Novel Data Encryption Algorithm

Rajat Goel[1], Ripu R Sinha[2], O.P. Rishi[3]

[1]Central University of Rajasthan, Kishangarh, Rajasthan, India

[2]School of Computer and System Sciences, Jaipur national University

Jaipur, Rajasthan, India-302025

[3]Central University of Rajasthan, Kishangarh, Rajasthan, India

Abstract

We always strive to get better algorithms for securing data. A variety of such algorithms are being used in cryptography. Manly block and stream ciphers are available and one of them is International Data Encryption Algorithm (IDEA), which was regarded arguably as one of the best for encryption purposes. A considerable time has elapsed since its advent and this period has witnessed a wide development in process approaches and applications. The number of transactions and exchanges of data has increased exponentially. Consequently, better and novel attacks on data evolved. Researchers believe that the security of the algorithm needs to be improved keeping a check on the time and space complexity. Within this research work we are looking for a robust algorithm known as NDEA which can be applied for securing modern environment applications.

***Keywords*:** *Novel Data Encryption Algorithm (NDEA), Data Encryption Standard (DES), International Data Encryption Algorithm (IDEA), Fiestel Structure*

1. Introduction

IDEA is an iterated block cipher algorithm based on the Fiestel network. It was designed by Xuejia Lai and James Massey in 1991. Feistel network or Fiestel cipher is a symmetric structure used in the construction of block ciphers. It was named after the German cryptographer Horst Feistel. A Feistel network is an iterated cipher with an internal function called a r ound function. Iterated block ciphers are constructed by repeatedly applying the round function. The number of rounds varies from algorithm to algorithm. The general setup of each round is almost the same. A key schedule is an algorithm that, given the key, calculates the subkeys for these rounds. A large number of block ciphers use the scheme, including the Data Encryption Standard (DES), IDEA etc. The advantage of Fiestel Cipher is that the operations of encryption and decryption are very similar or even identical. This reduces the size of the code almost by half. The only change required is a reversal of the key schedule and inversion of their values. Hence, the Feistel network model scores over substitution and transposition models as the round function need not be invertible. A block cipher encryption algorithm (E) takes plaintext M of a particular length and key K as an input, and outputs a corresponding ciphertext of the same length. The decryption algorithm (E^{-1}) takes the cipher text as an input together with the key, and yields the original block of plaintext of the same length such that:

$$E_K(M) = C\,; \quad E_K^{-1}(C) = M$$

Lucifer was the first block cipher developed at IBM in the 1970s. The Data Encryption Standard (DES) appeared in 1976.

1.1 IDEA

- Key size : **128** bits
- Plaintext Block size : **64** bits
- Rounds: 8.5

It provides high level security not based on keeping the algorithm a secret, but by keeping the key secret which makes it suitable for use in a wide range of applications worldwide. It can be economically implemented in electronic components (VLSI Chip).

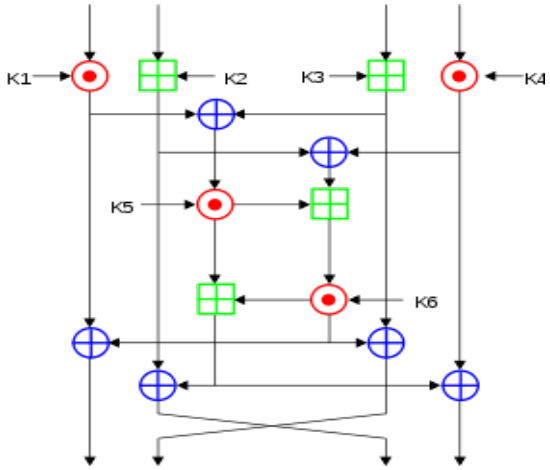

Fig. 1 First 8 rounds of IDEA

Plaintext is divided into four 16-bit sub-blocks: X1, X2, X3 and X4 (see fig. 1) and identical operations are performed on the four parts in 8 rounds. The 128-bit key is split into eight 16-bit blocks, which become eight subkeys. The first six subkeys are used in round one, and the remaining two subkeys are used in round two, similarly each round uses six 16-bit sub-keys for 8 rounds while the last half-round uses four, i.e. a total of 52 keys. First 6 keys are extracted directly from the main key. Further groups of keys are created by rotating the main key left by 25 bits.

