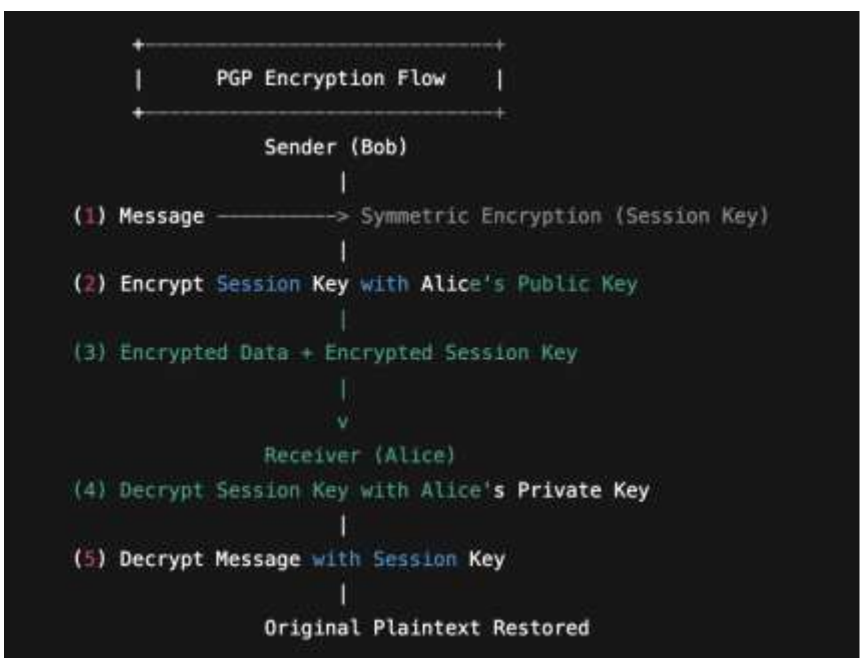
BHARATIYA VIDYA BHAVAN'S
SARDAR PATEL INSTITUTE OF TECHNOLOGY
Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai – 400058-India

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING
SUBJECT : CRYPTOGRAPHY AND SYSTEM SECURITY

NAME	**Atharva Tamhankar**
UID	**2022300127**

Experiment no. 6	
AIM:	**Implementing Pretty Good Privacy (PGP)**
OBJECTIVE:	**To implement and demonstrate the process of key generation, encryption–decryption, and digital signing–verification using PGP to ensure confidentiality, authentication, and integrity in data exchange.**
THEORY:	**Pretty Good Privacy (PGP) is a hybrid cryptographic system that combines symmetric encryption (for fast data encryption) and asymmetric encryption (for secure key exchange).** **Each user has a public key (shared) and a private key (kept secret).** **Messages are encrypted with the recipient's public key and decrypted with their private key.** **For authenticity, the sender signs the message using their private key, and the recipient verifies it using the sender's public key.** **PGP also supports digital signatures, key revocation, and trust verification via key fingerprints.**



CODE AND CELL OUTPUT:

```
!apt-get install gnupg -y
!gpg --version
```

1. Generate a new key pair And revocation certificate:

```
# 1. Create Alice's key

%%bash

cat > alice_key_params <<EOF

%no-protection

Key-Type: RSA

Key-Length: 2048

Name-Real: Alice Example

Name-Email: alice@example.com

Expire-Date: 0

%commit

EOF

gpg --batch --generate-key alice_key_params
```

```
gpg: directory '/root/.gnupg' created
gpg: keybox '/root/.gnupg/pubring.kbx' created
gpg: /root/.gnupg/trustdb.gpg: trustdb created
gpg: key FB953AB07E4E314A marked as ultimately trusted
gpg: directory '/root/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/root/.gnupg/openpgp-revocs.d/F5743B3298956E6AD035C72AFB953AB07E4E314A.rev'
```

```
# 2. Create Bob's key
%%bash
cat > bob_key_params <<EOF
%no-protection
Key-Type: RSA
Key-Length: 2048
Name-Real: Bob Example
Name-Email: bob@example.com
Expire-Date: 0
%commit
```

```
EOF
gpg --batch --generate-key bob_key_params
```

```
gpg: key 9F72224C7077D4F3 marked as ultimately trusted
gpg: revocation certificate stored as '/root/.gnupg/openpgp-revocs.d/E51840D9E2B8AF439CF7ECD89F72224C7077D4F3.rev'
```

