

CRYPTOGRAPHY AND NETWORK SECURITY

LAB 6: IMPLEMENTATION OF DES **ENCRYPTION USING SUITABLE** **PROGRAMMING LANGUAGE**

NAME : OM SUBRATO DEY

REGISTER NO.: 21BAI1876

CODE:

```
from Crypto.Cipher import DES
from Crypto.Util.Padding import pad, unpad
def des_encrypt(key, plaintext):
    # Create a DES cipher object
    cipher = DES.new(key.encode('utf-8'), DES.MODE_CBC)
    # Pad the plaintext to be a multiple of 8 bytes
    padded_text = pad(plaintext.encode('utf-8'),
DES.block_size)
    # Encrypt the padded plaintext
    ciphertext = cipher.encrypt(padded_text)
    return cipher.iv, ciphertext # Return the IV and
ciphertext

def des_decrypt(key, iv, ciphertext):
    # Create a DES cipher object with the same IV
    cipher = DES.new(key.encode('utf-8'), DES.MODE_CBC,
iv)
    # Decrypt the ciphertext
    padded_plaintext = cipher.decrypt(ciphertext)
    # Unpad the plaintext
    return unpad(padded_plaintext,
DES.block_size).decode('utf-8')

# Example usage
if __name__ == "__main__":
    key = "NewKey69" # New key, must be 8 bytes long
```

```
plaintext = "This is a secret message." # New  
plaintext
```

```
# Encrypt the plaintext
```

```
iv, ciphertext = des_encrypt(key, plaintext)
```

```
print("Ciphertext (hex):", ciphertext.hex())
```

```
print("IV (hex):", iv.hex())
```

```
# Decrypt the ciphertext
```

```
decrypted_text = des_decrypt(key, iv, ciphertext)
```

```
print("Decrypted text:", decrypted_text)
```

OUTPUT:

21BA1876 - Cryptography and Network Security LAB - 5 DES Encryption and Decryption.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text

```
[2] 1 !pip install pycryptodome
```

Collecting pycryptodome
Downloading pycryptodome-3.20.0-cp35-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (3.4 kB)
Downloading pycryptodome-3.20.0-cp35-abi3-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (2.1 MB)
----- 2.1/2.1 MB 28.5 MB/s eta 0:00:00
Installing collected packages: pycryptodome
Successfully installed pycryptodome-3.20.0

```
[19] 1 from Crypto.Cipher import DES  
2 from Crypto.Util.Padding import pad, unpad
```

```
1 def des_encrypt(key, plaintext):  
2     # Create a DES cipher object  
3     cipher = DES.new(key.encode('utf-8'), DES.MODE_CBC)  
4     # Pad the plaintext to be a multiple of 8 bytes  
5     padded_text = pad(plaintext.encode('utf-8'), DES.block_size)  
6     # Encrypt the padded plaintext  
7     ciphertext = cipher.encrypt(padded_text)  
8     return cipher.iv, ciphertext # Return the IV and ciphertext  
9  
10 def des_decrypt(key, iv, ciphertext):  
11     # Create a DES cipher object with the same IV  
12     cipher = DES.new(key.encode('utf-8'), DES.MODE_CBC, iv)  
13     # Decrypt the ciphertext  
14     padded_plaintext = cipher.decrypt(ciphertext)  
15     # Unpad the plaintext  
16     return unpad(padded_plaintext, DES.block_size).decode('utf-8')  
17  
18 # Example usage  
19 if __name__ == "__main__":  
20     key = "NewKey69" # New key, must be 8 bytes long  
21     plaintext = "This is a secret message." # New plaintext  
22  
23     # Encrypt the plaintext
```

Connected to Python 3 Google Compute Engine backend

21BA1876 - Cryptography and Network Security LAB - 5 DES Encryption and Decryption.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

```
1 def des_encrypt(key, plaintext):  
2     # Create a DES cipher object  
3     cipher = DES.new(key.encode('utf-8'), DES.MODE_CBC)  
4     # Pad the plaintext to be a multiple of 8 bytes  
5     padded_text = pad(plaintext.encode('utf-8'), DES.block_size)  
6     # Encrypt the padded plaintext  
7     ciphertext = cipher.encrypt(padded_text)  
8     return cipher.iv, ciphertext # Return the IV and ciphertext  
9  
10 def des_decrypt(key, iv, ciphertext):  
11     # Create a DES cipher object with the same IV  
12     cipher = DES.new(key.encode('utf-8'), DES.MODE_CBC, iv)  
13     # Decrypt the ciphertext  
14     padded_plaintext = cipher.decrypt(ciphertext)  
15     # Unpad the plaintext  
16     return unpad(padded_plaintext, DES.block_size).decode('utf-8')  
17  
18 # Example usage  
19 if __name__ == "__main__":  
20     key = "NewKey69" # New key, must be 8 bytes long  
21     plaintext = "This is a secret message." # New plaintext  
22  
23     # Encrypt the plaintext  
24     iv, ciphertext = des_encrypt(key, plaintext)  
25     print("Ciphertext (hex):", ciphertext.hex())  
26     print("IV (hex):", iv.hex())  
27  
28     # Decrypt the ciphertext  
29     decrypted_text = des_decrypt(key, iv, ciphertext)  
30     print("Decrypted text:", decrypted_text)
```

Ciphertext (hex): ed534b40727dd8589c22e475d6f5ffba05071ff81711cfe8efcb900ae2b0cb3b
IV (hex): 62247ddd37734dc0
Decrypted text: This is a secret message.

Connected to Python 3 Google Compute Engine backend