A Hybrid Encryption Scheme using RPrime RSA Cryptosystem and VMPC Stream Cipher

Mohammad Andri Budiman[1] and Dian Rachmawati[1]
[1]*Departemen Ilmu Komputer, Fakultas Ilmu Komputer dan Teknologi Informasi, Universitas Sumatera Utara, Medan, Indonesia*

Keywords: Cryptography, Encryption, H-Rabin Algorithm, VMPC, Hybrid Cryptosystem.

Abstract: Encrypting digital messages using an asymmetric cryptography algorithm such as RPrime RSA cryptosystem may increase the size of the ciphertexts, and as a result, it increases the transmission time. To overcome this problem, in this study, RPrime RSA algorithm is combined with a symmetric cryptography algorithm, VMPC (Variably Modified Permutation Composition) stream cipher in a hybrid encryption scheme. In this scheme, a message is encrypted with VMPC, and the key of VMPC is encrypted with RPrime RSA algorithm. Since the key of VMPC is small, the size increase of the cipherkey can be omitted. As a result, the size of the ciphertext does not increase, and, therefore, this scheme does not increase the transmission time.

1 INTRODUCTION

Security issue is one of the most important aspects in the process of sending digital information. If secret information is leaked, then it may cause losses on the sides of the sender and the legitimate recipients. The solution for securing digital information is by implementing cryptography. Cryptography is a combination of art and science, especially mathematics, to maintain message security (Schneier and Diffie, 2015).

A symmetric cryptography algorithm can be called a conventional cryptography algorithm because it uses the same key for both the encryption process and the decryption process. On the other hand, an asymmetric algorithm uses two different keys for the processes of encryption and decryption (Smart, 2016). RPrime RSA is a variant of RSA algorithm (one of the first asymmetric algorith) which improves the computational cost of RSA mainly at the decryption side (Verma and Garg, 2015) (Paixao and Filho, 2003).

The security of asymmetric algorithm such as RPrime RSA usually depends on the size of the keys being used. If the sizes of the keys are long, the security is also high, but the resulting ciphertext will have a size that is many times longer than its plaintext. Therefore, the transmission cost is also high.

In this study, the RPrime RSA is put into a hybrid encryption scheme with VMPC stream cipher, a symmetric encryption algorithm, in order to avoid the size increase of the ciphertext.

2 METHODS

In this section we explain the RPrime RSA public key encryption, the VMPC stream cipher, and our hybrid encryption scheme.

2.1 RPrime RSA Public Key Encryption Algorithm

As a public key encryption algorithm, the RPrime RSA algorithm also has three phases, which are key setup (key generation), enciphering (encryption), and deciphering (decryption) (Batten, 2013).

The key setup phase is as follows (see Paixao and Filho (2003); Verma and Garg (2015)):

1. Choose an integer s <= n/k.
2. Choose k primes of n/k bits randomly, p1, p2, ..., pk, and make sure that gcd(p1 – 1) = gcd(p2 – 1) = ... = gcd(pk – 1) = 1
3. Compute N = p1 × p2 × ... × pk.

Budiman, M. and Rachmawati, D.
A Hybrid Encryption Scheme using RPrime RSA Cryptosystem and VMPC Stream Cipher.
DOI: 10.5220/0010093019191923
In *Proceedings of the International Conference of Science, Technology, Engineering, Environmental and Ramification Researches (ICOSTEERR 2018) - Research in Industry 4.0*, pages 1919-1923
ISBN: 978-989-758-449-7

4. Compute $\Phi(N) = (p1 - 1) \times (p2 - 1) \times ... \times (pk - 1)$.
5. Generate k random numbers dp1, dp2, ..., dpk of s bits and make sure that gcd(dp1, p1 – 1) = gcd(dp2, p2 – 1) = ... = gcd(dpk, pk – 1) = 1 and dp1 ≡ dp2 ≡ ... ≡ dpk ≡ 1 (mod 2).
6. With Chinese Remainder Theorem, compute d such that d ≡ dp1 (mod p1 – 1), d ≡ dp2 (mod p1=2 – 1), ..., d ≡ dpk (mod pk – 1).
7. Compute $e \equiv d^{-1} \bmod \Phi(N)$.
8. Keep the private keys (p1, p2, ..., pk, dp1, dp2, ..., dpk).
9. Publish the public keys (N, e)

The enciphering phase is as follows (see Paixao and Filho (2003); Verma and Garg (2015)):

1. Obtain the public keys (N, e)
2. Input m, the message to be enciphered.
3. Compute $c = m^e \bmod N$.
4. Send c to the recipient.

The deciphering phase is as follows (see Paixao and Filho (2003); Verma and Garg (2015)):

1. Get c from the recipient.
2. Compute the original message $m = c^d \bmod N$.

2.2 VMPC (Variably Modified Permutation Composition) Stream Cipher

VMPC stream cipher consists of two phases which are KSA (Key Scheduling Algorithm) and PRGA (Pseudo Random Number Generator). The purpose of the KSA is to randomize the S list and the purpose of the PRGA is to generate a keystream from the randomized S list. The keystream should be random and has the same length as the plaintext.

