Vivekanand Education Society's Institute of Technology

Department of AI & DS Engineering

Subject: Cryptography and System Security

Class: D11AD

Roll No: 46	**Name: Ashish**
Practical No:9	**Title:GPG Tool**
DOP:	**DOS:**
Grades:	**LOs Mapped:**
Signature:	

Title: GPG Tool
DOP: /3/24
DOS: /3/24

(Attach output screenshots)

Aim: To explore the GPG tool of linux to encrypt and decrypt file.

Theory:

gpg Commands

1.Generate Key Pair

```
$ gpg --gen-key
```

Above command will take you through series of questions like type of encryption (DSA, RSA), key size, key validity days, Real name, email address, Pass phrase, etc. and generate public and private key.

```
ash123@sheesh:/mnt/c/Users/ASHIS$ gpg --full-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.

gpg: directory '/home/ash123/.gnupg' created
gpg: keybox '/home/ash123/.gnupg/pubring.kbx' created
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) 2048
Requested keysize is 2048 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
```

2. Create revocation certificate

```
$ gpg --gen-revoke myname@mydomain.com
```

This is required to invalidate the key pair and should be created when key pairs are created.

```
ash123@sheesh:/mnt/c/Users/ASHIS$ gpg  --gen-revoke ashishp1729@gmail.com

sec  rsa2048/3A8EB64D6B84D9CA 2024-03-31 Ashtrobuff1 (first try) <ashishp1729@g

Comment: This is a revocation certificate

iQE2BCABCgAgFiEEUiTHEkJKvFfOhgfvOo62TWuE2coFAmYJumACHQAACgkQOo62
TWuE2cpMiAf9EtdjuqvhdBgFyGXuHvXpeE28iyOGeJHHtkm1u9vURpKJRZSH934c
lW6PVFBBJuIbRSmf6kpRU5ui0E9TcW9bdBn+DRHxKSUdGKah047NjitS60BSHX0v
RoHkRKxIUDt3nNRW1FOQdBGozWaIgVLMqIvqAksSJAl9lWQWtA6AtgrK1BUHCKkd
A7lBs9S2nCMXiLvTDZs36w/72LGz/TjQ+HrLRNk7xyerKv70wQf7tMRZjBTFD1IY
yuP2cSIHoBiyDVfd+BXneR4KirvlywcSv+TMrPc+8gGZGK82Ef2HjQ+Asu6OQ3eQ
voJqxiVv7NJJ3I9TRuVa606+L1eA2IqiNQ==
=7flQ
-----END PGP PUBLIC KEY BLOCK-----
Revocation certificate created.

Please move it to a medium which you can hide away; if Mallory gets
```

3. Import other's public key 3

```
$ gpg --import public_key_file
```

```
> gpg --import /home/ash123/.gnupg/openpgp-revocs.d/5AC28A105D7FD15EA8EAEA38D27EB24362F339A2.rev
```

4. Sign the key received from other person

```
$ gpg --sign-key yourname@yourdomain.com
```

```
ash123@sheesh:/mnt/c/Users/ASHIS$ gpg --import '/home/ash123/.gnupg/openpgp-revocs.d/5AC28A105
339A2.rev
>
> gpg --import /home/ash123/.gnupg/openpgp-revocs.d/5AC28A105D7FD15EA8EAEA38D27EB24362F339A2.
> gpg --sign-key ashishp1729@gmail.com
```

Signing the key means, you trust the key which has been given to you.

5. Send the signed key back to sender

```
$ gpg --export --armor yourname@yourdomain.com
```

```
> gpg --export --armor ashishp1729@gmail.com
```

6. Import the received signed key

```
$ gpg --import signed_key_file_name
```

7. List public keys

```
$ gpg --list-keys
```

```
ash123@sheesh:/mnt/c/Users/ASHIS$ 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 2026-03-31
/home/ash123/.gnupg/pubring.kbx
------------------------------
pub   rsa2048 2024-03-31 [SC]
      5224C712424ABC57CE8607EF3A8EB64D6B84D9CA
uid           [ultimate] Ashtrobuff1 (first try) <ashishp1729@gmail.com>
sub   rsa2048 2024-03-31 [E]

pub   rsa3072 2024-03-31 [SC] [expires: 2026-03-31]
```

8. View private keys

```
$ gpg --list-secret-keys
```

```
ash123@sheesh:/mnt/c/Users/ASHIS$ gpg --list-secret-keys
/home/ash123/.gnupg/pubring.kbx
------------------------------
sec   rsa2048 2024-03-31 [SC]
      5224C712424ABC57CE8607EF3A8EB64D6B84D9CA
uid           [ultimate] Ashtrobuff1 (first try) <ashishp1729@gmail.com>
ssb   rsa2048 2024-03-31 [E]

sec   rsa3072 2024-03-31 [SC] [expires: 2026-03-31]
      5AC28A105D7FD15EA8EAEA38D27EB24362F339A2
```

