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 Computer Science and Engineering (Data Science)

SUB: Information Security

AY 2023-24 (Semester-V)

Experiment No: 6
60009210105
Amitesh Sawarkar
D 12

Aim: To Implement Encryption and Decryption using Product Cipher.

Theory:

1. Product Cipher.

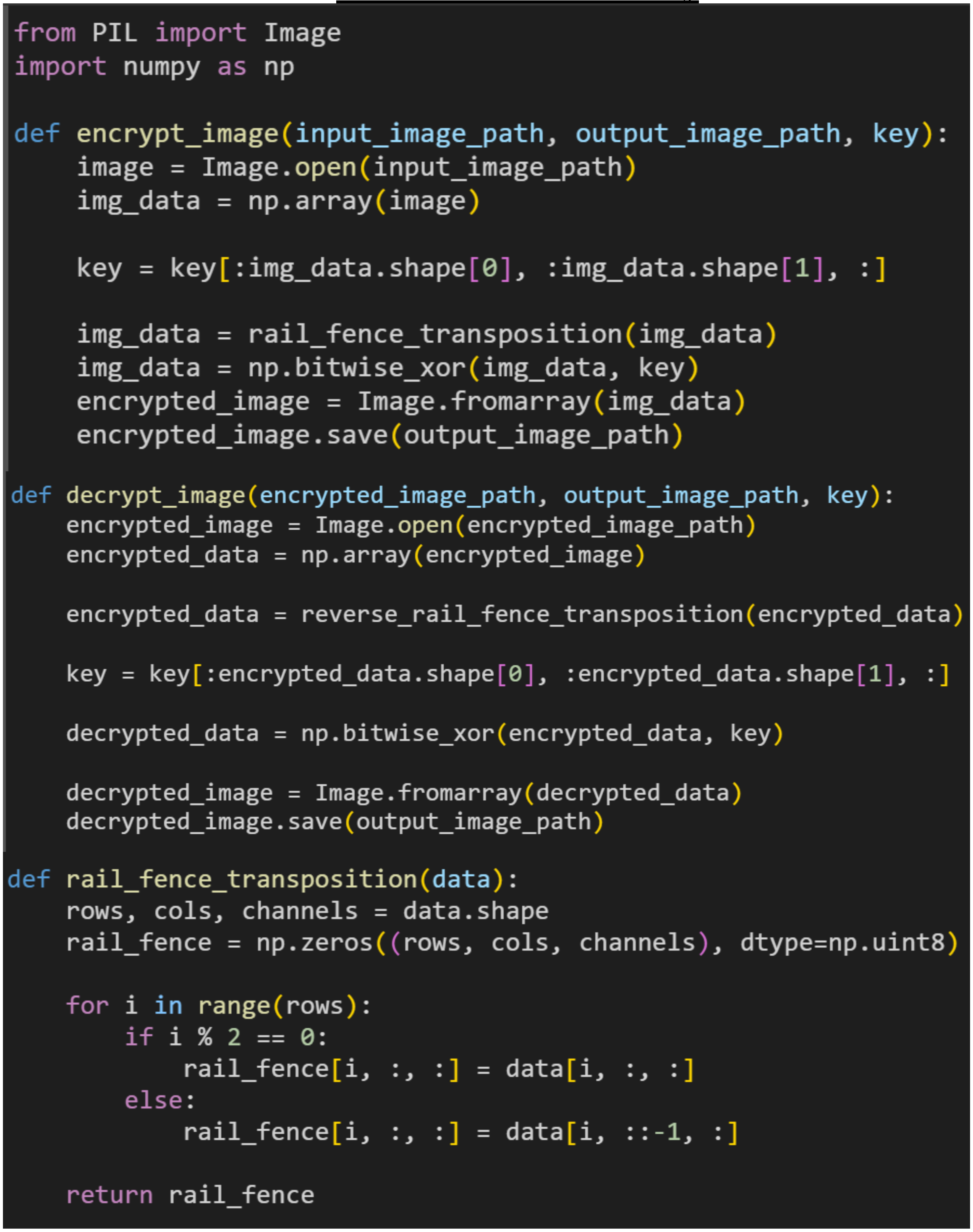
Product cipher is a combination of substitution and transposition cipher.

