PHABLET

Encryption Algorithm

NIHAL AWASTHI
nihalawasthi498@gmail.com

INTRODUCTION

The phablet() encryption algorithm is an intricate and robust multi-layered encryption scheme designed to safeguard plaintext data. It encompasses a series of sequential steps, integrating various sophisticated cryptographic techniques such as substitution-permutation networks (SPNs), Feistel Networks, bitwise operations, custom block ciphers, and the RSA encryption scheme.

This algorithm follows a meticulous process divided into distinct encryption layers, each layer contributing unique operations to enhance the security of the plaintext data.

At its core, the phablet() algorithm emphasizes several fundamental aspects:

1. Multi-layered Security:
Employing a multi-layered approach, the algorithm incorporates several encryption layers, augmenting the overall security of the data.

2. Diverse Cryptographic Techniques:
Utilizes an array of cryptographic methods including substitution-permutation networks (SPNs), Feistel Networks, bitwise operations, custom block ciphers, and RSA encryption, ensuring a complex and diverse approach to encryption.

3. Key Management:
Combines fixed keys and user-provided keys for different stages of encryption, emphasizing key management and its role in securing data.

4. Complex Operations:
Implements custom block ciphers, bitwise manipulations, predefined tables for substitution and permutation, and data conversion between ASCII, hexadecimal, and binary formats, enhancing the complexity of the encryption mechanism.

5. Final Encryption Stage:
Culminates in a final encryption stage utilizing the RSA encryption scheme, leveraging modular exponentiation and prime-related functions to further fortify the security of the encrypted data.

ALGORITHM STEPS

initial Encryption (Encryption 1):
Using Feistel network encryption with keys and S-box transformations (key2 and specific constants) on the previously generated encoded text.

Secondary Encryption (Encryption 2):
Utilizing round keys and specific operations (key3 and Feistel network) to further encrypt the data.

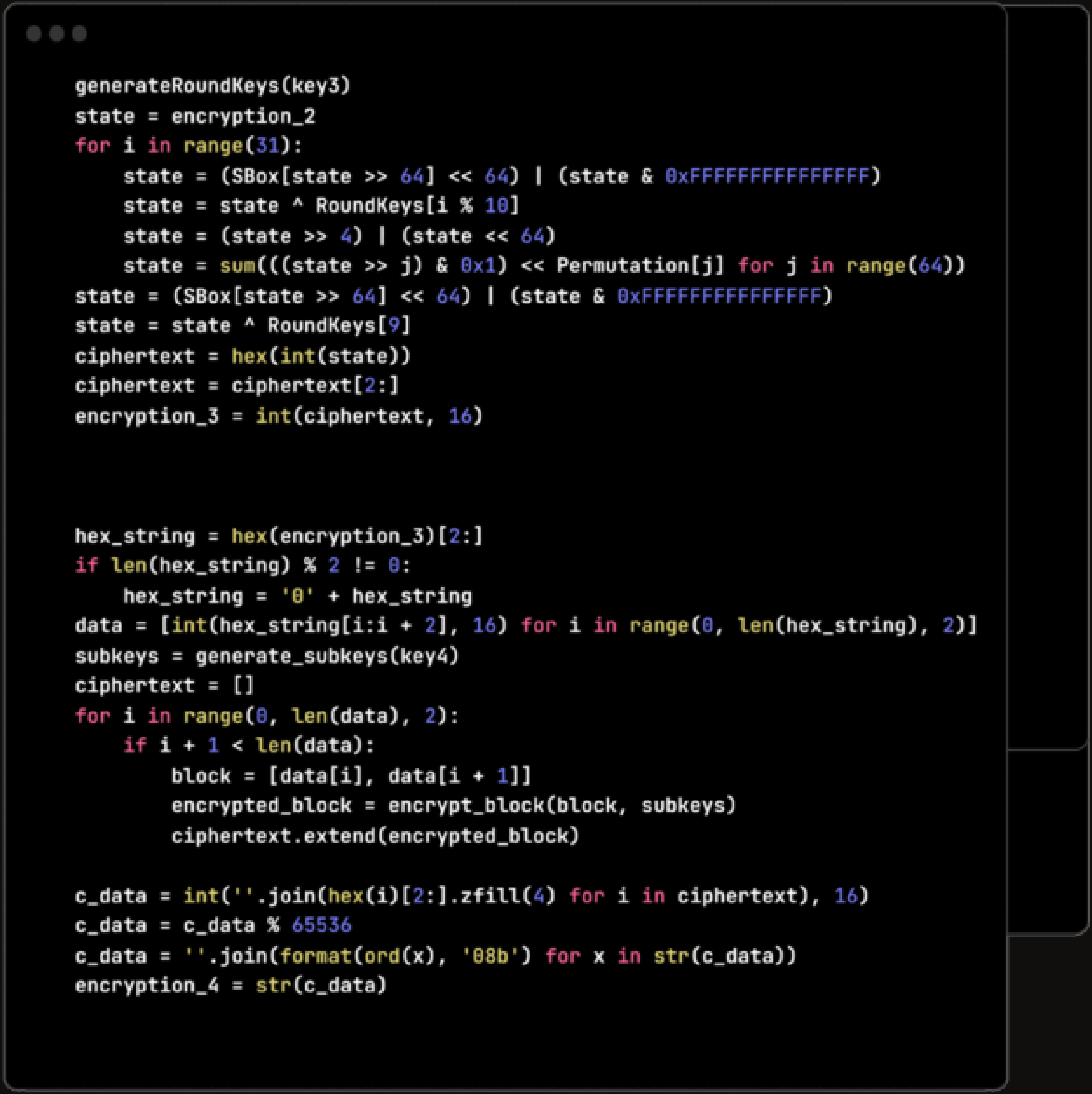
Tertiary Encryption (Encryption 3):
Employing another encryption round with additional keys and bit manipulations (key4, RoundKeys, Permutation, and SBox).