2. Export/Share your public key:

```
# 3. Export & import public keys
%%bash
gpg --armor --export alice@example.com > alice_pub.asc
gpg --armor --export bob@example.com > bob_pub.asc
gpg --import alice_pub.asc
gpg --import bob_pub.asc
```

```
gpg: key FB953AB07E4E314A: "Alice Example <alice@example.com>" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
gpg: key 9F72224C7077D4F3: "Bob Example <bob@example.com>" not changed
gpg: Total number processed: 1
gpg:              unchanged: 1
```

3. Encrypt a file for a recipient:

```
# 4. Encrypt (Bob → Alice)
%%bash
echo "Hi Alice, this is a secret message" > message.txt
gpg --encrypt --recipient alice@example.com --armor -o
message_to_alice.asc message.txt
```

```
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
```

4. Decrypt the file:

```
# 5. Decrypt (Alice)
%%bash
gpg --output decrypted.txt --decrypt message_to_alice.asc
cat decrypted.txt
```

```
Hi Alice, this is a secret message
gpg: encrypted with 2048-bit RSA key, ID FB953AB07E4E314A, created 2025-10-17
      "Alice Example <alice@example.com>"
```

5. Sign a file and Verify the Signature:

```
# 6. Sign + Verify example
%%bash
echo "Report content" > report.pdf
gpg --output report.sig --detach-sign report.pdf
gpg --verify report.sig report.pdf
```

```
gpg: Signature made Fri 17 Oct 2025 05:57:47 PM UTC
gpg:                using RSA key F5743B3298956E6AD035C72AFB953AB07E4E314A
gpg: Good signature from "Alice Example <alice@example.com>" [ultimate]
```

6. Good Verification + Bad (Tampered) Verification:

```
%%bash
# allow commands to fail without raising an exception
set +e

# show good verification first (untampered)
cp report_original.pdf report.pdf
gpg --verify report.sig report.pdf
echo "gpg exit code (good): $?"

# now tamper and show BAD verification
echo "tampered!" >> report.pdf
gpg --verify report.sig report.pdf  # will print BAD
signature
echo "gpg exit code (bad): $?"
```

```
gpg exit code (good): 1
gpg exit code (bad): 1
cp: cannot stat 'report_original.pdf': No such file or directory
gpg: Signature made Fri 17 Oct 2025 05:57:47 PM UTC
gpg:                using RSA key F5743B3298956E6AD035C72AFB953AB07E4E314A
gpg: BAD signature from "Alice Example <alice@example.com>" [ultimate]
gpg: Signature made Fri 17 Oct 2025 05:57:47 PM UTC
gpg:                using RSA key F5743B3298956E6AD035C72AFB953AB07E4E314A
gpg: BAD signature from "Alice Example <alice@example.com>" [ultimate]
```

7. Revoke a key:

```
# Install expect (needed only once)
%%bash
```

```
apt-get update > /dev/null
apt-get install expect -y > /dev/null
%%bash
cat > new_key_params <<EOF
%no-protection
Key-Type: RSA
Key-Length: 2048
Name-Real: Charlie Example
Name-Email: charlie@example.com
Expire-Date: 0
%commit
EOF
gpg --batch --generate-key new_key_params
```

```
gpg: key 6EF419122B9E3417 marked as ultimately trusted
gpg: revocation certificate stored as '/root/.gnupg/openpgp-revocs.d/C8730C1FC020026E47B2E1806EF419122B9E3417.rev'
```

Original Msg and Decrypted Text:

Original Text:

```
message.txt ×    decrypted.txt
1 Hi Alice, this is a secret message
2
```

Decrypted Text:

```
message.txt    decrypted.txt ×
1 Hi Alice, this is a secret message
2
```

CONCLUSION: **This experiment successfully demonstrates the complete PGP workflow — including key generation, public key exchange, encryption/decryption, and digital signature verification.**

	The use of hybrid encryption ensures both speed (through symmetric encryption) and security (through asymmetric key exchange). The digital signature provides message integrity and non-repudiation, while the revocation certificate ensures key lifecycle management. The verification of both valid and tampered messages confirms that PGP effectively detects unauthorized modification and maintains communication authenticity. Overall, PGP remains a robust and widely used standard for securing electronic communication.

Submit on Moodle before the Deadline and in PDF format only.