In Python language, the KSA is as follows (Zoltak , 2004).

```
SIZE = 256

S = []
for i in range(SIZE):
     S.append(i)

keylen = len(key)

j = 0

for i in range(SIZE):
     j = (j + S[i] + key[i % keylen]) %
     SIZE

     S[i], S[j] = S[j], S[i]

return S
```

In Python language, the PRGA is as follows (modified from Zoltak (2004)).

```
n = 0

count = 0

keystream = []

s = 0

while count < plaintext_length:

     s = S[(s + S[n]) % SIZE]

     keystream.append(S[ (S[ S[s] ] +
1) % SIZE ])

     S[n], S[s] = S[s], S[n]

     n = (n + 1) % SIZE

     count += 1

return keystream
```

2.3 The Hybrid Scheme

In a hybrid scheme, a message is encrypted with a symmetric encryption algorithm, while the key of the symmetric algorithm is encrypted with an asymmetric encryption algorithm (Rachmawati, Budiman, and Siburian, 2018) (Rachmawati, Sharif, Jaysilen, and Budiman, 2018).

Our proposed hybrid scheme is as follows:

A. The sender:
   1. Input m, the message to be encrypted.
   2. Choose the VMPC key (or VMPC password).
   3. To get the keystream, process the VMPC password to the phases of KSA and PRGA.
   4. Encrypt the message m by XOR-ing each bit of m with each bit of the keystream.
   5. Send the ciphertext to the recipient.
   6. Obtain the recipient's RPrime RSA public keys.

7. To get the encrypted key (or cipherkey), encrypt the VMPC password with the recipient's RPrime RSA public keys.
8. Send the cipherkey to the recipient.

B. The recipient:
1. With the recipient's private keys, decrypt the cipherkey, and get the VMPC password.
2. Decrypt the message by XOR-ing VMPC password with the ciphertext, bit by bit.

3 RESULTS AND DISCUSSIONS

Let us compare the result of the encryption. First, we encrypt a message using RPrime RSA without our scheme. Second, we encrypt that message using RPrime RSA and VMPC in our hybrid scheme. Let the message to be encrypted = "CRYPTOGRAPHY IS EASY".

3.1 Without the Hybrid Scheme

The result of the encryption using RPrime RSA without the hybrid scheme is as follows (generated using Python programming language).

```
Rprime RSA

Key Generation
k       = 3
p1      = 757
p2      = 983
p3      = 359
N       = 267143029
totient = 265776336
dp1     = 5
dp2     = 29
dp3     = 313

plaintext =  CRYPTOGRAPHY IS EASY

Encryption
'C'     67      91252973
'R'     82      8737993
'Y'     89      49056292
'P'     80      244450892
'T'     84      120789705
'O'     79      158428305
'G'     71      199302582
'R'     82      8737993
'A'     65      61508415
'P'     80      244450892
'H'     72      32487971
'Y'     89      49056292
' '     32      233053050
'I'     73      114100286
'S'     83      233248211
' '     32      233053050
'E'     69      182140055
'A'     65      61508415
'S'     83      233248211
'Y'     89      49056292

Decryption
91252973   67      'C'
8737993    82      'R'
49056292   89      'Y'
244450892  80      'P'
120789705  84      'T'
158428305  79      'O'
199302582  71      'G'
8737993    82      'R'
61508415   65      'A'
244450892  80      'P'
32487971   72      'H'
49056292   89      'Y'
233053050  32      ' '
114100286  73      'I'
233248211  83      'S'
233053050  32      ' '
182140055  69      'E'
61508415   65      'A'
233248211  83      'S'
49056292   89      'Y'
```

As can be seen from above, using 3-digit p1, p2, and p3 will create 9-digit of N. This means that each

ASCII representation of the message is encrypted to at most 9 digits of ciphertext number. As a result, the message "CRYPTOGRAPHY IS EASY" is encrypted in a series of large integers that are roughly 4 times its original size.

