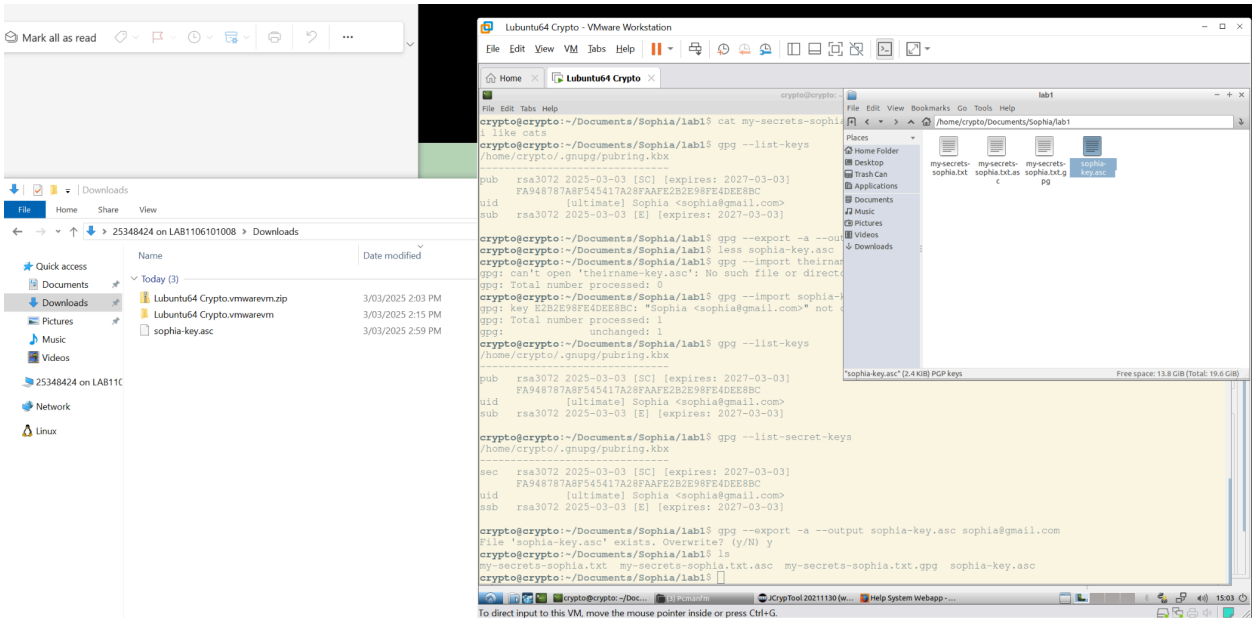


Distributing and trusting keys

- To be able to decrypt the files of another person, a copy of their public key is needed
- Steps
    1. Import the key of the recipient
    2. Encrypt the file using the key of the recipient (ensures they'll be able to decrypt the file using their key)
    3. Email the encrypted file to the recipient for them to decrypt
        - Receiving an email 2 times

