St. Francis Institute of Technology, Mumbai-400 103
Department Of Information Technology

A.Y. 2025-2026
Class: BE-IT A/B, Semester: VII

Subject: Secure Application Development Lab

Student Name: **Keith Fernandes** Student Roll No: **25**

Experiment – 9 : Study and Implement Symmetric and Asymmetric Encryption Algorithm

Aim: To study and Implement symmetric and asymmetric encryption algorithm
a. Symmetric Key Cryptography (Data Encryption Standard)
b. Asymmetric Key Cryptography (Diffie Hellman)

Objectives: After study of this experiment, the student will be able to

- understand what is cryptography
- understand the concept of symmetric and asymmetric cryptography
- understand where encryption/decryption standards are used

Lab objective mapped: ITL703.6 To **apply** secure coding for cryptography

Prerequisite: Basic knowledge of symmetric and asymmetric key cryptography.

Requirements: C/C++/JAVA/PYTHON

Pre-Experiment Theory:

(a) **Symmetric Key Cryptography**

The Data Encryption Standard (DES) is a symmetric key block cipher algorithm used for data encryption and decryption. It was widely used in the past but is now considered outdated and insecure due to its relatively short key length (56 bits). Nonetheless, understanding how DES works can provide insight into the historical development of encryption algorithms.

Key components of the DES algorithm:

Key Generation:

- The original DES uses a 56-bit key, but only 64 bits are used for the key schedule, with the remaining 8 bits being used for error checking and parity.
- The 56-bit key undergoes a permutation and shifting process to produce 16 subkeys, each of 48 bits in length.

Encryption Process:

- DES operates on 64-bit blocks of plaintext and ciphertext.
- The initial step involves permuting the 64-bit plaintext using a fixed permutation table (initial permutation, IP).
- The 64-bit block is then divided into two halves, each 32 bits in size: the left half (L0) and the right half (R0).

- The core of the DES algorithm consists of 16 rounds, where each round uses the corresponding subkey generated during the key schedule.
- In each round, the right half of the data from the previous round (Ri-1) is expanded to 48 bits using an expansion permutation (E-bit selection).
- The expanded right half is then XORed with the current subkey (Ki) to introduce key mixing.
- The XOR result is passed through a series of eight substitution boxes (S-boxes) to perform nonlinear transformations.
- After S-box substitution, a permutation (P-box) is applied to the result.
- The output is then XORed with the left half of the data from the previous round (Li-1).
- The left and right halves are swapped, and the next round begins.

Decryption Process:

- The decryption process in DES is the same as encryption, but the subkeys are used in reverse order.

Final Permutation:

- After completing all 16 rounds, the two halves are combined and undergo a final permutation (inverse initial permutation, IP^-1) to produce the final 64-bit ciphertext.

(b) **Asymmetric Key Cryptography**

Diffie Hellman key exchange is a cryptographic protocol that allows two parties that have no prior knowledge of each other to jointly establish a shared secret key over an insecure communications channel. This key can then be used to encrypt subsequent communications using a symmetric key cipher.

Algorithm:

Alice and Bob, two users who wish to establish secure communications. We can assume that Alice and Bob know nothing about each other but are in contact.

1. Communicating in the clear, Alice and Bob agree on two large positive integers, p and g, where p is a prime number and g is a primitive root mod p.
2. Alice randomly chooses another large positive integer, XA, which is smaller than p. XA will serve as Alice's private key.
3. Bob similarly chooses his own private key, XB.
4. Alice computes her public key, YA, using the formula YA = (g^XA) mod p.
5. Bob similarly computes his public key, YB, using the formula YB = (g^XB) mod p.
6. Alice and Bob exchange public keys over the insecure circuit.
7. Alice computes the shared secret key, k, using the formula k = (YB ^XA) mod p.
8. Bob computes the same shared secret key, k, using the formula k = (YA ^XB) mod p.
9. Alice and Bob communicate using the symmetric algorithm of their choice and the shared secret key, k, which was never transmitted over the insecure circuit.

