The Understanding of GOST Crytography Technique

Muhammad Iqbal[1], Yudi Sahputra[2], Andysah Putera Utama Siahaan[3]
Faculty of Computer Science
Universitas Pembangunan Panca Budi
Jl. Jend. Gatot Subroto Km. 4,5 Sei Sikambing, 20122, Medan, Sumatera Utara, Indonesia

Abstract - *Nowadays, it is often found a wide range of the securing information. The understanding that the author wants to design is about cryptography. Cryptography is the study of the security of data or information, commonly found in transferring data. One of them is the method of GOST, GOST is an abbreviation of "Gosudarstvennyi Standard" or "Government Standard." The algorithm is simple encryption algorithm which has some processes as many as 32 rounds and uses 64-bit block cipher with 256-bit key. GOST method also uses the S-Box 8 pieces of permanent and XOR operations and Rotate Left Shift. The author chose the GOST topic since the author wants to describe the understanding of the GOST method. The learning is presenting calculation of the encryption and decryption. The learning course is the study of cryptographic methods GOST. Learning software is also designed so that the method can be more easily understood GOST both algorithms and operations contained in this GOST method.*

Keywords – *Cryptography, GOST, Encryption, Decryption*

I. INTRODUCTION

Encryption is the way to hide information from being stolen [2][7][8]. GOST is a cryptography algorithm made in Russia. This algorithm is a rival of the DES algorithm created by the United States [1][4][6]. Structurally, this algorithm is very similar to the DES algorithm. As with algorithms DES, GOST algorithm uses the structure Feistel network encryption. This algorithm has a block size of the 64-bit message. Unlike the AES, it has a block size of the 128-bit message. Every encryption and decryption algorithms have a key [5]. This algorithm also has some rounds more than AES-256, which is 32 rounds. Each round uses eight key pieces scheduled internal use. Functions used in each round in the GOST algorithm is very simple. It just adds the subkey that in-mod with 232. Then the results are put into the S-box and rotate these results to the left as much as 11 bits. The results will be used in the next round. And so on up to 32 rounds. Key scheduling done by GOST is very simple. First, the primary key into eight pieces measuring 32-bit subkey. Each subkey is used four times in the algorithm. Twenty-four first round use keywords in the order while eight last round uses the key in the reverse order.

This understanding tries to describe the flow of the GOST algorithm. It describes the basic rule of the key generating. It also covers the round of the key. The encryption and decryption will prove the GOST method has been done.

II. THEORIES

A. Definition

GOST stands for "Gosudarstvennyi Standard" or "Government Standard." Method GOST is a block cipher algorithm developed by a national of the Soviet Union. This method was developed by the Soviet Union during the Cold War to hide data or information that is confidential at the time of the communication [3]. This algorithm is a simple encryption algorithm which has some processes as much as 32 round (round) and uses 64-bit block cipher with 256-bit key. GOST method also uses the S-Box 8 pieces different and XOR operations and Circular Shift Left. Weakness GOST known until now is because its key schedule is simply that in certain circumstances be the weak point of the method of cryptanalysis as Related-key cryptanalysis. However, this can be resolved by passing the keys to a strong hash function in cryptography such as SHA-1, and then use the results to input initialization hash key. The advantage of this method is the speed GOST pretty good, although not as fast as Blowfish, faster than IDEA.

GOST structure such as:

1. Key Store Unit (KSU) stores 256-bit string by 32-bit register (K0, K1, …, K7).
2. Twoof 32 bit register (R1, R2)
3. 32 bit adder modulo 232 (CM1)
4. Bitwise Adder XOR (CM2)
5. Substitusion block (S), an eight of 64 bit S-Box.
6. Left rotation shift register (R),11 bit.

B. Key Structure

The key structure process is the technique to compose the password to encrypt the plaintext. This process can be seen as follow.

1. Input key, 256-bit key (k1, k2, k3, k4, …, k256)
2. Generating of eight KSU

K0 = (k32, …, k1)
K1 = (k64, …, k33)
K2 = (k96, …, k65)
K3 = (k128, …, k97)
K4 = (k160, …, k129)
K5 = (k192, …, k161)
K6 = (k224, …, k193)
K7 = (k256, …, k225)

III. Result and Discussion

A. Key Generator

The process of formation of this key requires data input with the key length of 256 bits or 64 hexadecimal digits or 32 pieces of character. This process can be seen in the following example: Suppose key: “Kriptografi Metode GOST, Andysah”, then the process of formation of the key above the key is in the following table.

