A 128-bit Block Cipher Based on Three Group Arithmetics [*]

Shenghui Su [1, 2] and Shuwang Lü [3]

[1] College of Computers, Beijing University of Technology, Beijing 100124
[2] College of Information Engineering, Yangzhou University, Yangzhou 225009
[3] Graduate School, Chinese Academy of Sciences, Beijing 100039

Abstract: Enlightened by the IDEA block cipher, the authors put forward the REESSE3+ block cipher (a symmetric key cryptosystem) based on three group arithmetics: addition modulo 2 (bit XOR), addition modulo 2 ^ 16, and multiplication modulo 2 ^ 16 + 1. Different from IDEA, REESSE3+ uses 128-bit block inputs, a 256-bit key, and a renovative round function. The authors describe the REESSE3+ cipher algorithm in the graph, and expound the encryption subkeys, encryption operation, decryption subkeys, and decryption operation. Further, demonstrate the correctness of the REESSE3+ cipher algorithm, and analyze the security of REESSE3+ from three aspects. The measures for assuring the security of REESSE3+ cover those for assuring the security of IDEA, and thus, the ability of REESSE3+ in resisting differential analysis is at least equivalent to that of IDEA.

Keywords: Block cipher algorithm; Symmetric key; Round function; Additive group; Multiplicative group

1 Introduction

In April 1997, America National Institute of Standard Technology (NIST) began to collect symmetrical key cryptosystems for the advanced encryption standard (AES) in the concerned countries. NIST required the candidates of the AES algorithm to have faster speed, security no less than the triplicate DES, blocks of 128 bits, and a key of 128, 192 or 256 bits.

Although the IDEA block cipher (a symmetric key cryptosystem) for 64-bit blocks announced in 1990 [1][2] does not satisfy the NIST′s requirements, the thought and structure of IDEA are still of enlightening effects.

The REESSE3+ block cipher (a symmetric key cryptosystem) proposed in this paper is the extension and renovation of IDEA. The block length is extended to 128 bits from old 64 bits, the key length is extended to 256 bits from old 128 bits, the round function is changed greatly, and the security inherits all the original good characteristics. Because 26 operations but not 28 operations are executed in encryption of a 128-bit plaintext block, the encryption speed of REESSE3+ is faster than IDEA.

2 Description of the REESSE3+ Cryptosystem

2.1 Cipher Algorithm

The REESSE3+ cryptosystem employs an identical algorithm called a cipher algorithm for encryption and decryption and an identical key called a session key for encryption and decryption. There are differences between the encryption subkeys and the decryption subkeys, but the decryption subkeys can be derived from the encryption subkeys. The cipher algorithm consists of 8 rounds of the iteration followed by an output transformation (see Fig. 1).

Assume that X is a 128-bit plaintext block, which is partitioned into 8 subblocks X_1, X_2, X_3, X_4, X_5, X_6, X_7, and X_8, and every X_i as an input is 16 bits long.

Assume that Y is a related 128-bit ciphertext which consists of 8 subblocks Y_1, Y_2, Y_3, Y_4,

[*] This work is supported by MOST with Project 2007CB311100 and 2009AA01Z441. Corresponding email: reesse@126.com.

Y_5, Y_6, Y_7, and Y_8, and every Y_i as an output is 16 bits long.

Assume that Z is a 256-bit session key from which the encryption subkeys and decryption subkeys are derived, and every $Z_i^{(j)}$ is a 16-bit subkey, where $1 \le i \le 8$, $1 \le j \le 9$, and j represents a round number.

In Fig. 1, the operation ($a \oplus b$) represents the bitwise XOR of two 16-bit subblocks a and b, the operation (a [+] b) represents the addition modulo 2^{16} of two 16-bit subblocks a and b, and the operation ($a \odot b$) represents the multiplication modulo (2^{16} + 1) of two 16-bit subblocks a and b with 0 corresponding to $2^{16} \in \mathbb{Z}_{2^{16}+1}$.

