

LAB ASSIGNMENT NO. 12

AIM: Explore the GPG tool of linux to implement email security.

LAB OUTCOME ATTAINED:

LO 6: Demonstrate the network security system using open source tools.

THEORY:

GPG, which stands for **GNU Privacy Guard**, is a free and open-source encryption software that provides cryptographic privacy and authentication for data communication. It's a modern replacement for PGP (Pretty Good Privacy) and is widely used for securing emails and files. GPG is available for various platforms, including Linux, macOS, and Windows. In GPG, the terms "private key ring" and "public key ring" refer to collections of cryptographic keys used for encryption and decryption:

Private Key Ring:

- The private key ring is a file or storage location where the sender's private keys are stored.
- Private keys are used by the sender for decrypting messages that were encrypted with the receiver's public key and for digitally signing messages or files.
- The sender should guard their private key(s) very carefully because anyone with access to the private key can decrypt their messages and files, impersonate them, and potentially compromise their security.

Public Key Ring:

- The public key ring is a file or storage location where the public keys of the receiver and other potential recipients are stored.
- Public keys are used by the sender for encrypting messages or files that they want to send securely to the receiver. The sender uses the receiver's public key to encrypt the data, and only the receiver can decrypt it with their private key.
- Additionally, public keys are used to verify digital signatures created by the sender or others. If the sender receives a digitally signed message or file, they can use the sender's public key to verify that it was indeed signed by them and hasn't been tampered with.

Commands used for Key Generation and Encryption/Decryption:

Step 1: Generate private key and public key pairs for sender and receiver using command

gpg --gen-key or gpg --full-generate-key (repeat for sender and receiver)

Step 2: Create a file containing sender's public key which then can be sent to other users.

gpg --export -a username > filename (creates file in ascii format) or
gpg --output filename --armor --export user's_email (for sender)

Step 3: Similarly create a file containing the sender's private key.

gpg --export-secret-key -a username > filename (for sender)

Step 4: You can create a fingerprint of key using the command
gpg --fingerprint receiver's_email (for receiver)

Step 5: Sender needs to add in his public key ring, the public key of receiver
(for sender)
gpg --import filename_containing_public_key_of_receiver

Step 6: Listing public keys in keyring
gpg --list-keys (from public key rings of all users)
gpg --list-keys emailid@gmail.com (from public key rings of specific users)

Step 7: Sender can sign the public key of receiver using command
gpg --sign-key receiver_email

Step 8: Encrypt the data to send.
gpg --encrypt -r receiver_email name_of_file
OR
gpg --encrypt --sign --armor -r receiver_email name_of_file
OR
gpg --encrypt --sign -r receiver_email name_of_file