Product cipher, data encryption scheme in which the cipher text produced by encrypting a plaintext document is subjected to further encryption.

By combining two or more simple transposition ciphers or substitution ciphers, a more secure encryption may result.

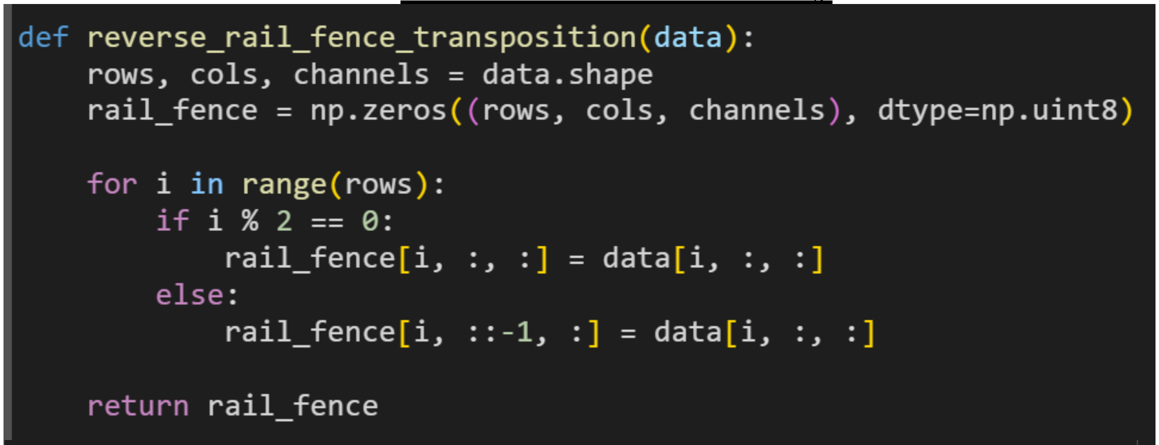
To turn data from plaintext into ciphertext, product ciphers carry out multiple rounds of substitutions and permutations, each round using a different subkey derived from the main key. It results in securely encrypted data that's very difficult to unencrypt without the proper key.

SUB: Information Security

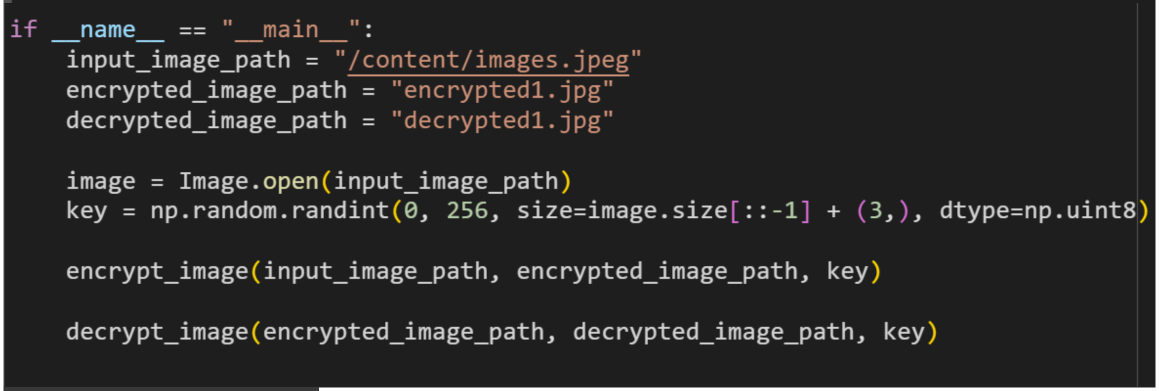


```
from PIL import Image
import numpy as np

def encrypt_image(input_image_path, output_image_path, key):
    image = Image.open(input_image_path)
    img_data = np.array(image)

    key = key[:img_data.shape[0], :img_data.shape[1], :]

    img_data = rail_fence_transposition(img_data)
    img_data = np.bitwise_xor(img_data, key)
    encrypted_image = Image.fromarray(img_data)
    encrypted_image.save(output_image_path)

def decrypt_image(encrypted_image_path, output_image_path, key):
    encrypted_image = Image.open(encrypted_image_path)
    encrypted_data = np.array(encrypted_image)

    encrypted_data = reverse_rail_fence_transposition(encrypted_data)

    key = key[:encrypted_data.shape[0], :encrypted_data.shape[1], :]

    decrypted_data = np.bitwise_xor(encrypted_data, key)

    decrypted_image = Image.fromarray(decrypted_data)
    decrypted_image.save(output_image_path)

def rail_fence_transposition(data):
    rows, cols, channels = data.shape
    rail_fence = np.zeros((rows, cols, channels), dtype=np.uint8)

    for i in range(rows):
        if i % 2 == 0:
            rail_fence[i, :, :] = data[i, :, :]
        else:
            rail_fence[i, :, :] = data[i, ::-1, :]

    return rail_fence
```