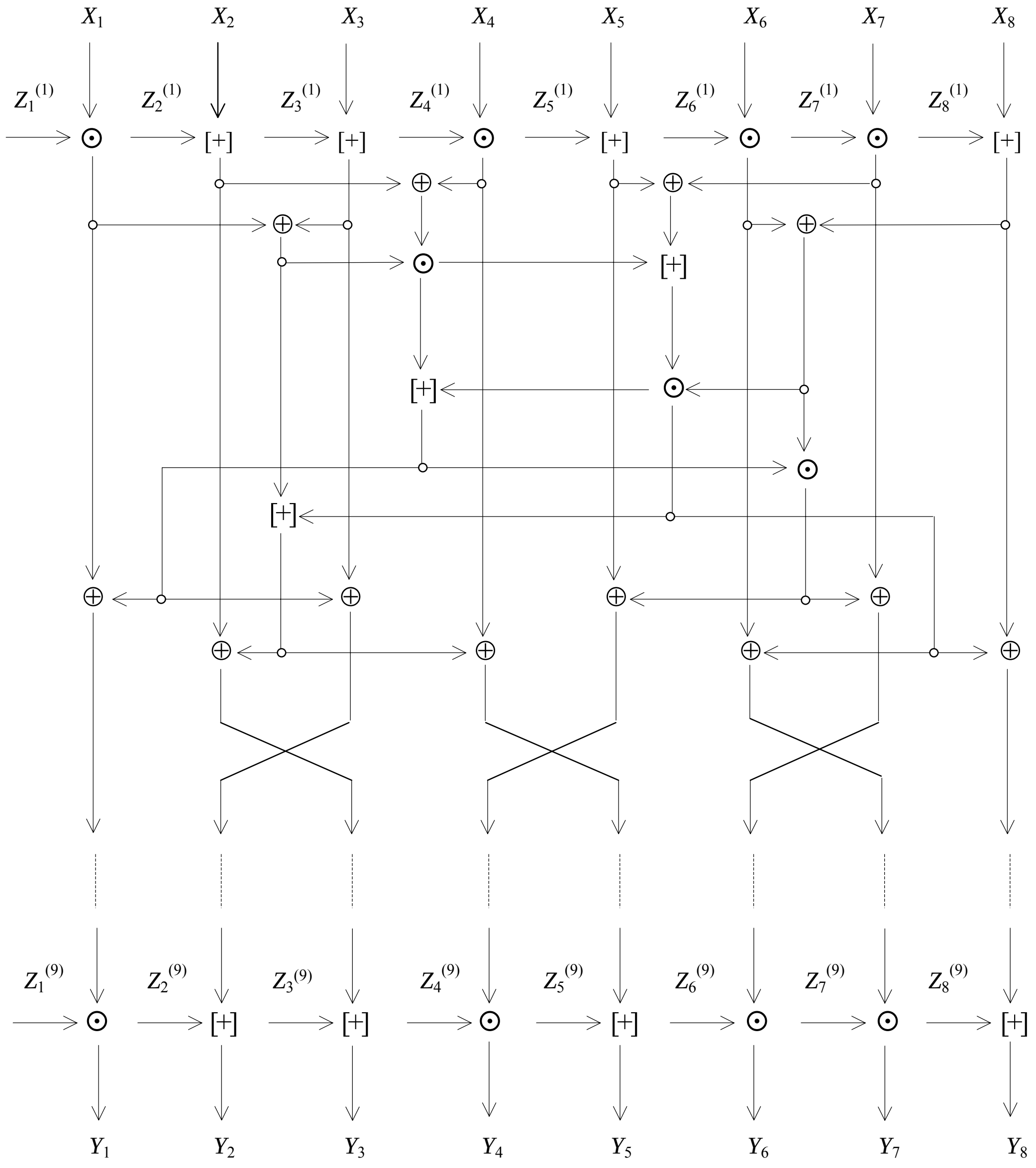

Fig.1 REESSE3+ Cipher Structure

2.2 Explanation of the REESSE3+ Cryptosystem

2.2.1 Encryption Subkeys

The leftmost 128 bits of the 256-bit key Z are divided into 8 blocks in sequence, and the 8 blocks are assigned directly to the 8 subkeys $Z_1^{(1)}$, $Z_2^{(1)}$, $Z_3^{(1)}$, $Z_4^{(1)}$, $Z_5^{(1)}$, $Z_6^{(1)}$, $Z_7^{(1)}$, and $Z_8^{(1)}$, of which each is 16 bits long, and regarded as one key input element of the first round of the iteration. When beginning the second round of the iteration, we shift cyclically Z left 25 bits (note that $25 = 16 + 9$ and $16 \times 8 = 256 - 128$), partition the leftmost 128 bits of Z into 8 blocks, and assign these blocks to the next 8 subkeys $Z_1^{(2)}$, $Z_2^{(2)}$, $Z_3^{(2)}$, $Z_4^{(2)}$, $Z_5^{(2)}$, $Z_6^{(2)}$, $Z_7^{(2)}$, and $Z_8^{(2)}$. The rest may be deduced in analogy. Thereby, we obtain the 72 subkeys for the 8 rounds of the iteration and the output transformation as follows:

Round	Encryption subkeys							
1	$Z_1^{(1)}$	$Z_2^{(1)}$	$Z_3^{(1)}$	$Z_4^{(1)}$	$Z_5^{(1)}$	$Z_6^{(1)}$	$Z_7^{(1)}$	$Z_8^{(1)}$