3.2 With the Hybrid Scheme

Let us encrypt that message again using RPrime RSA and VMPC in our hybrid scheme. We use VMPC password = "ABC". The result is as follows (also generated using Python programming language).

```
VMPC Encryption

password        = ABC
key             = [65, 66, 67]
keybits         = ['01000001',
'01000010', '01000011']

plaintext       = CRYPTOGRAPHY IS EASY

plainnum        = [67, 82, 89, 80, 84,
79, 71, 82, 65, 80, 72, 89, 32, 73, 83,
32, 69, 65, 83, 89]

plainbits       = ['01000011',
'01010010', '01011001', '01010000',
'01010100', '01001111', '01000111',
'01010010', '01000001', '01010000',
'01001000', '01011001', '00100000',
'01001001', '01010011', '00100000',
'01000101', '01000001', '01010011',
'01011001']

keystream       = [155, 18, 39, 7, 64,
208, 17, 8, 55, 34, 10, 211, 111, 140,
190, 232, 167, 61, 165, 116]

keystreambits   = ['10011011',
'00010010', '00100111', '00000111',
'01000000', '11010000', '00010001',
'00001000', '00110111', '00100010',
'00001010', '11010011', '01101111',
'10001100', '10111110', '11101000',
'10100111', '00111101', '10100101',
'01110100']

ciphernum       = [216, 64, 126, 87,
20, 159, 86, 90, 118, 114, 66, 138, 79,
197, 237, 200, 226, 124, 246, 45]

cipherbits      = ['11011000',
'01000000', '01111110', '01010111',
'00010100', '10011111', '01010110',
'01011010', '01110110', '01110010',
'01000010', '10001010', '01001111',
'11000101', '11101101', '11001000',
'11100010', '01111100', '11110110',
'00101101']
```

The ciphernum is then sent to the recipient. Please note that the cipherbits size is the same as the plainbits size. Then, we encrypt the VMPC password with the RPrime RSA as follows.

```
Rprime RSA

Key Generation
k       = 3
p1      = 757
p2      = 983
p3      = 359
N       = 267143029
totient = 265776336
dp1     = 5
dp2     = 29
dp3     = 313

plaintext =  ABC

RPrime RSA Encryption
'A'     65      61508415
'B'     66      114268051
'C'     67      91252973
```

The ciphertext numbers (61508415, 114268051, 91252973) are then sent to the recipient. Please note that the ciphertext numbers that are being sent has smaller size than the ciphertext that was generated by the encryption system that was not using hybrid scheme in section 3.1.

The recipient decrypts the ciphertext numbers using his own RPrime RSA private keys as follows.

```
RPrime RSA Decryption
61508415   65    'A'
114268051  66    'B'
91252973   67    'C'
```

Now, the recipient has known the VMPC password, which is "ABC". Then, using VMPC decryption algorithm, which is simply XOR-ing the VMPC password with the ciphertext, we get the final decryption as follows.

```
VMPC Decryption

password         = ABC
key              = [65, 66, 67]
keybits          = ['01000001',
'01000010', '01000011'

deciphernum      = [67, 82, 89, 80, 84,
79, 71, 82, 65, 80, 72, 89, 32, 73, 83,
32, 69, 65, 83, 89]

decipherbits     = ['01000011',
'01010010', '01011001', '01010000',
'01010100', '01001111', '01000111',
'01010010', '01000001', '01010000',
'01001000', '01011001', '00100000',
'01001001', '01010011', '00100000',
'01000101', '01000001', '01010011',
'01011001']

deciphertext     = CRYPTOGRAPHY IS EASY
```

4 CONCLUSIONS

In conclusion, it is clearly noted that:

1. The RPrime RSA and VMPC are working fine in the hybrid scheme, since the ciphertext can be recovered back to the plaintext.
2. The cipherbits size is the same as the plainbits size, and it means that the ciphertext size does not increase, and this is beneficial in transmitting the ciphertext.

ACKNOWLEDGEMENTS

We gratefully acknowledge that this research is funded by Lembaga Penelitian Universitas Sumatera Utara. The support is under the research grant TALENTA USU of Year 2018 Contract Number: 77/UN5.2.3.1/PPM/KP-TALENTA USU/2018.

REFERENCES

Batten, L, M., 2013. *Public key cryptography: applications and attacks* (Hoboken, NJ: John Wiley & Sons).

Paixao C, A, M., Gazzoni, Filho, D, L., 2003. An efficient variant of the RSA cryptosystem. *IACR Cryptology ePrint Archive* 159.

Rachmawati, D., Budiman, M, A., Siburian, W, S, E., 2018. Hybrid cryptosystem implementation using fast data encipherment algorithm (FEAL) and Goldwasser-Micali algorithm for file security *Journal of Physics: Conference Series* 1013 012159

Rachmawati, D., Sharif, A, Jaysilen., Budiman, M, A., 2018. Hybrid Cryptosystem Using Tiny Encryption Algorithm and LUC Algorithm *IOP Conference Series: Materials Science and Engineering* 300 012042

Schneier, B and Diffie, W., 2015. *Applied cryptography protocols, algorithms, and source code in C* (Indianapolis (Ind.): Wiley).

Smart, N, P., 2016. *Cryptography Made Simple* (Cham: Springer International Publishing).

Verma, S, and Garg, D., 2015. Improvement in rebalanced CRT RSA *International Arab Journal of Information Technology (IAJIT)* 12 6.

Zoltak, B., 2004. VMPC One-Way Function and Stream Cipher *Fast Software Encryption Lecture Notes in Computer Science* 210–25.