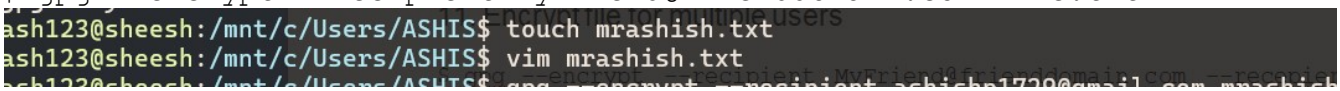
9. Update keys

```
$ gpg --refresh-keys
```

```
ash123@sheesh:/mnt/c/Users/ASHIS$ gpg --refresh-keys
gpg: refreshing 2 keys from hkps://keys.openpgp.org
```

10. Encrypt file for particular user 4

```
$ gpg --encrypt --recipient MyFriend@frienddomain.com file.txt
```

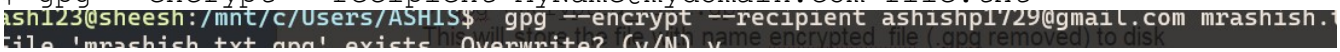


11. Encrypt file for multiple users

```
$ gpg --encrypt --recipient MyFriend@frienddomain.com --recepient
OneMoreFriend@frienddomain.com file.txt
```

12. Encrypt file for self

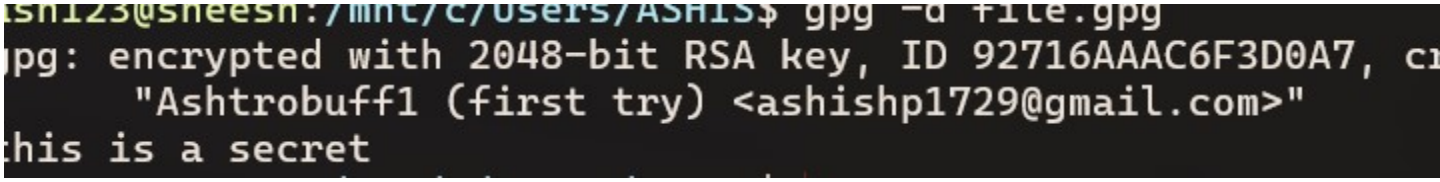
```
$ gpg --encrypt --recipient MyName@mydomain.com file.txt
```



13. Decrypt text file 5

```
$ gpg --decrypt encrypted_file.txt.gpg
```

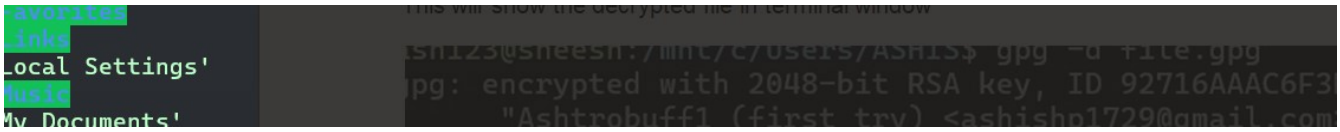
This will show the decrypted file in terminal window