2	$Z_1^{(2)}$	$Z_2^{(2)}$	$Z_3^{(2)}$	$Z_4^{(2)}$	$Z_5^{(2)}$	$Z_6^{(2)}$	$Z_7^{(2)}$	$Z_8^{(2)}$
3	$Z_1^{(3)}$	$Z_2^{(3)}$	$Z_3^{(3)}$	$Z_4^{(3)}$	$Z_5^{(3)}$	$Z_6^{(3)}$	$Z_7^{(3)}$	$Z_8^{(3)}$
4	$Z_1^{(4)}$	$Z_2^{(4)}$	$Z_3^{(4)}$	$Z_4^{(4)}$	$Z_5^{(4)}$	$Z_6^{(4)}$	$Z_7^{(4)}$	$Z_8^{(4)}$
5	$Z_1^{(5)}$	$Z_2^{(5)}$	$Z_3^{(5)}$	$Z_4^{(5)}$	$Z_5^{(5)}$	$Z_6^{(5)}$	$Z_7^{(5)}$	$Z_8^{(5)}$
6	$Z_1^{(6)}$	$Z_2^{(6)}$	$Z_3^{(6)}$	$Z_4^{(6)}$	$Z_5^{(6)}$	$Z_6^{(6)}$	$Z_7^{(6)}$	$Z_8^{(6)}$
7	$Z_1^{(7)}$	$Z_2^{(7)}$	$Z_3^{(7)}$	$Z_4^{(7)}$	$Z_5^{(7)}$	$Z_6^{(7)}$	$Z_7^{(7)}$	$Z_8^{(7)}$
8	$Z_1^{(8)}$	$Z_2^{(8)}$	$Z_3^{(8)}$	$Z_4^{(8)}$	$Z_5^{(8)}$	$Z_6^{(8)}$	$Z_7^{(8)}$	$Z_8^{(8)}$
Output Tra	$Z_1^{(9)}$	$Z_2^{(9)}$	$Z_3^{(9)}$	$Z_4^{(9)}$	$Z_5^{(9)}$	$Z_6^{(9)}$	$Z_7^{(9)}$	$Z_8^{(9)}$.

2.2.2 Encryption Operation

At the beginning, the 128-bit plaintext block X is partitioned into 8 16-bit subblocks X_1, X_2, X_3, X_4, X_5, X_6, X_7, and X_8 which are regarded as the input elements of the first round of the iteration.