Step 9: Decrypt the file
gpg -o myfiledecrypted -d myfile.txt.gpg

OUTPUT:

```
shreyakamath@LAPTOP-UEM x + v - □ ×
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 5.15.90.1-microsoft-standard-WSL2 x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/advantage

* Strictly confined Kubernetes makes edge and IoT secure. Learn how MicroK8s
just raised the bar for easy, resilient and secure K8s cluster deployment
.

https://ubuntu.com/engage/secure-kubernetes-at-the-edge

* Introducing Expanded Security Maintenance for Applications.
Receive updates to over 25,000 software packages with your
Ubuntu Pro subscription. Free for personal use.

https://ubuntu.com/pro

This message is shown once a day. To disable it please create the
/home/shreyakamath/.hushlogin file.
shreyakamath@LAPTOP-UEMD4SBH:~$ sudo apt-get install gpg
[sudo] password for shreyakamath:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
gpg is already the newest version (2.2.27-3ubuntu2.1).
gpg set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 141 not upgraded.
shreyakamath@LAPTOP-UEMD4SBH:~$
```

```
shreyakamath@LAPTOP-UEM x + v - □ ×
gpg: agent_genkey failed: Timeout
Key generation failed: Timeout
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --full-generate-key
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

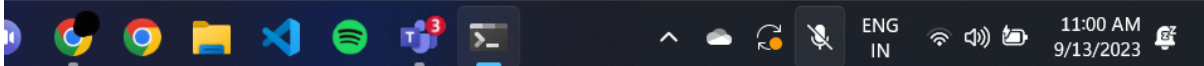
Please select what kind of key you want:
  (1) RSA and RSA (default)
  (2) DSA and Elgamal
  (3) DSA (sign only)
  (4) RSA (sign only)
  (14) Existing key from card
Your selection? 1
RSA keys may be between 1024 and 4096 bits long.
What keysize do you want? (3072) 1024
Requested keysize is 1024 bits
Please specify how long the key should be valid.
    0 = key does not expire
    <n> = key expires in n days
    <n>w = key expires in n weeks
    <n>m = key expires in n months
    <n>y = key expires in n years
Key is valid for? (0) 2
Key expires at Fri Sep 15 10:59:30 2023 IST
Is this correct? (y/N) y

GnuPG needs to construct a user ID to identify your key.

Real name: shreya
Email address: shreya@abc.com
Comment: sender
You selected this USER-ID:
    "shreya (sender) <shreya@abc.com>"

Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
```

```
shreyakamath@LAPTOP-UEM  ×  +  ▾  
    0 = key does not expire  
    <n> = key expires in n days  
    <n>w = key expires in n weeks  
    <n>m = key expires in n months  
    <n>y = key expires in n years  
Key is valid for? (0) 2  
Key expires at Fri Sep 15 10:59:30 2023 IST  
Is this correct? (y/N) y  
  
GnuPG needs to construct a user ID to identify your key.  
  
Real name: shreya  
Email address: shreya@abc.com  
Comment: sender  
You selected this USER-ID:  
    "shreya (sender) <shreya@abc.com>"  
  
Change (N)ame, (C)omment, (E)mail or (O)kay/(Q)uit? o  
We need to generate a lot of random bytes. It is a good idea to perform  
some other action (type on the keyboard, move the mouse, utilize the  
disks) during the prime generation; this gives the random number  
generator a better chance to gain enough entropy.  
We need to generate a lot of random bytes. It is a good idea to perform  
some other action (type on the keyboard, move the mouse, utilize the  
disks) during the prime generation; this gives the random number  
generator a better chance to gain enough entropy.  
gpg: /home/shreyakamath/.gnupg/trustdb.gpg: trustdb created  
gpg: key 49AB0F2E7B2F09E2 marked as ultimately trusted  
gpg: directory '/home/shreyakamath/.gnupg/openpgp-revocs.d' created  
gpg: revocation certificate stored as '/home/shreyakamath/.gnupg/openpgp-rev  
ocs.d/6F4A182836FD0B9399A49F0849AB0F2E7B2F09E2.rev'  
public and secret key created and signed.  
  
pub   rsa1024 2023-09-13 [SC] [expires: 2023-09-15]  
      6F4A182836FD0B9399A49F0849AB0F2E7B2F09E2  
uid           shreya (sender) <shreya@abc.com>  
sub   rsa1024 2023-09-13 [E] [expires: 2023-09-15]  
  
shreyakamath@LAPTOP-UEMD4SBH:~$ |
```



```
shreyakamath@LAPTOP-UEM × + ▾  
6F4A182836FD0B9399A49F0849AB0F2E7B2F09E2  
uid          shreya (sender) <shreya@abc.com>  
sub  rsa1024 2023-09-13 [E] [expires: 2023-09-15]  
  
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --gen-key  
gpg (GnuPG) 2.2.27; Copyright (C) 2021 Free Software Foundation, Inc.  
This is free software: you are free to change and redistribute it.  
There is NO WARRANTY, to the extent permitted by law.  
  
Note: Use "gpg --full-generate-key" for a full featured key generation dialo  
g.  
  
GnuPG needs to construct a user ID to identify your key.  
  
Real name: meetali  
Email address: meetali@abc.com  
You selected this USER-ID:  
  "meetali <meetali@abc.com>"  
  
Change (N)ame, (E)mail, or (O)kay/(Q)uit? o  
We need to generate a lot of random bytes. It is a good idea to perform  
some other action (type on the keyboard, move the mouse, utilize the  
disks) during the prime generation; this gives the random number  
generator a better chance to gain enough entropy.  
We need to generate a lot of random bytes. It is a good idea to perform  
some other action (type on the keyboard, move the mouse, utilize the  
disks) during the prime generation; this gives the random number  
generator a better chance to gain enough entropy.  
gpg: key 194A986FFA7551D5 marked as ultimately trusted  
gpg: revocation certificate stored as '/home/shreyakamath/.gnupg/openpgp-rev  
ocs.d/C5CC70220D8D8B99F49CCAD9194A986FFA7551D5.rev'  
public and secret key created and signed.  
  
pub  rsa3072 2023-09-13 [SC] [expires: 2025-09-12]  
     C5CC70220D8D8B99F49CCAD9194A986FFA7551D5  
uid          meetali <meetali@abc.com>  
sub  rsa3072 2023-09-13 [E] [expires: 2025-09-12]  
  
shreyakamath@LAPTOP-UEMD4SBH:~$ |
```

```
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --export -a shreya>shreyapublic
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --output meetalipublic --armor --export
meetali@abc.com
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --export-secret-key -a shreya>shreyaprivate
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --export-secret-key -a meetali>meetaliprivate
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --import meetalipublic
gpg: key 194A986FFA7551D5: "meetali <meetali@abc.com>" not changed
gpg: Total number processed: 1
gpg:      unchanged: 1
shreyakamath@LAPTOP-UEMD4SBH:~$ |
```

```
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --import meetalipublic
gpg: key 194A986FFA7551D5: "meetali <meetali@abc.com>" not changed
gpg: Total number processed: 1
gpg:      unchanged: 1
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --list-keys
gpg: checking the trustdb
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid: 2  signed: 0  trust: 0-, 0q, 0n, 0m, 0f, 2u
gpg: next trustdb check due at 2023-09-15
/home/shreyakamath/.gnupg/pubring.kbx
-----
pub   rsa1024 2023-09-13 [SC] [expires: 2023-09-15]
      6F4A182836FD0B9399A49F0849AB0F2E7B2F09E2
uid           [ultimate] shreya (sender) <shreya@abc.com>
sub   rsa1024 2023-09-13 [E] [expires: 2023-09-15]

pub   rsa3072 2023-09-13 [SC] [expires: 2025-09-12]
      C5CC70220D8D8B99F49CCAD9194A986FFA7551D5
uid           [ultimate] meetali <meetali@abc.com>
sub   rsa3072 2023-09-13 [E] [expires: 2025-09-12]

shreyakamath@LAPTOP-UEMD4SBH:~$ |
```

```
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --list-keys shreya@abc.com
pub   rsa1024 2023-09-13 [SC] [expires: 2023-09-15]
      6F4A182836FD0B9399A49F0849AB0F2E7B2F09E2
uid           [ultimate] shreya (sender) <shreya@abc.com>
sub   rsa1024 2023-09-13 [E] [expires: 2023-09-15]

shreyakamath@LAPTOP-UEMD4SBH:~$ gedit myfile
```