14. Decrypt text/binary file

```
$ gpg encrypted_file.gpg
```

This will store the file with name encrypted_file (.gpg removed) to disk



Export Your Public Key 2

Others need your public key to send encrypted message to you and only your private key can decrypt it. Use the following command to export your public key. `--armor` option means that the output is ASCII armored. The default is to create the binary OpenPGP format. `user-id` is your email address.

```
gpg --armor --export user-id> pubkey.asc
```

```
ash123@sheesh:/mnt/c/Users/ASHIS$ gpg --armor --export ashishp1729@gmail.com pubkey.
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQENBGYJt9QBCAC/R07MqtRZ5yywU31Pr9MDG/3683m0C19z61LqHouE/ZFzY9y+
yUVHa9UzXKfOG8EDrGVF4N4B/83qX0kWK5cezxjYaluhGVWxv1pEh32o7rEzVvFb
1FmHUN/9yRlCgdMRTavfJ8awb52m8vi9srZ0yOjD0hICAs5pLA982rmHGwcfbXwj
UgU7VHPC/aa7KOTJkBCZmIh2VzKRRgnblhZbGFBIWCOfRAzZUfwSn54Ds3WzU22G
MSwWY1eHLDs8mTnGP/XYpoYkQE9tRyc7lm//Zz5ODSDTAMm3avn1dWVYm4jvMuHz
n79dkjPsXyFtBKIEcd+L0xEk7vm7HJ2PJRbPABEBAAG0L0FzaHRyb2J1ZmYxICHm
aXJzdCB0cnkpIDxhc2hpc2hwMTcyOUBnbWFpbC5jb20+iQFOBBMBCgA4FiEEUiTH
EkJKvFfOhgfvOo62TWuE2coFAmYJt9QCGwMFCwkIBwIGFQoJCAsCBBYCAwECHgEC
F4AACgkQOo62TWuE2cpQPAgAobmOX/o09zNp+KytoKp4pi2zricQcy1NEIA+qsoy
mKc8FeP2rkndHTjunjNC6Hg7vpZ2hMxQRGg8gIyvyhaZrjuxrT1GNddh0JXvbMY2
Nvnp6GxjvnngvfaT0abaMgcbQrm87ortOBIXqhmoE9Xqk7gics1p6pWkWDYs4EvA
h+jOpwX0dWJKoPp5YIbGQYiLaTwM7mpMDToh6gkfxNVD9EZfqp69qsZLTmWbAbyg
Nocq0NMNEtMjDj+9FaKwKh0JsTNk6R4LIQpfIQy87gp/pqFOdLEzG6QJ/f9Tg2fr
6u8nJMlB+cHEg7J3e9llDPVPgtI7P3VG3SADpL8LhX0S6LkBDQRmCbfUAQgAvfUn
DOAcAue+0y71NjEAFEm9+rszZx7e1nS9MWi20l3IsYESEV+4Y7ZlhsFDXl4FSCeM
FOYW9g2WBKHndRuMhW1nKIQECyuv2jOrNjVNMaWdS3IYmmq2SascEiRyTKR0NG0B
qRIK910evsTsVbEdxcjJjZ39ywXaJy6WOvjV3QRQHG/yDql3pRITapwrMxP7cXp7
8MuwKMjSneu7d8+xZCk8khVtdLHDKI6QxsqZCdYZDRxJ4x8MfTMjKJKL50YzwTWJ
XkRTEVwPx3Or2tHdz+xqC13aTsFiBmpM1y34+on43+SEHsskP0tFmYZAG3FHWxKr
p+hlkiYqK45GECZUjwARAQABiQE2BBgBCgAgFiEEUiTHEkJKvFfOhgfvOo62TWuE
2coFAmYJt9QCGwwACgkQOo62TWuE2cqJLwf/T9KFYl3QC37wnggbJ9t8XDtshYbU
IqxfxrVRtrKZ2Vuswc401KE8Kpihe3S0yXoLa5GVUumCnwlTUssH7QVCh8m3YPaV
```

The exported public key is written to `pubkey.asc` file.

Export Your Private Key

Issue the following command to export your private key.

```
gpg --export-secret-keys --armor user-id> privkey.asc
```

The exported key is written to `privkey.asc` file.

```
ash123@sheesh:/mnt/c/Users/ASHIS$ gpg --armor --export-secret-keys ashishp1729@gmail.com privkey.asc
-----BEGIN PGP PRIVATE KEY BLOCK-----

lQPGBGYJt9QBCAC/R07MqtRZ5yywU31Pr9MDG/3683m0C19z61LqHouE/ZFzY9y+
yUVHa9UzXKfOG8EDrGVF4N4B/83qX0kWK5cezxjYaluhGVWxv1pEh32o7rEzVvFb
1FmHUN/9yRlCgdMRTavfJ8awb52m8vi9srZ0yOjD0hICAs5pLA982rmHGwcfbXwj
UgU7VHPC/aa7KOTJkBCZmIh2VzKRRgnblhZbGFBIWCOfRAzZUfwSn54Ds3WzU22G
MSwWY1eHLDs8mTnGP/XYpoYkQE9tRyc7lm//Zz5ODSDTAMm3avn1dWVYm4jvMuHz
n79dkjPsXyFtBKIEcd+L0xEk7vm7HJ2PJRbPABEBAAH+BwMCiKfnRnYjZ8b/4oRa
GUO456dvnJBHkhJSPHgBTB0TY2rvAAXAjb/P/vd6tPcYmovPc3TPIJYgF77e0EfA
rMM0xc866+1afMyqRtfXbAHuXQqybJf8ojTfWc52AzwIPAg0Pm/WtcjM4Svw8Cp9
ZsmGgS5S28oOlmBNM66lWxnEIDzv1ZB6IHRuS2PXMng3kslzTbRBFXvr1KGDW/v1
E+5Dof6tEzWVacw36jfKUGU8LjVL7hzOD1Vur6PgxXUpGDklpf0xjgB3EEr9gZkv
CqWiTp0CySNmmv635zRqsTmxGNWKx0BmifqmNfrzfcYktp4oNoTB2yNdsfqussyd
mWpj4bBCPKQEug6s21TyESuPIHYxjSOeentoxoTUf/ZZpEPGCp4MCr0C6UYgNjqE
emaOvaz4JXDZWWjOrfPuDkHRa3d+hORbPVz9R/7Gq7sqfpsdfbklz8jJum9gnvoF
3BQoYG++OaUVBwNkv6PTDRL0WJzV2DG2cPTCwceC3OQNG44nZ/gviYjSCF46Qp11
KQq3CxMDlLoBnChsjuZrE4gemnfVzd/Z6TUUpveJSkks/tiN8la0m6HdnCs4E4NB
gLD99J1VDOzhK6DUjrXR+BIcF20LuTEd5sswEpHbCHbFWbWLKHS9LU34j4HlJkXd
YVkffqw+5/sPfOg9ar0bh1d+aouVY31uAb6EmzB7ggH4YXpzzdVF5MUcqxnVckD/
lhmMGph77X3x7l8aAUVy7pBn2hATzNHfJfvMY4SsfOywUCXuUwZOGne/DBl5obQR
g4zqXGdLSVr/aALv2+QWETk2fyVaVrYZG3ySieHywYzdg4/IbphilOLXS5eT7yc9
wvbkbq4XcfCTqLy6YlD8hRvHuroTT/Eud8sUZ5JXJWQITFkCg9LJNDABk5MVotfv
wJHKldQmRZxbtC9Bc2h0cm9idWZmMSAoZmlyc3QgdHJ5KSA8YXNoaXNocDE3MjlA
Z21haWwuY29tPokBTgQTAQoAOBYhBFIkxxJCSrxXzoYH7zqOtk1rhNnKBQJmCbfU
AhsDBQsJCAcCBhUKCQgLAgQWAgMBAh4BAheAAAoJEDqOtk1rhNnKUDwIAKG5jl/6
NPczafisraCqeKYts64nEHMtTRCAPqrKMpinPBXj9q5J3R047p4zQuh4O76WdoTM
```