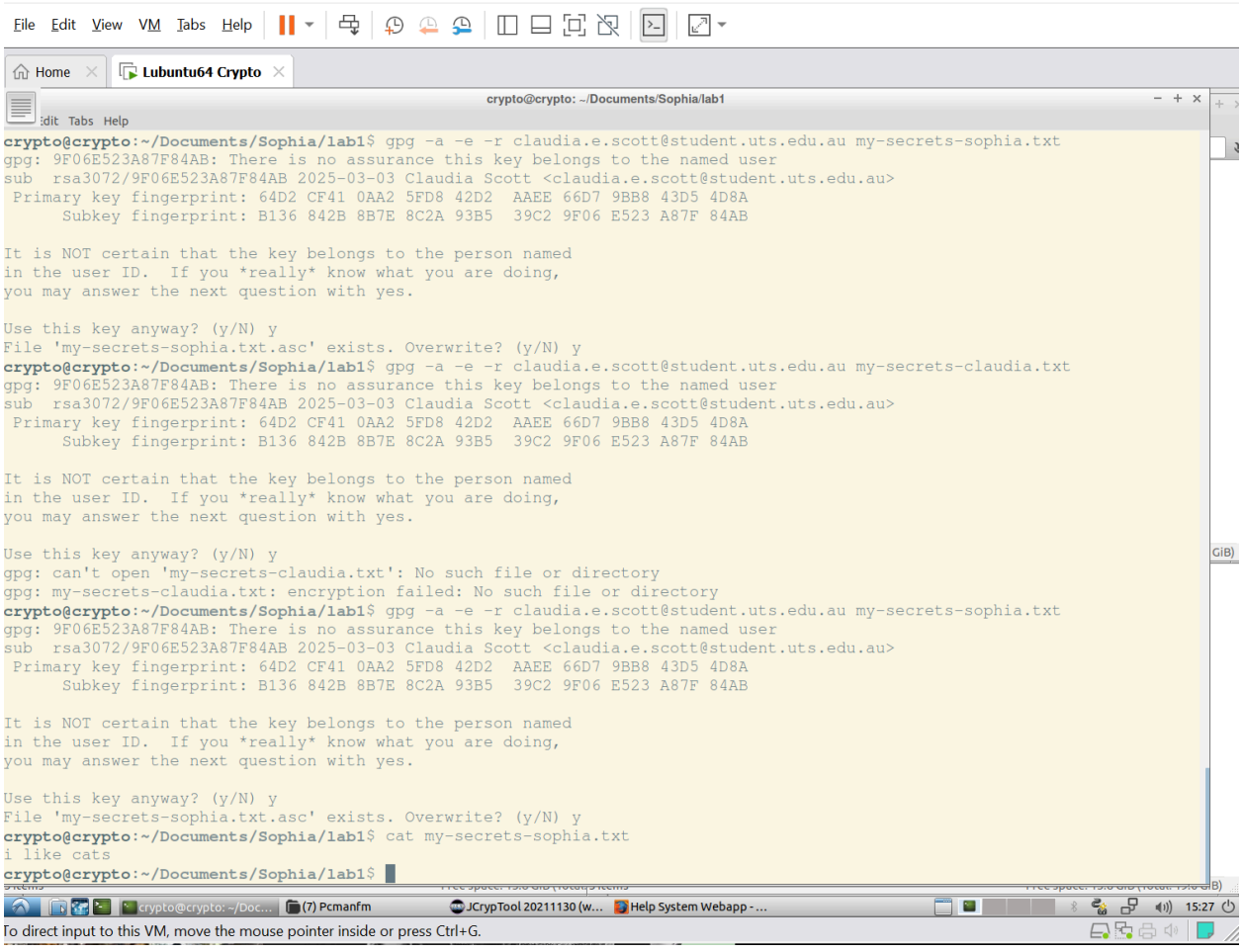


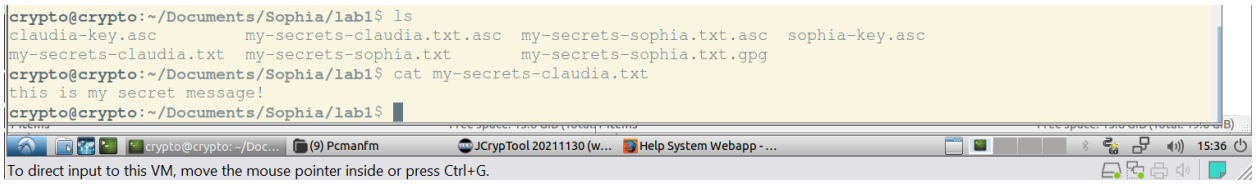





```
crypto@crypto:~/Documents/Sophia/lab1$ gpg --export -a --output sophia-key.asc sophia@gmail.com
File 'sophia-key.asc' exists. Overwrite? (y/N) y
crypto@crypto:~/Documents/Sophia/lab1$ ls
my-secrets-sophia.txt  my-secrets-sophia.txt.asc  my-secrets-sophia.txt.gpg  sophia-key.asc
crypto@crypto:~/Documents/Sophia/lab1$ --import claudia-key.asc
--import: command not found
crypto@crypto:~/Documents/Sophia/lab1$ --import claudia-key.asc
--import: command not found
crypto@crypto:~/Documents/Sophia/lab1$ gpg --import claudia-key.asc
gpg: key B0FDE4185A3719D5: public key "Claudia Scott <claudia.e.scott@student.uts.edu.au>" imported
gpg: key 66D79BB843D54D8A: public key "Claudia Scott <claudia.e.scott@student.uts.edu.au>" imported
gpg: Total number processed: 2
gpg:               imported: 2
crypto@crypto:~/Documents/Sophia/lab1$ gpg --list-keys
/home/crypto/.gnupg/pubring.kbx
------------------------------
pub   rsa3072 2025-03-03 [SC] [expires: 2027-03-03]
      FA948787A8F545417A28FAAFE2B2E98FE4DEE8BC
uid           [ultimate] Sophia <sophia@gmail.com>
sub   rsa3072 2025-03-03 [E] [expires: 2027-03-03]

pub   rsa3072 2025-03-03 [SC] [expires: 2027-03-03]
      F211BF8101E3CCC99956E756B0FDE4185A3719D5
uid           [ unknown] Claudia Scott <claudia.e.scott@student.uts.edu.au>
sub   rsa3072 2025-03-03 [E] [expires: 2027-03-03]

pub   rsa3072 2025-03-03 [SC] [expires: 2027-03-03]
      64D2CF410AA25FD842D2AAEE66D79BB843D54D8A
uid           [ unknown] Claudia Scott <claudia.e.scott@student.uts.edu.au>
sub   rsa3072 2025-03-03 [E] [expires: 2027-03-03]
```

```
crypto@crypto:~/Documents/Sophia/lab1$ pgp --list-secret-keys

Command 'pgp' not found, but can be installed with:

sudo apt install pgpgpg

crypto@crypto:~/Documents/Sophia/lab1$ gpg --list-secret-keys
/home/crypto/.gnupg/pubring.kbx
------------------------------
sec   rsa3072 2025-03-03 [SC] [expires: 2027-03-03]
      FA948787A8F545417A28FAAFE2B2E98FE4DEE8BC
uid           [ultimate] Sophia <sophia@gmail.com>
