Assignment No 6

Cryptography and Network Security Lab (5CS453)

Name: Mrunal Anil Khade PRN: 2020BTECS00057 Class: Final Year - CSE

Title:

Advanced Encryption Standard (AES)

Aim:

To study and Implement Advanced Encryption Standard (AES)

Thory:

Advanced Encryption Standard (AES)

→

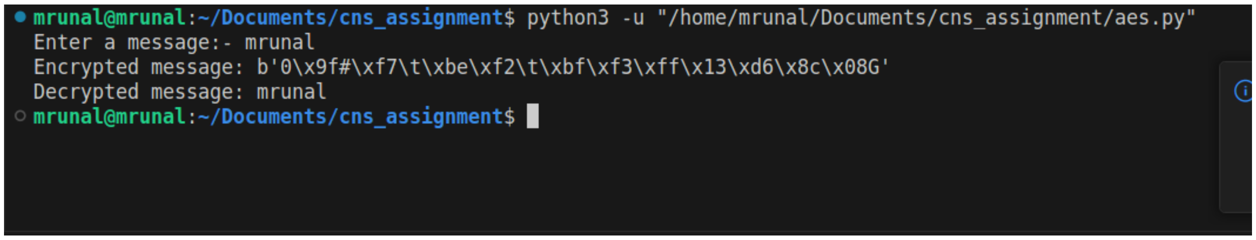
- AES, or Advanced Encryption Standard, is a widely used symmetric encryption algorithm.
- It was established by the National Institute of Standards and Technology (NIST) in 2001 as a standard for encrypting electronic data.
- Symmetric encryption means the same key is used for both encryption and decryption.
- AES operates on blocks of data and supports key sizes of 128, 192, or 256 bits.
- The algorithm consists of a series of transformations, including substitution, permutation, and mixing of the data. These operations are repeated for multiple rounds, with the number of rounds depending on the key size.
- In essence, AES provides a secure way to protect sensitive information, and its widespread adoption makes it a crucial component of modern cryptographic systems.
- It's commonly used in securing data transmitted over networks, encrypting files, and ensuring the confidentiality of sensitive information in various applications.

Code:

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

def generate_aes_key(key_size):
    if key_size not in [16, 24, 32]:
        raise ValueError("Invalid key size. Key size must be 16, 24, or 32
bytes.")
    return get_random_bytes(key_size)

def encrypt(plaintext, key):
    cipher = AES.new(key, AES.MODE_CBC)
    ciphertext = cipher.encrypt(pad(plaintext.encode('utf-8'), AES.block_size))
    return cipher.iv + ciphertext
```

```python
def decrypt(ciphertext, key):
    iv = ciphertext[:AES.block_size]
    cipher = AES.new(key, AES.MODE_CBC, iv)
    plaintext = unpad(cipher.decrypt(ciphertext[AES.block_size:]), 
AES.block_size)
    return plaintext.decode('utf-8')

key_size = int(input("Enter the key size (16, 24, or 32 bytes): "))

try:
    key = generate_aes_key(key_size)
except ValueError as e:
    print(e)
else:
    # Example usage
    text_to_encrypt = input("Enter the text to encrypt: ")

    print("Original text:", text_to_encrypt)

    encrypted_text = encrypt(text_to_encrypt, key)
    print("Encrypted text:", encrypted_text)

    decrypted_text = decrypt(encrypted_text, key)
    print("Decrypted text:", decrypted_text)
```

Output:



```
mrunal@mrunal:~/Documents/cns_assignment$ python3 -u "/home/mrunal/Documents/cns_assignment/aes.py"
Enter a message:- mrunal
Encrypted message: b'0\x9f#\xf7\t\xbe\xf2\t\xbf\xf3\xff\x13\xd6\x8c\x08G'
Decrypted message: mrunal
mrunal@mrunal:~/Documents/cns_assignment$ 
```