TABLE I
KEY GENERATOR

No.	Coordinate	X	Y
1	K	75	01001011
2	r	114	01110010
3	i	105	01101001
4	p	112	01110000
5	t	116	01110100
6	o	111	01101111
7	g	103	01100111
8	r	114	01110010
9	a	97	01100001
10	f	102	01100110
11	i	105	01101001
12		32	00100000
13	M	77	01001101
14	e	101	01100101
15	t	116	01110100
16	o	111	01101111
17	d	100	01100100
18	e	101	01100101
19		32	00100000
20	G	71	01000111
21	O	79	01001111
22	S	83	01010011
23	T	84	01010100
24	,	44	00101100
25		32	00100000
26	A	65	01000001
27	n	110	01101110
28	d	100	01100100
29	y	121	01111001
30	s	115	01110011
31	a	97	01100001
32	h	104	01101000

In Table 1, there are 32 pieces of characters which are converted to the binary digit. It is used as a key to process the encryption in GOST algorithm. The conversion to the binary key of the 256-bit is as follow.

```
0100101101110010011010010111000001110
0100011011110110011101110010011000010
1100110011010010010000000100110101100
1010111010001101111011001000110010100
1000000010001110100111101010011010100
1000001011000010000000100000101101110
1100100011110010111001101100001011010
00
```

The key will be categorized to eight parts.

```
K[0] :
      01001011011100100110100101110000
K[1] :
      01110100011011110110011101110010
K[2] :
      01100001011001100110100100100000
K[3] :
      01001101011001010111010001101111
K[4] :
      01100100011001010010000001000111
K[5] :
      01001111010100110101010000101100
K[6] :
      00100000010000010110111001100100
K[7] :
      01111001011100110110000101101000
```

B. Encryption Process

GOST encryption process of the method of processing the input plaintext data 64-bit or 16 hexadecimal digits or characters 8 through 32 stages of iterations (rounds). Suppose picking up the key on the formation and plaintext "ENKRIPSI", then the encryption process is as follows:

ENCRYPTION, ROUND 0

(1) PLAIN TEXT = ENKRIPSI
Convert to Binary=
0100010101001110010010110101001001001001010
100000101001101001001
L(0) = 10010010110010100000101010010010
R(0) = 01001010110100100111001010100010

(2) R(0) + K(0) mod 232
R(0) = 1255305890
K(0) = 244731602
-------------------------------- +
R = 1500037492 mod 232
= 1500037492
=
01011001011010001100000101110100

(3) Split to 8 part and put to SBox.
0101 = 5 = SBOX(0) = 8 = 1000
1001 = 9 = SBOX(1) = 3 = 0011
0110 = 6 = SBOX(2) = 4 = 0100
1000 = 8 = SBOX(3) = 14 = 1110
1100 = 12 = SBOX(4) = 0 = 0000
0001 = 1 = SBOX(5) = 11 = 1011
0111 = 7 = SBOX(6) = 9 = 1001
0100 = 4 = SBOX(7) = 5 = 0101

(4) Concatenate and Rotate Left Shift 11 times.
RLS(11)=0111000001011100101011000001101
0

(5) R(1) = R(0) XOR L(0)
R(0) = 01110000010111001010110000011010
L(0) = 10010010110010100000101010010010

XOR
R(1) = 11100010100101101010011010001000

(6) L(1) = R(0) before process.
L(1) = 01001010110100100111001010100010

ENCRYPTION, ROUND 1

(1) L(1) = 01001010110100100111001010100010
R(1) = 11100010100101101010011010001000

(2) R(1) + K(1) mod 232
R(1) = 3801523848
K(1) = 1323759150
-------------------------------- +
R = 5125282998 mod 232
= 830315702
= 00110001011111011001110010110110

(3) Split to 8 part and put to SBox.
0011 = 3 = SBOX(0) = 2 = 0010
0001 = 1 = SBOX(1) = 11 = 1011
0111 = 7 = SBOX(2) = 2 = 0010
1101 = 13 = SBOX(3) = 2 = 0010
1001 = 9 = SBOX(4) = 10 = 1010
1100 = 12 = SBOX(5) = 9 = 1001
1011 = 11 = SBOX(6) = 7 = 0111
0110 = 6 = SBOX(7) = 10 = 1010

(4) Concatenate and Rotate Left Shift 11 times.
RLS(11)=0001010101001011110100010101100
1

(5) R(2) = R(1) XOR L(1)
R(1) = 00010101010010111101000101011001
L(1) = 01001010110100100111001010100010
-- XOR
R(2) = 01011111100110011010001111111011

(6) L(2) = R(1) before process.
L(2) = 11100010100101101010011010001000
o
o

The process will continue until Round 31.