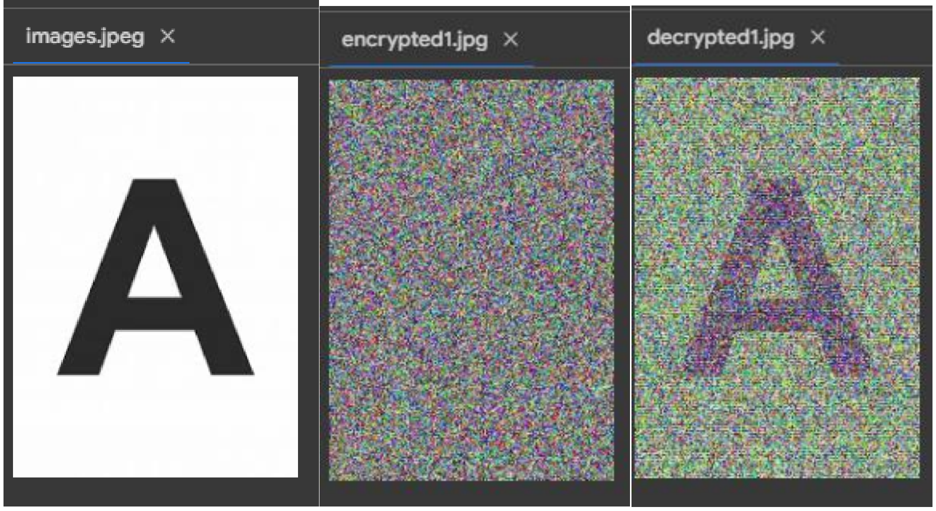
SUB: Information Security



```
def reverse_rail_fence_transposition(data):
    rows, cols, channels = data.shape
    rail_fence = np.zeros((rows, cols, channels), dtype=np.uint8)

    for i in range(rows):
        if i % 2 == 0:
            rail_fence[i, :, :] = data[i, :, :]
        else:
            rail_fence[i, ::-1, :] = data[i, :, :]

    return rail_fence
```



```
if __name__ == "__main__":
    input_image_path = "/content/images.jpeg"
    encrypted_image_path = "encrypted1.jpg"
    decrypted_image_path = "decrypted1.jpg"

    image = Image.open(input_image_path)
    key = np.random.randint(0, 256, size=image.size[::-1] + (3,), dtype=np.uint8)

    encrypt_image(input_image_path, encrypted_image_path, key)

    decrypt_image(encrypted_image_path, decrypted_image_path, key)
```



Conclusion:

In conclusion, a product cipher is a cryptographic technique that combines multiple simpler ciphers to enhance data security. It increases complexity and resistance to attacks, making

SUB: Information Security

it a valuable tool for secure data encryption. However, effective key management is essential when using a product cipher due to the involvement of multiple keys.