Experiment: Implementing and Analyzing PGP for Secure Email Communicati

Objective:

- To understand how PGP encryption and decryption work.
- To learn how to generate PGP keys, sign and verify messages, and encrypt/decrypt emails.
- To explore the security mechanisms of PGP such as digital signatures and encryption.

Prerequisites:

- Basic knowledge of cryptography concepts (public key, private key, digital signatures).
- Understanding of asymmetric encryption and hashing.
- A computer with a PGP software tool (e.g., GnuPG, Kleopatra, or an email client with PGP support like Thunderbird with Enigmail plugin).

Required Tools/Software:

1. GnuPG (GPG): A free and open-source implementation of the OpenPGP standard.
2. Email Client: Mozilla Thunderbird with Enigmail, or another client that supports PGP.
3. Virtual Lab Setup (optional): If on a lab network, setting up two or more virtual machines to simulate communication between users.

Step-by-Step Procedure:

Part 1: PGP Key Generation

1. Install PGP Tool: Install GnuPG on your system or use a tool like Kleopatra or Thunderbird Enigmail for managing keys.
2. Generate a Key Pair:
   - Use GnuPG or your tool's interface to generate a PGP key pair.

- Set a passphrase to protect your private key.
- The key generation command for GnuPG is: gpg --full-generate-key
- Choose the encryption algorithm (e.g., RSA, 2048-bit or higher).

3. Export the Public Key:
   - After key generation, export the public key for sharing:

     ```
     gpg --armor --export your_email@example.com
     ```

   - Save this public key in a text file (e.g., public_key.txt) for later use.

Part 2: Sending an Encrypted Email

1. Setup Email Client: Configure Thunderbird with Enigmail or another email client for PGP use.
2. Send Public Keys: Exchange public keys with your experiment partner by emailing the public key as an attachment.
3. Encrypt and Send Email:
   - Import your partner's public key using GnuPG:

     ```
     gpg --import partner_public_key.txt
     ```

   - Encrypt a message using the recipient's public key:

     ```
     gpg --encrypt --armor -r recipient_email@example.com message.txt
     ```

   - This will generate an encrypted file (e.g., encrypted_message.txt) that you can email to your partner.

Part 3: Receiving and Decrypting an Email

1. Receive Encrypted Email:
   - Download the encrypted message sent by your partner.
2. Decrypt the Message:
   - Use GnuPG or the email client to decrypt the message. The GPG command for decryption is: gpg --decrypt encrypted_message.txt

- Provide the passphrase for your private key to successfully decrypt the message.

Part 4: Signing and Verifying Emails

1. Sign a Message:
   - Write a new message and sign it with your private key to ensure authenticity.

     ```
     gpg --armor --clearsign message.txt
     ```

   - This will create a message.asc file with your digital signature appended to the message.
2. Send the Signed Message:
   - Email the signed message to your partner.
3. Verify the Signature:
   - Your partner will verify the signature using your public key:

     ```
     gpg --verify message.asc
     ```

   - If the signature is valid, the recipient will see that the message was truly sent by you and wasn't tampered with.

Analysis:

1. Encryption and Decryption:
   - Discuss how PGP ensures confidentiality by using asymmetric encryption, where the sender encrypts a message with the recipient's public key, and the recipient decrypts it using their private key.
2. Digital Signatures:
   - Explain how the digital signature provides integrity and non-repudiation.
3. Key Management:
   - Discuss the importance of managing private keys securely and the implications of a compromised private key.
4. Security Features:

- Highlight the security benefits of using PGP for emails, including encryption, authentication, integrity, and non-repudiation.

Report/Documentation:

- Introduction: Briefly explain the purpose of PGP.
- Procedure: Provide detailed steps for generating keys, encrypting/decrypting messages, and signing/verifying emails.
- Results: Document the output for each command.
- Discussion: Analyze results, discuss security implications, and evaluate PGP in practical use.
- Conclusion: Summarize the importance of PGP.

Extension Activities:

- Group Key Management: Simulate group communication with multiple parties.
- PGP in Modern Applications: Explore integration of PGP in file encryption and data sharing.