Steps for experiment :

1. Generate key pair

```
$ gpg --gen-key
```

Above command will take you through series of questions like type of encryption (DSA, RSA), key size, key validity days, Real name, email address, Pass phrase, etc. and generate public and private key.

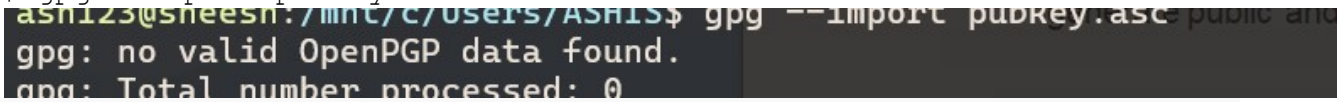
2. Export your public key

Others need your public key to send encrypted message to you and only your private key can decrypt it. Use the following command to export your public key. `--armor` option means that the output is ASCII armored. The default is to create the binary OpenPGP format. `user-id` is your email address.

```
gpg --armor --export user-id> pubkey.asc
```

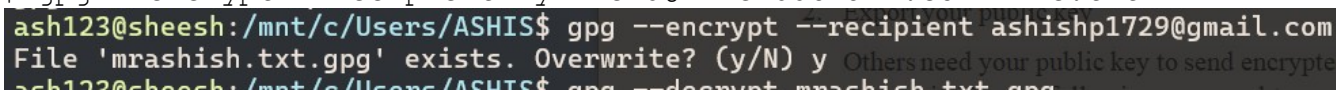
3. Send that file to your friend
4. Friend should import that file

```
$ gpg --import pubkey.asc
```

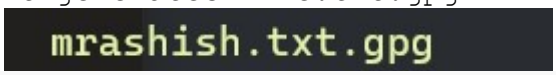


5. Friend should encrypt using public key

```
$ gpg --encrypt --recipient MyFriend@frienddomain.com file.txt
```

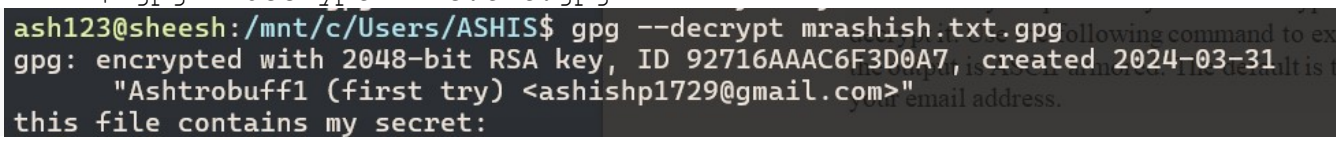



```
It generates file.txt.gpg
```



6. Friend should Send mail encrypted file
7. Upon receiving file decrypt that
8.

```
$ gpg --decrypt file.txt.gpg
```



Conclusion: We have successfully understood the gpg module on linux and made public and private keys to securely send encrypted data to each other .