Every round of the iterations performs three types of operations between the (interim) plaintext subblocks and the encryption subkeys: addition mod 2^{16}, multiplication mod $(2^{16} + 1)$, and addition mod 2, namely bitwise XOR. The operation order is described in the next section.

The output elements of each of the previous 7 rounds, where the 2nd subblock and 3rd sub-block are exchanged for each other, the 4th subblock and 5th subblock are exchanged for each other, and the 6th subblock and 7th subblock are exchanged for each other, are regarded as the input elements of the next round. The exchange between some two outputs of Round 8 is not needed (See Fig.1).

After finishing the 8 rounds of the iteration, we do an extra output transformation, and then acquire 8 output subblocks Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7, and Y_8, which are combined into a 128-bit ciphertext block Y.

2.2.3 Decryption Subkeys

The decryption subkeys are deduced from the encryption subkeys, while the encryption subkeys are derived directly from the 256-bit session key Z. In terms of the inverse operation rules, the decryption subkeys corresponding to the 8 rounds of the iteration and the output transformation are as follows:

Round	Decryption subkeys							
1	$Z_1^{(9)\,-1}$	$-Z_2^{(9)}$	$-Z_3^{(9)}$	$Z_4^{(9)\,-1}$	$-Z_5^{(9)}$	$Z_6^{(9)\,-1}$	$Z_7^{(9)\,-1}$	$-Z_8^{(9)}$

2	$Z_1^{(8)\,-1}$	$-Z_3^{(8)}$	$-Z_2^{(8)}$	$Z_5^{(8)\,-1}$	$-Z_4^{(8)}$	$Z_7^{(8)\,-1}$	$Z_6^{(8)\,-1}$	$-Z_8^{(8)}$
3	$Z_1^{(7)\,-1}$	$-Z_3^{(7)}$	$-Z_2^{(7)}$	$Z_5^{(7)\,-1}$	$-Z_4^{(7)}$	$Z_7^{(7)\,-1}$	$Z_6^{(7)\,-1}$	$-Z_8^{(7)}$
4	$Z_1^{(6)\,-1}$	$-Z_3^{(6)}$	$-Z_2^{(6)}$	$Z_5^{(6)\,-1}$	$-Z_4^{(6)}$	$Z_7^{(6)\,-1}$	$Z_6^{(6)\,-1}$	$-Z_8^{(6)}$
5	$Z_1^{(5)\,-1}$	$-Z_3^{(5)}$	$-Z_2^{(5)}$	$Z_5^{(5)\,-1}$	$-Z_4^{(5)}$	$Z_7^{(5)\,-1}$	$Z_6^{(5)\,-1}$	$-Z_8^{(5)}$
6	$Z_1^{(4)\,-1}$	$-Z_3^{(4)}$	$-Z_2^{(4)}$	$Z_5^{(4)\,-1}$	$-Z_4^{(4)}$	$Z_7^{(4)\,-1}$	$Z_6^{(4)\,-1}$	$-Z_8^{(4)}$
7	$Z_1^{(3)\,-1}$	$-Z_3^{(3)}$	$-Z_2^{(3)}$	$Z_5^{(3)\,-1}$	$-Z_4^{(3)}$	$Z_7^{(3)\,-1}$	$Z_6^{(3)\,-1}$	$-Z_8^{(3)}$
8	$Z_1^{(2)\,-1}$	$-Z_3^{(2)}$	$-Z_2^{(2)}$	$Z_5^{(2)\,-1}$	$-Z_4^{(2)}$	$Z_7^{(2)\,-1}$	$Z_6^{(2)\,-1}$	$-Z_8^{(2)}$
Output Tra	$Z_1^{(1)\,-1}$	$-Z_2^{(1)}$	$-Z_3^{(1)}$	$Z_4^{(1)\,-1}$	$-Z_5^{(1)}$	$Z_6^{(1)\,-1}$	$Z_7^{(1)\,-1}$	$-Z_8^{(1)}$,

where $Z_i^{(j)\,-1}$ denotes the multiplication inverse of $Z_i^{(j)}$ mod ($2^{16} + 1$), namely $Z_i^{(j)} \odot Z_i^{(j)\,-1} \equiv 1$ (mod $2^{16} + 1$), and $-Z_i^{(j)}$ denotes the addition inverse of $Z_i^{(j)}$ mod 2^{16}, namely $Z_i^{(j)}$ [+] ($-Z_i^{(j)}$) $\equiv 0$ (mod 2^{16}).