The mathematical operations involved in each of the rounds are:

- Bitwise Exclusive OR (denoted by $\oplus$).
- Addition modulo 2^{16} (denoted by $\boxplus$).
- Multiplication modulo $2^{16}+1$ (denoted by $\odot$).

After the eight rounds there is a half round (as illustrated in fig. 2):

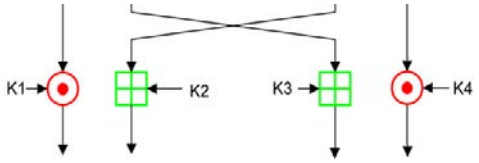

Fig. 2 The last half round of IDEA

1.1.1 Algorithm

Bruce Schneier [4] provided a f ourteen-step algorithm of IDEA. Here are the fourteen steps of a complete round (multiply means multiplication modulo $2^{16} + 1$, and add means addition modulo 2^{16}):

1. Multiply X1 and the first subkey K1.
2. Add X2 and the second subkey K2.
3. Add X3 and the third subkey K3.
4. Multiply X4 and the fourth subkey K4.
5. Bitwise XOR the results of steps 1 and 3.
6. Bitwise XOR the results of steps 2 and 4.
7. Multiply the result of step 5 and the fifth subkey K5.
8. Add the results of steps 6 and 7.
9. Multiply the result of step 8 and the sixth subkey K6.
10. Add the results of steps 7 and 9.
11. Bitwise XOR the results of steps 1 and 9.
12. Bitwise XOR the results of steps 3 and 9.
13. Bitwise XOR the results of steps 2 and 10.
14. Bitwise XOR the results of steps 4 and 10.

For every round except the final transformation, a swap occurs, and the input to the next round is: result of step 11 k result of step 13 k result of step 12 k result of step 14, which becomes X1 k X2 k X3 k X4, the input for the next round. After round 8, a ninth "half round" final transformation occurs:

After the eighth round, there is a final output transformation:

(1) Multiply X_1 and the first subkey.
(2) Add X_2 and the second subkey.
(3) Add X_3 and the third subkey.
(4) Multiply X_4 and the fourth subkey.

Finally, the four sub-blocks are reattached to produce the ciphertext.

1.1.2 Decryption

Decryption algorithm is exactly the same. The subkeys are reversed and slightly different. The decryption subkeys are either the additive or multiplicative inverses of the encryption subkeys. Table 1 shows the encryption subkeys and the corresponding decryption subkeys.

