Shri Vile Parle Kelavani Mandal's

DWARKADAS J. SANGHVI COLLEGE OF ENGINEERING

(Autonomous College Affiliated to the University of Mumbai)

NAAC Accredited with "A" Grade (CGPA : 3.18)

DEPARTMENT OF INFORMATION TECHNOLOGY

COURSE CODE: DJS22ITL504 **DATE:** 24/09/2024

COURSE NAME: Cryptography and Network Security Laboratory **CLASS:** TYBTech

EXPERIMENT NO. 5

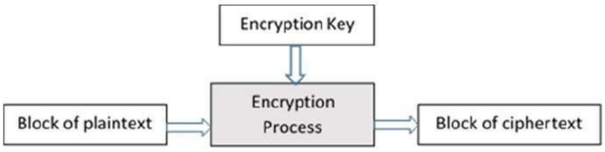
CO/LO: Design secure system using appropriate security mechanism

AIM / OBJECTIVE:

Analysis of Modern Block Ciphers (use crypt APIs)

DESCRIPTION OF EXPERIMENT:

A block cipher takes a block of plaintext bits and generates a block of ciphertext bits, generally of same size. The size of block is fixed in the given scheme. The choice of block size does not directly affect to the strength of encryption scheme. The strength of cipher depends up on the key length.



Analysis:

1. Use crypt API to encrypt/decrypt a plaintext block using AES, DES
2. Avalanche Effect: Change in Plaintext
3. Avalanche Effect: Change in key

SOURCE CODE:

```
pip install pycryptodome

from Crypto.Cipher import AES, DES
from Crypto.Util.Padding import pad, unpad
import binascii
import time

# AES encryption/decryption
def aes_encrypt_decrypt(plaintext, key):
    # Ensure the key is 16 bytes long for AES-128
    key = key.ljust(16, ' ')[:16].encode('utf-8')
    cipher = AES.new(key, AES.MODE_ECB)

    # Pad plaintext to be a multiple of block size (16 bytes for AES)
    padded_text = pad(plaintext.encode('utf-8'), 16)

    # Encrypt and measure encryption time
    start_time = time.time()
    ciphertext = cipher.encrypt(padded_text)
    encryption_time = time.time() - start_time
    print("AES Ciphertext:", binascii.hexlify(ciphertext).decode('utf-8'))
    print("AES Encryption Time: {:.6f} seconds".format(encryption_time))

    # Decrypt and measure decryption time
    start_time = time.time()
    decrypted_text = unpad(cipher.decrypt(ciphertext), 16)
    decryption_time = time.time() - start_time
    print("AES Decrypted Text:", decrypted_text.decode('utf-8'))
    print("AES Decryption Time: {:.6f} seconds".format(decryption_time))

# DES encryption/decryption
def des_encrypt_decrypt(plaintext, key):
    # Ensure the key is 8 bytes long for DES
    key = key.ljust(8, ' ')[:8].encode('utf-8')
    cipher = DES.new(key, DES.MODE_ECB)

    # Pad plaintext to be a multiple of block size (8 bytes for DES)
    padded_text = pad(plaintext.encode('utf-8'), 8)

    # Encrypt and measure encryption time
    start_time = time.time()
    ciphertext = cipher.encrypt(padded_text)
    encryption_time = time.time() - start_time
    print("DES Ciphertext:", binascii.hexlify(ciphertext).decode('utf-8'))
    print("DES Encryption Time: {:.6f} seconds".format(encryption_time))

    # Decrypt and measure decryption time
    start_time = time.time()
    decrypted_text = unpad(cipher.decrypt(ciphertext), 8)
    decryption_time = time.time() - start_time
    print("DES Decrypted Text:", decrypted_text.decode('utf-8'))
    print("DES Decryption Time: {:.6f} seconds".format(decryption_time))

# Test with plaintext and keys
plaintext = "123456"
aes_key = "122"
des_key = "122"

print("AES Encryption/Decryption:")
aes_encrypt_decrypt(plaintext, aes_key)

print("\nDES Encryption/Decryption:")
des_encrypt_decrypt(plaintext, des_key)
```

OBSERVATIONS AND CONCLUSION:

	AES-128	**DES**
plaintext	123456	123456
ciphertext	1a51af062d20914c80f2fd5d258e02ea	0d9f7cc175816137
Encryption time	0.000000 seconds	0.000000 seconds
Decryption time	0.000000 seconds	0.000210 seconds

Avalanche effect (1 bit change in plaintext)	AES	DES
Original plaintext	123456	123456
ciphertext	1a51af062d20914c80f2fd5d258e02ea	0d9f7cc175816137
Changed plaintext	123454	123454
New CT	a64ef772d4b3d6818741c71987e393dc	b50282e41779b491
No of bits changed	65	36

Avalanche effect (1 bit change in key)	AES	DES
Original plaintext	123456	123456
Original Key	123	123
ciphertext	1a51af062d20914c80f2fd5d258e02ea	0d9f7cc175816137
Changed key	122	122
New CT	516a676a6dbafdbf592610e368a9fae6	0d9f7cc175816137
No of bits changed	67	0

Q1) Based on amount of time taken for encryption/decryption comment wrt to performance.
Ans) AES has a better performance compared to DES in terms of decryption time since it took significantly less time for decryption. Both algorithms have the same encryption time in this case.

Q2) Which algo exhibits better avalanche effect wrt to change in plaintext?
Ans) AES: A 1-bit change in the plaintext resulted in 65 bits changing in the ciphertext.
DES: A 1-bit change in the plaintext resulted in 36 bits changing in the ciphertext.

AES exhibits a better avalanche effect with respect to changes in plaintext, as it causes a greater number of bits to change in the ciphertext.

Q3) Which algo exhibits better avalanche effect wrt to change in key?
Ans) AES: A 1-bit change in the key resulted in 67 bits changing in the ciphertext.
DES: A 1-bit change in the key resulted in 0 bits changing in the ciphertext.

AES demonstrates a much stronger avalanche effect with respect to key changes, as it causes significant changes in the ciphertext, whereas DES does not exhibit any change in the ciphertext.