In particular, for multiplication mod ($2^{16} + 1$), the inverse of 2^{16} is still 2^{16}, and because the lower 16 bits of 2^{16} are all zero, we use the 16-bit zero subblock to represent 2^{16} or its inverse, that is, the multiplication inverse of 0 is still 0.

2.2.4 Decryption Operation

Decryption is the reverse operation of encryption.

The decryption employs the same structure: 8 rounds of the iteration and the output transformation as what is illustrated by Fig.1.

The 128-bit ciphertext block *Y* and the decryption subkeys are the inputs of the cipher algorithm. Refer to the next section for the operation order.

3 Proof of Correctness of the Cipher Algorithm

Assume that there is only one round of the iteration before the output transformation. This does not influence the correctness of the algorithm because we only need to show that the round function is reversible.

The plaintext subblocks X_i and the subkeys $Z_i^{(1)}$ ($i = 1, \ldots, 8$) are regarded as inputs, and according to Fig. 1, the encryption operation is as follows:

(01) $A = X_1 \odot Z_1^{(1)}$ (02) $B = X_2\ [+]\ Z_2^{(1)}$
(03) $C = X_3\ [+]\ Z_3^{(1)}$ (04) $D = X_4 \odot Z_4^{(1)}$
(05) $E = X_5\ [+]\ Z_5^{(1)}$ (06) $F = X_6 \odot Z_6^{(1)}$
(07) $G = X_7 \odot Z_7^{(1)}$ (08) $H = X_8\ [+]\ Z_8^{(1)}$
(09) $I = A \oplus C$ (10) $J = B \oplus D$
(11) $K = E \oplus G$ (12) $L = F \oplus H$
(13) $M = I \odot J$ (14) $N = K\ [+]\ M$
(15) $\Gamma = L \odot N$ (16) $P = M\ [+]\ \Gamma$
(17) $\Phi = I\ [+]\ \Gamma$ (18) $\Omega = L \odot P$
(19) $Q = A \oplus P$ (20) $R = C \oplus P$
(21) $S = E \oplus \Omega$ (22) $T = G \oplus \Omega$
(23) $U = B \oplus \Phi$ (24) $V = D \oplus \Phi$
(25) $W = F \oplus \Gamma$ (26) $\Lambda = H \oplus \Gamma$

Notice that exchange is not done between some two subblocks of the iteration output.

The output transformation is as follows:

(01) $Y_1 = Q \odot Z_1^{(2)}$

(02) $Y_2 = U \boxplus Z_2^{(2)}$

(03) $Y_3 = R \boxplus Z_3^{(2)}$

(04) $Y_4 = V \odot Z_4^{(2)}$

(05) $Y_5 = S \boxplus Z_5^{(2)}$

(06) $Y_6 = W \odot Z_6^{(2)}$

(07) $Y_7 = T \odot Z_7^{(2)}$

(08) $Y_8 = \Lambda \boxplus Z_8^{(2)}$

The decryption operation employs the same algorithm as what is illustrated by Fig 1. The ciphertext sub-blocks Y_i and the inverses of the subkeys $Z_i^{(2)}$ (i = 1, …, 8) are regarded as inputs. According to Fig 1, the decryption operation is as follows:

(01) $A' = Y_1 \odot (Z_1^{(2)})^{-1} = Q$

(02) $B' = Y_2 \boxplus (-Z_2^{(2)}) = U$

(03) $C' = Y_3 \boxplus (-Z_3^{(2)}) = R$

(04) $D' = Y_4 \odot (Z_4^{(2)})^{-1} = V$

(05) $E' = Y_5 \boxplus (-Z_5^{(2)}) = S$

(06) $F' = Y_6 \odot (Z_6^{(2)})^{-1} = W$

(07) $G' = Y_7 \odot (Z_7^{(2)})^{-1} = T$

(08) $H' = Y_8 \boxplus (-Z_8^{(2)}) = \Lambda$

(09) $I' = A' \oplus C' = Q \oplus R = A \oplus C = I$

(10) $J' = B' \oplus D' = U \oplus V = B \oplus D = J$

(11) $K' = E' \oplus G' = S \oplus T = E \oplus G = K$

(12) $L' = F' \oplus H' = W \oplus \Lambda = F \oplus H = L$

(13) $M' = I' \odot J' = I \odot J = M$

(14) $N' = K' \boxplus M' = K \boxplus M = N$

(15) $\Gamma' = L' \odot N' = L \odot N = \Gamma$

(16) $P' = M' \boxplus \Gamma' = M \boxplus \Gamma = P$

(17) $\Phi' = I' \boxplus \Gamma' = I \boxplus \Gamma = \Phi$

(18) $\Omega' = L' \odot P' = L \odot P = \Omega$

(19) $Q' = A' \oplus P' = Q \oplus P = A \oplus P \oplus P = A$

(20) $R' = C' \oplus P' = R \oplus P = C \oplus P \oplus P = C$

(21) $S' = E' \oplus \Omega' = S \oplus \Omega = E \oplus \Omega \oplus \Omega = E$

(22) $T' = G' \oplus \Omega' = T \oplus \Omega = G \oplus \Omega \oplus \Omega = G$

(23) $U' = B' \oplus \Phi' = U \oplus \Phi = B \oplus \Phi \oplus \Phi = B$

(24) $V' = D' \oplus \Phi' = V \oplus \Phi = D \oplus \Phi \oplus \Phi = D$

(25) $W' = F' \oplus \Gamma' = W \oplus \Gamma = F \oplus \Gamma \oplus \Gamma = F$

(26) $\Lambda' = H' \oplus \Gamma' = \Lambda \oplus \Gamma = H \oplus \Gamma \oplus \Gamma = H$

Notice that exchange is not done between some two subblocks of the iteration output.

The output transformation of the decryption operation is as follows:

(01) $Y_1' = Q' \odot (Z_1^{(1)})^{-1} = A \odot (Z_1^{(1)})^{-1} = X_1$

(02) $Y_2' = U' \boxplus (-Z_2^{(1)}) = B \boxplus (-Z_2^{(1)}) = X_2$

(03) $Y_3' = R' \boxplus (-Z_3^{(1)}) = C \boxplus (-Z_3^{(1)}) = X_3$

(04) $Y_4' = V' \odot (Z_4^{(1)})^{-1} = D \odot (Z_4^{(1)})^{-1} = X_4$

(05) $Y_5' = S' \boxplus (-Z_5^{(1)}) = E \boxplus (-Z_5^{(1)}) = X_5$

(06) $Y_6' = W' \odot (Z_6^{(1)})^{-1} = F \odot (Z_6^{(1)})^{-1} = X_6$

(07) $Y_7' = T' \odot (Z_7^{(1)})^{-1} = G \odot (Z_7^{(1)})^{-1} = X_7$

(08) $Y_8' = \Lambda' \boxplus (-Z_8^{(1)}) = H \boxplus (-Z_8^{(1)}) = X_8$

Through the decryption operation, the original plaintext block X is obtained. Therefore the cipher algorithm illustrated by Fig.1 can decrypt a ciphertext correctly.

4 Analysis of Security of REESSE3+

4.1 Inheriting the Some Characteristic of IDEA in Security

The design of the REESSE3+ complies with the principle of "confusion" and "diffusion", and inherits the some characteristic of IDEA in security [3].