ssb   rsa3072 2025-03-03 [E] [expires: 2027-03-03]

crypto@crypto:~/Documents/Sophia/lab1$ gpg -a -e -r claudia.e.scott@student.uts.edu.au my-secrets-sophia.txt
gpg: 9F06E523A87F84AB: There is no assurance this key belongs to the named user
sub  rsa3072/9F06E523A87F84AB 2025-03-03 Claudia Scott <claudia.e.scott@student.uts.edu.au>
 Primary key fingerprint: 64D2 CF41 0AA2 5FD8 42D2  AAEE 66D7 9BB8 43D5 4D8A
      Subkey fingerprint: B136 842B 8B7E 8C2A 93B5  39C2 9F06 E523 A87F 84AB

It is NOT certain that the key belongs to the person named
in the user ID.  If you *really* know what you are doing,
you may answer the next question with yes.

Use this key anyway? (y/N) y
File 'my-secrets-sophia.txt.asc' exists. Overwrite? (y/N) y
crypto@crypto:~/Documents/Sophia/lab1$
```





Q: Try the following cases and summarise your observations

1. Select the wrong sender's private key. You may need to generate new private keys with different email addresses. You can remove a private key by running **gpg –delete-secret-key email@email.email**

- When selecting the wrong private key to decrypt or encrypt something it will fail because the wrong identity is being used.

2. The receiver has not imported the sender public key. You can remove an imported public key by running gpg –delete-key email@email.email

- By not importing the sender's public key, you won't be able to decrypt the encrypted message they sent

Q: There are two operations in PGP, i.e., PGP sign and PGP Encrypt. What is the difference between them? You may search online.

- Pretty Good Privacy, PGP sign- ensures the integrity and authenticity of a message through digital signatures, this legitimises a user's identity
- PGP encrypt- ensures the confidentiality of a message by securing the private key and making it accessible to intended recipients

Lab Summary and Discussion

- This lab is a basic introduction to cryptography, concepts such as decrypting and encrypting were practiced through the PGP activity. Part 3 was a bit confusing and needed further clarification.