ENCRYPTION, ROUND 31

(1) L(31) = 10110101101111001100010101011110
R(31) = 10011000110000000111100010010101

(2) R(31) + K(0) mod 232
R(31) = 2562750613
K(0) = 244731602
-------------------------------- +
R = 2807482215 mod 232
= 2807482215
= 10100111010101101100011101100111

(3) Split to 8 part and put to SBox.
1010 = 10 = SBOX(0) = 1 = 0001
0111 = 7 = SBOX(1) = 10 = 1010
0101 = 5 = SBOX(2) = 3 = 0011
0110 = 6 = SBOX(3) = 9 = 1001
1100 = 12 = SBOX(4) = 0 = 0000
0111 = 7 = SBOX(5) = 13 = 1101
0110 = 6 = SBOX(6) = 5 = 0101
0111 = 7 = SBOX(7) = 4 = 0100

(4) Concatenate and Rotate Left Shift 11 times.
RLS(11)=1100100001101010101000001101000
1

(5) R(32) = R(31) before process.
R(32) = 10011000110000000111100010010101

(6) L(32) = R(31) XOR L(31)
R(31) = 11001000011010101010000011010001
L(31) = 10110101101111001100010101011110
--- XOR
L(32) = 01111101110101100110010110001111

(7) L(32) = b(32), b(31), ... b(1)
R(32) = a(32), a(31), ... a(1)
R = a(1), ... a(32), b(1), ... b(32)
Rin binary = 10101001000111100000
00110001100111110001101001100110101110111
1110 = convert to text

CIPHER TEXT = © **ñ¦k¾**

C. *Decryption Process*

The decryption process is the reverse of the encryption process. GOST decryption process of the method of using the same algorithm with the encryption process. Suppose picking up the key establishment and ciphertext above, the decryption process is as follows:

DECRYPTION, ROUND 0

(1) CIPHER TEXT = © ñ¦k¾
Convert to binary =

10101001000111100000000110001100111110001101
00110011010111011111110
L(0) = 01111101110101100110010110001111
R(0) = 10011000110000000111100010010101

(2) R(0) + K(0) mod 232
R(0) = 2562750613
K(0) = 244731602
-------------------------------- +
R = 2807482215 mod 232
= 2807482215
= 10100111010101101100011101100111

(3) Split to 8 part and put to SBox.
1010 = 10 = SBOX(0) = 1 = 0001
0111 = 7 = SBOX(1) = 10 = 1010
0101 = 5 = SBOX(2) = 3 = 0011
0110 = 6 = SBOX(3) = 9 = 1001
1100 = 12 = SBOX(4) = 0 = 0000
0111 = 7 = SBOX(5) = 13 = 1101
0110 = 6 = SBOX(6) = 5 = 0101
0111 = 7 = SBOX(7) = 4 = 0100

(4) Concatenate and Rotate Left Shift 11 times.
RLS(11)=11001000011010101010000011010001

(5) R(1) = R(0) XOR L(0)
R(0) = 11001000011010101010000011010001
L(0) = 01111101110101100110010110001111
-- XOR
R(1) = 10110101101111001100010101011110

(6) L(1) = R(0) before process.
L(1) = 10011000110000000111100010010101

DECRYPTION, ROUND 1

(1) L(1) = 10011000110000000111100010010101
R(1) = 10110101101111001100010101011110

(2) R(1) + K(1) mod 232
R(1) = 3049047390
K(1) = 1323759150
-------------------------------- +
R = 4372806540 mod 232
= 77839244
= 00000100101000111011101110001100