Roll no. : 53

Name: Shreya Kamath

Date: 13th September, 2023.

Open ▾

+

myfile
~/

Save

≡

×

1 This is a sample file

2

3 gpg experiment cns lab

4

5 13th september

6





7 <3

Plain Text ▾

Tab Width: 8 ▾

Ln 6, Col 1 ▾

INS







	meetaliprivate	9/13/2023 11:07 AM	File	6 KB
	meetalipublic	9/13/2023 11:06 AM	File	3 KB
	myfile	9/13/2023 11:14 AM	File	1 KB
	myfile.gpg	9/13/2023 11:16 AM	GPG File	1 KB


```
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --encrypt -r meetali@abc.com myfile
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --encrypt --sign --armor -r meetali@abc.
com
^C
gpg: signal 2 caught ... exiting

shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --encrypt --sign --armor -r meetali@abc.
com myfile
shreyakamath@LAPTOP-UEMD4SBH:~$ |
```

```
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --encrypt -r meetali@abc.com myfile
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --encrypt --sign --armor -r meetali@abc.
com
^C
gpg: signal 2 caught ... exiting

shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --encrypt --sign --armor -r meetali@abc.
com myfile
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg --encrypt --sign -r meetali@abc.com myfi
le
File 'myfile.gpg' exists. Overwrite? (y/N) y
shreyakamath@LAPTOP-UEMD4SBH:~$ gpg -o myfiledecrypted -d myfile.gpg
gpg: encrypted with 3072-bit RSA key, ID 7683BC5E819F5859, created 2023-09-1
3
      "meetali <meetali@abc.com>"
gpg: Signature made Wed Sep 13 11:21:20 2023 IST
gpg:      using RSA key 6F4A182836FD0B9399A49F0849AB0F2E7B2F09E2
gpg: Good signature from "shreya (sender) <shreya@abc.com>" [ultimate]
shreyakamath@LAPTOP-UEMD4SBH:~$ |
```

 meetaliprivate	9/13/2023 11:07 AM	File	6 KB
 meetalipublic	9/13/2023 11:06 AM	File	3 KB
 myfile	9/13/2023 11:14 AM	File	1 KB
 myfile.asc	9/13/2023 11:18 AM	ASC File	2 KB
 myfile.gpg	9/13/2023 11:21 AM	GPG File	1 KB
 myfiledecrypted	9/13/2023 11:23 AM	File	1 KB

CONCLUSION:

Hence, I have understood the basic concept and fundamentals of the GPG tool for email security purposes. I also executed related commands for encryption, decryption and key generation using the GPG tool.