The confusion is implemented through the mixing operations over the three different groups of which any two cannot construct a field due to the inconsistency of the sets or the dissatisfaction of the distribution law. The diffusion is implemented through the extended Multiplication-Addition structure which makes the output of every operator be regarded as the input of another distinct operator, and each of output subblocks be involved in every input subblock and every subkey [2]. Besides, the exchange of output subblock 4 and 5 makes the mutual infiltration of the leftmost 64 bits and the rightmost 64 bits faster.

Hence, the ability of REESSE3+ in resisting differential analysis [4] is at least equivalent to that of IDEA.

4.2 More Complex Round Function

The round function of REESSE3+ contains seven modular multiplications and seven modular additions, and it is more complex than that of IDEA.

The more complex the round function is, the more expeditious diffusion of the bits in a block is, the more irregular confusion of the bits is, and the more ineffective the differential analysis method is [4][5].

4.3 More Extensive Variation Range

The confusion and diffusion in REESSE3+ are made within the range of 128 bits instead of the old range of 64 bits.

As a result of extending the range, the round trail of the iteration will be diluted. Therefore, differential analysis approach will become more ineffective [6].

5 Conclusions

The block length of 128 bits will be securer and fitter for the future trends and the interfaces of the most application systems.

For encrypting 128-bit plaintext block, the IDEA algorithm needs to be called two times, and contains 28 operations in all while the REESSE3+ algorithm needs to be called only one time, and contains 26 operations. Thus, the latter saves one multiplication and one addition, and is faster than the former.

The round function of REESSE3+ is some more complex, but realizing the round function with hardware or software is still convenient.

Obviously, if we change the arrangement sequence of the operators “⊙” and “[+]” at the two terminals of Fig.1, then the correctness of the algorithm will be not influenced, but the security of the algorithm will probably be influenced, increased or decreased.

Interested readers may make a deep comparison analysis of the security of the REESSE3+ cryptosystem.

Acknowledgment

The authors would like to thank the Academicians Jiren Cai, Zhongyi Zhou, Jianhua Zheng, Changxiang Shen, Zhengyao Wei, Binxing Fang, Guangnan Ni, Andrew C. Yao, and Xicheng Lu for their important guidance, suggestions, and helps.

The authors also would like to thank the Professors Dingyi Pei, Dengguo Feng, Jie Wang, Ronald L. Rivest, Moti Yung, Adi Shamir, Dingzhu Du, Mulan Liu, Huanguo Zhang, Yixian Yang, Xuejia Lai, Yongfei Han, Yupu Hu, Dongdai Lin, Chuankun Wu, Rongquan Feng, Ping Luo, Jianfeng Ma, Zhenfu Cao, Lusheng Chen, Chao Li, Wenbao Han, Bogang Lin, Qibin Zhai, Hong Zhu, Renji Tao, Bingru Yang, Zhiying Wang, Quanyuan Wu, and Zhichang Qi for their important advice,

suggestions, and corrections.

References

[1] X. Lai and J. Massey, “A Proposal for a New Block Encryption Standard”, *Proc. of Advances in Cryptology-EUROCRYPT ’90*, Berlin: Springer-Verlag, 1991, pp. 389-404.

[2] B. Schneier, *Applied Cryptography* (2nd Ed.), New York: John Wiley & Sons, Inc., 1996.

[3] A. Vysala, “The Strong Block Cipher: IDEA”, *CSI Communications*, May 2013, pp. 24-25.

[4] E. Biham and A. Shamir, “Differential Cryptanalysis of DES-like Cryptosystems”, *Proc. of Advances in Cryptology — CRYPTO '90*, Berlin: Springer-Verlag, 1990, pp. 2-21.

[5] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*, London: CRC Press, 1997.

[6] E. Biham and A. Shamir, “Differential Cryptanalysis of the Full 16-Round DES”, *Proc. of Advances in Cryptology — CRYPTO '92*, Berlin: Springer-Verlag, 1993, pp. 487-496.