

PRACTICAL 3

Aim :- Performing an Diffie Hellman Key Encryption

Software :- Google Collab / Jupyter Notebook

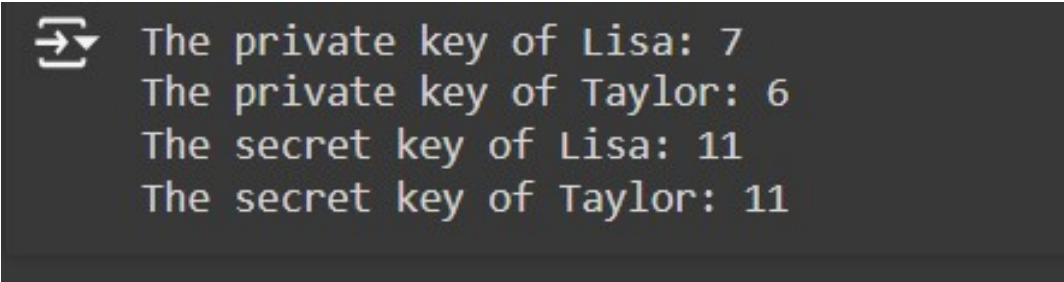
Description –

- Diffie-Hellman is an asymmetric key exchange algorithm.
- Allows two parties to securely generate a shared secret key over an unsecured channel.
- Each party generates a private key and a corresponding public key.
- Public keys are exchanged, and each party computes the shared secret using their private key and the other party's public key.
- The shared key is used for symmetric encryption (e.g., AES) in subsequent communication.
- Commonly used in secure protocols like TLS/SSL.

Code:-

```
from random import randint
p = 37
g = 5
a = 7
b = 6
x = int(pow(g, a, p))
y = int(pow(g, b, p))
ka = int(pow(y, a, p))
kb = int(pow(x, b, p))
print("The private key of Lisa:", a)
print("The private key of Taylor:", b)
print("The secret key of Lisa:", ka)
print("The secret key of Taylor:", kb)
```

Output:-



```
➞ The private key of Lisa: 7
   The private key of Taylor: 6
   The secret key of Lisa: 11
   The secret key of Taylor: 11
```

PRACTICAL 4

Aim :- Performing an AES Encryption and Decryption

Software :- Google Collab / Jupyter Notebook

Description – Here are some brief Descriptions of AES

- AES (Advanced Encryption Standard) is a symmetric encryption algorithm.
- Encrypts data (plaintext) into ciphertext using a secret key.
- Decryption uses the same key to restore the original data.
- Supports key sizes of 128, 192, or 256 bits.
- Widely used in secure communications, data storage, and VPNs.
- Known for strong encryption and high performance.

Code:-

```
from cryptography.fernet import Fernet
key=Fernet.generate_key()
cipher_suite=Fernet(key)
plaintext=b"WE ARE IN DANGER"
cipher_text=cipher_suite.encrypt(plaintext)
decrypted_text=cipher_suite.decrypt(cipher_text)
print("Original Message:",plaintext.deCode('utf-8'))
print("Encrypted Message:",cipher_text.deCode('utf-8'))
print("Decrypted Message:",decrypted_text.deCode('utf-8'))
```

Output:-

```
Original Message: WE ARE IN DANGER
Encrypted Message: gAAAAABm5tm5GFpEBRcUhdR2_G67dzj5HC0RME9djFZ5GqNObOHIsh1ZC5HtY42msmVx21cZfLtm23xACDLGoJMhUcRZLxZBgG7_yuQJCtRe6qVnqrwFEdu=
Decrypted Message: WE ARE IN DANGER
```

PRACTICAL 5

Aim :- Implement RSA Algorithm

Software :- Google Collab /Jupyter Notebook

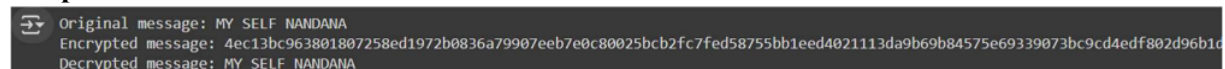
Description –

- RSA is an asymmetric encryption algorithm.
- Involves generating a public-private key pair for secure data transmission.
- A public key (e, n) is used for encryption, and a private key (d, n) is used for decryption.
- The sender encrypts the message using the recipient's public key.
- The recipient decrypts the message using their private key.
- RSA's security is based on the difficulty of factoring large prime numbers.
- Commonly used for secure data exchange and digital signatures in protocols like SSL/TLS.

Code:-

```
import rsa
# Generate a new public and private key pair with a key size of 2048 bits
(public_key, private_key) = rsa.newkeys(2048)
# Define the plaintext message as a bytes object
plaintext = b'MY SELF NANDANA'
# Encrypt the plaintext message using the public key
ciphertext = rsa.encrypt(plaintext, public_key)
# Decrypt the ciphertext using the private key
decrypted_message = rsa.decrypt(ciphertext, private_key)
# Print the original, encrypted, and decrypted messages
print('Original message:', plaintext.decode('utf-8'))
print('Encrypted message:', ciphertext.hex())
print('Decrypted message:', decrypted_message.decode('utf-8'))
```

Output:-



```
Original message: MY SELF NANDANA
Encrypted message: 4ec13bc963801807258ed1972b0836a79907eeb7e0c80025bcb2fc7fed58755bb1eed4021113da9b69b84575e69339073bc9cd4edf802d96b1d
Decrypted message: MY SELF NANDANA
```