Quaternary Encryption (Encryption 4):
Applying block encryption using a key (key5), performing Feistel encryption on 64-bit blocks of data.

Quinary Encryption (Encryption 5):
Performing bitwise operations, rotation, and XOR operations on blocks of data based on specific keys (key5).

Final Encryption (RSA Encryption):
Utilizing RSA encryption (key6 and n) on the binary representation of the data, transforming each character of the encrypted text.

CODE



```
generateRoundKeys(key3)
state = encryption_2
for i in range(31):
    state = (SBox[state >> 64] << 64) | (state & 0xFFFFFFFFFFFFFFFF)
    state = state ^ RoundKeys[i % 10]
    state = (state >> 4) | (state << 64)
    state = sum(((state >> j) & 0x1) << Permutation[j] for j in range(64))
state = (SBox[state >> 64] << 64) | (state & 0xFFFFFFFFFFFFFFFF)
state = state ^ RoundKeys[9]
ciphertext = hex(int(state))
ciphertext = ciphertext[2:]
encryption_3 = int(ciphertext, 16)



hex_string = hex(encryption_3)[2:]
if len(hex_string) % 2 != 0:
    hex_string = '0' + hex_string
data = [int(hex_string[i:i + 2], 16) for i in range(0, len(hex_string), 2)]
subkeys = generate_subkeys(key4)
ciphertext = []
for i in range(0, len(data), 2):
    if i + 1 < len(data):
        block = [data[i], data[i + 1]]
        encrypted_block = encrypt_block(block, subkeys)
        ciphertext.extend(encrypted_block)

c_data = int(''.join(hex(i)[2:].zfill(4) for i in ciphertext), 16)
c_data = c_data % 65536
c_data = ''.join(format(ord(x), '08b') for x in str(c_data))
encryption_4 = str(c_data)
```

CONCLUSION

The `phablet()` encryption algorithm applies a multi-stage process involving diverse cryptographic techniques to secure plaintext data. Each step performs distinct operations contributing to a complex encryption mechanism. However, the effectiveness and security of this algorithm depend on the strength of the employed techniques and the keys used in each stage.

NIHAL AWASTHI

22MEI10055