Procedure:

Insert following codes with output

a. Symmetric Key Cryptography (Data Encryption Standard)

CODE :

```
# Install pycryptodome if not already installed
!pip install pycryptodome

from Crypto.Cipher import DES
from Crypto.Random import get_random_bytes
import binascii

# --- Padding for DES block size ---
def pad(text):
    pad_len = 8 - (len(text) % 8)
    return text + chr(pad_len) * pad_len

def unpad(text):
    pad_len = ord(text[-1])
    return text[:-pad_len]

# --- Encrypt message using DES ---
def des_encrypt(plain_text, key):
    des = DES.new(key, DES.MODE_ECB)
    padded_text = pad(plain_text)
    encrypted_bytes = des.encrypt(padded_text.encode())
    return encrypted_bytes

# --- Decrypt message using DES ---
def des_decrypt(encrypted_bytes, key):
    des = DES.new(key, DES.MODE_ECB)
    decrypted_text = des.decrypt(encrypted_bytes).decode()
    return unpad(decrypted_text)

# --- Main DES Functionality ---
def run_des():
    key = get_random_bytes(8)  # 8-byte DES key
    text = "GPTDES123"

    print(f"\nOriginal Text: {text}")
    print(f"Key (hex): {binascii.hexlify(key).decode()}")

    encrypted = des_encrypt(text, key)
    print("Encrypted (hex):", binascii.hexlify(encrypted).decode())

    decrypted = des_decrypt(encrypted, key)
    print("Decrypted Text:", decrypted)

run_des()
```

```
Requirement already satisfied: pycryptodome in /usr/local/lib/python3.11/dist-packages (3.23.0)

Original Text: Hello KEITH
Key (hex): d0fbb0455594f6f7
Encrypted (hex): da47cfb08a69a0d2e5998ccc69945854
Decrypted Text: Hello KEITH
```

b. Asymmetric Key Cryptography (Diffie Hellman)

CODE :

```
import random

# --- Generate prime number and primitive root ---
P = 353  # A large prime number
G = 3    # A primitive root modulo P

# Alice's private and public keys
a_private = random.randint(2, P-2)
a_public = pow(G, a_private, P)

# Bob's private and public keys
b_private = random.randint(2, P-2)
b_public = pow(G, b_private, P)

# Shared secret computation
shared_secret_alice = pow(b_public, a_private, P)
shared_secret_bob = pow(a_public, b_private, P)

# --- Display results ---
print(f"\nPublic Parameters:\nP = {P}, G = {G}")
print(f"\nAlice's Private Key: {a_private}")
print(f"Alice's Public Key: {a_public}")

print(f"\nBob's Private Key: {b_private}")
print(f"Bob's Public Key: {b_public}")

print(f"\nShared Secret (computed by Alice): {shared_secret_alice}")
print(f"Shared Secret (computed by Bob):   {shared_secret_bob}")

if shared_secret_alice == shared_secret_bob:
   print("\n Shared secret successfully exchanged!")
else:
   print("\n Shared secret mismatch!")
```

```
Public Parameters:
P = 353, G = 3

Alice's Private Key: 277
Alice's Public Key: 290

Bob's Private Key: 267
Bob's Public Key: 75

Shared Secret (computed by Alice): 129
Shared Secret (computed by Bob):   129

 Shared secret successfully exchanged!
```

Post-Experiments Exercise:

Extended Theory: Nil

Post Experimental Exercise:

Questions:

- Compare symmetric and asymmetric encryption algorithms.
- Discuss the security of DES in the modern context, considering the advancements in cryptanalysis techniques and computing power.
- Discuss real-world applications of the Diffie-Hellman algorithm in various cryptographic protocols, such as secure key exchange for SSL/TLS in web browsers or VPN connections.

Conclusion:

- Write what was performed in the experiment.
- Write the significance of the topic studied in the experiment.

References:

http://cse29-iiith.vlabs.ac.in/