

## STUDY OF SYMMETRIC KEY - DES &amp; AES

AIM :- To Study and Implement Symmetric Key Cryptography using DES & AES Algorithm in Python.

## ALGORITHM OF (DES) :-

1. Import DES module.
2. Create a secret key and pad the message to match the block size.
3. Encrypt the plain text.
4. Decrypt the cipher-text.
5. Display the results.

## PROGRAM :-

```
!pip install pycryptodome
from Crypto.Cipher import DES
from Crypto.Util.Padding import pad, unpad

key = b'12345678'
cipher = DES.new(key, DES.MODE_ECB)
plaintext = input("Enter message: ")
padded_text = pad(plaintext.encode(), DES.block_size)
encrypted_text = cipher.encrypt(padded_text)
print("Encrypted Text:", encrypted_text)
decrypted_text = unpad(cipher.decrypt(encrypted_text),
DES.block_size)
print("Decrypted Text:", decrypted_text.decode())
```

## Output:-

```
Enter message: hello
Encrypted Text: b'\x9a\x1f\x8b\x9e\x9e\x1f\x8b\x9e'
Decrypted Text: hello
```

ALGORITHM OF (AES) :- 1. Import AES module.  
2. Create a 16-byte secret key and pad the message.  
3. Encrypt and then decrypt using AES.  
4. Display the output.

PROGRAM:-

```
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad, unpad

key = b'thisismysecretkey'      # 16-byte key for AES-128
cipher = AES.new(key, AES.MODE_ECB)

plaintext = input("Enter message: ")

padded_text = pad(plaintext.encode(), AES.block_size)
encrypted_text = cipher.encrypt(padded_text)
print("Encrypted Text:", encrypted_text)

decrypted_text = unpad(cipher.decrypt(encrypted_text),
AES.block_size)
print("Decrypted Text:", decrypted_text.decode())
```

### **OUTPUT :-**

```
... Enter message: SECRET MESSAGE
   Encrypted Text: b'\xd6]0\xe7\xc4\xd2.$\xfc\x1b\xf5\x97\xcb\x0fb,'
   Decrypted Text: SECRET MESSAGE
```

---

CONCLUSION :- In symmetric key cryptography, the same key is used for both encryption and decryption. DES and AES are two important symmetric algorithms, with AES being more secure and faster due to its larger key size and modern design.