(3) Split to 8 part and put to SBox.
0000 = 0 = SBOX(0) = 4 = 0100
0100 = 4 = SBOX(1) = 6 = 0110
1010 = 10 = SBOX(2) = 12 = 1100
0011 = 3 = SBOX(3) = 1 = 0001
1011 = 11 = SBOX(4) = 14 = 1110
1011 = 11 = SBOX(5) = 5 = 0101
1000 = 8 = SBOX(6) = 0 = 0000
1100 = 12 = SBOX(7) = 6 = 0110

(4) Concatenate and Rotate Left Shift 11 times.
RLS(11)=0000111100101000000110010001101 1
0

(5) R(2) = R(1) XOR L(1)
R(1) = 00001111001010000001100100011011 0
L(1) = 10011000110000000111100010010101
-- XOR
R(2) = 10010111111010000100101010100011

(6) L(2) = R(1) before process.
L(2) = 10110101101111001100010101011110
o
o

The process continues until Round 31.

DECRYPTION, ROUND 31

(1) L(31) = 11100010100101101010011010001000
R(31) = 01001010110100100111001010100010

(2) R(31) + K(0) mod 232
R(31) = 1255305890
K(0) = 244731602
-------------------------------- +
R = 1500037492 mod 232
= 1500037492
= 01011001011010001100000101110100

(3) Split to 8 part and put to SBox.
0101 = 5 = SBOX(0) = 8 = 1000

1001 = 9 = SBOX(1) = 3 = 0011
0110 = 6 = SBOX(2) = 4 = 0100
1000 = 8 = SBOX(3) = 14 = 1110
1100 = 12 = SBOX(4) = 0 = 0000
0001 = 1 = SBOX(5) = 11 = 1011
0111 = 7 = SBOX(6) = 9 = 1001
0100 = 4 = SBOX(7) = 5 = 0101

(4) Concatenate and Rotate Left Shift 11 times.
RLS(11)=01110000010111001010110000011010

(5) R(32) = R(31) before process.
R(32) = 01001010110100100111001010100010

(6) L(32) = R(31) XOR L(31)
R(31) = 01110000010111001010110000011010
L(31) = 11100010100101101010011010001000
--- XOR
L(32) = 10010010110010100000101010010010

(7) L(32) = b(32), b(31), ... b(1)
R(32) = a(32), a(31), ... a(1)
R = a(1), ... a(32), b(1), ... b(32)
R in binary = 01000101010011100100
1011010100100100100101010000010100110100
1001 = change to Text
PLAIN TEXT = **ENKRIPSI**

IV. CONCLUSION

The downside of the GOST algorithm is a simple key schedule so that in certain circumstances be the weak point of the method of cryptanalysis as Related-key cryptanalysis. However, this can be resolved by passing the keys to a strong hash function in cryptography such as SHA-1, and then use the results to input initialization hash key. The advantage of this method is the speed GOST is pretty good.Although not as fast as Blowfish, it is faster than IDEA.

REFERENCES

[1] M. Varnamkhasti, "Overview of the Algorithms for Solving the Multidimensional Knapsack Problems," *Advanced Studies in Biology,* vol. 4, no. 1, pp. 37-47, 2012.

[2] N. Courtois dan G. V. Bard, "Algebraic Cryptanalysis of the Data Encryption Standard, In Cryptography and Coding," dalam *11-th IMA Conference*, 2007.

[3] L. Babenko dan E. Maro, "Algebraic Cryptanalysis of GOST Encryption Algorithm," *Journal of Computer and Communications,* vol. 2, no. 1, pp. 10-17, 2014.

[4] C. E. Shannon, "Communication Theory of Secrecy Systems," *Bell System Technical Journal,* vol. 28, no. 1, pp. 656-715, 1949.

[5] N. T. Courtois, "Cryptanalysis of GOST In The Multiple-Key Scenario," *Tatra Mountains Mathematical Publication,* vol. 57, no. 1, pp. 45-63, 2013.

[6] E. Biham dan A. Shamir, "Differential Cryptanalysis of DES-like Cryptosystems," *Journal of Cryptology,* vol. 4, no. 1, pp. 3-72, 1991.

[7] A. P. U. Siahaan dan R. Rahim, "Dynamic Key Matrix of Hill Cipher Using Genetic Algorithm," *International Journal of Security and Its Applications,* vol. 10, no. 8, pp. 173-180, 2016.

[8] A. P. U. Siahaan, "Factorization Hack of RSA Secret Numbers," *International Journal of Engineering Trends and Technology,* vol. 37, no. 1, pp. 15-18, 2016.