Table 1: IDEA Encryption and Decryption Subkeys

Round	Encryption Subkeys	Decryption Subkeys
1st	$K_1^{(1)}\ K_2^{(1)}\ K_3^{(1)}$ $K_4^{(1)}\ K_5^{(1)}\ K_6^{(1)}$	$K_1^{(9)-1}\ -K_2^{(9)}\ -K_3^{(9)}$ $K_4^{(9)-1}\ K_5^{(8)}\ K_6^{(8)}$
2nd	$K_1^{(2)}\ K_2^{(2)}\ K_3^{(2)}$ $K_4^{(2)}\ K_5^{(2)}\ K_6^{(2)}$	$K_1^{(8)-1}\ -K_3^{(8)}\ -K_2^{(8)}$ $K_4^{(8)-1}\ K_5^{(7)}\ K_6^{(7)}$
3rd	$K_1^{(3)}\ K_2^{(3)}\ K_3^{(3)}$ $K_4^{(3)}\ K_5^{(3)}\ K_6^{(3)}$	$K_1^{(7)-1}\ -K_3^{(7)}\ -K_2^{(7)}$ $K_4^{(7)-1}\ K_5^{(6)}\ K_6^{(6)}$
4th	$K_1^{(4)}\ K_2^{(4)}\ K_3^{(4)}$ $K_4^{(4)}\ K_5^{(4)}\ K_6^{(4)}$	$K_1^{(6)-1}\ -K_3^{(6)}\ -K_2^{(6)}$ $K_4^{(6)-1}\ K_5^{(5)}\ K_6^{(5)}$
5th	$K_1^{(5)}\ K_2^{(5)}\ K_3^{(5)}$ $K_4^{(5)}\ K_5^{(5)}\ K_6^{(5)}$	$K_1^{(5)-1}\ -K_3^{(5)}\ -K_2^{(5)}$ $K_4^{(5)-1}\ K_5^{5(4)}\ K_6^{(4)}$
6th	$K_1^{(6)}\ K_2^{(6)}\ K_3^{(6)}$ $K_4^{(6)}\ K_5^{(6)}\ K_6^{(6)}$	$K_1^{(4)-1}\ -K_3^{(4)}\ -K_2^{(4)}$ $K_4^{(4)-1}\ K_5^{(3)}\ K_6^{(3)}$
7th	$K_1^{(7)}\ K_2^{(7)}\ K_3^{(7)}$ $K_4^{(7)}\ K_5^{(7)}\ K_6^{(7)}$	$K_1^{(3)-1}\ -K_3^{(3)}\ -K_2^{(3)}$ $K_4^{(3)-1}\ K_5^{(2)}\ K_6^{(2)}$
8th	$K_1^{(8)}\ K_2^{(8)}\ K_3^{(8)}$ $K_4^{(8)}\ K_5^{(8)}\ K_6^{(8)}$	$K_1^{(2)-1}\ -K_3^{(2)}\ -K_2^{(2)}$ $K_4^{(2)-1}\ K_5^{(1)}\ K_6^{(1)}$
8.5^{th}	$K_1^{(9)}\ K_2^{(9)}$ $K_3^{(9)}\ K_4^{(9)}$	$K_1^{(1)-1}\ -K_2^{(1)}$ $-K_3^{(1)}\ K_4^{(1)-1}$

Here, $K_4^{(1)}$ denotes the 4^{th} key of the 1^{st} round
$K_4^{(1)-1}$ denotes the multiplicative inverse of $K_4^{(1)}$
$-K_2^{(1)}$ denotes the additive inverse of $K_2^{(1)}$

1.2 Related Work

IDEA is one of the world's most secure cryptographic algorithms but many researchers now consider it obsolete and feel a need to modify it. It has been emphasized that the modifications should be such that the algorithm remains efficient i.e. the time and space complexity should not increase too much, so increasing the rounds is not an intelligent approach as per Nick Hoffman [8]. Increasing the plaintext block size is also not feasible as per [9] because unlike 65,537 i.e. 2^16+1, 2^32+1 is not prime, so IDEA cannot be scaled up to a 128-bit block size.

Joe Daemen, Rene Govaerts and Joos Vandewalle [5], Alex Biryukov, Jorge Nakahara Jr, Bart Preneel [4] and Philip Hawkes [7] have highlighted the need to change the key schedule of IDEA and they have found a number of weak keys through different methods with numbers varying from 2^{51} to 2^{64}.

Kelsey, Bruce Schneier and David Wagner [6] have suggested that the problem of weak keys and key attacks can be minimized in situations where random keys and a secure key distribution system are used.

2. NDEA

After an analysis of IDEA, some viable and feasible changes were made into it and so in this way NDEA came into being.

2.1 Random Number and the Ordering of the subkeys:

A random number is generated. The number of keys required in IDEA is 52, so it is ensured that random number is in the range 1 to 52. The subkey of that number becomes the first subkey. System date-time may also be used to generate the random number as it will always be unique. It acts as a seed for the random function. In the original IDEA, as previously mentioned, the key schedule is static but it becomes dynamic in the NDEA. Hence, the security increases. The NDEA Encryption algorithm requires three inputs – Plaintext, Key and a random number as against only two in the Original IDEA. So even if an eavesdropper is able to lay his hands on the ciphertext and the key, he cannot obtain the plaintext. To make it secure further, the random number can be encoded by a different encryption algorithm say RSA. This will add the complexity of the RSA with that of the IDEA and thus, make cryptanalysis attack even more difficult. So at the receiving end, the decryption process requires the following additional steps: The key has to be first reshuffled back to the original form using the random number and then fed into the decryption algorithm. The random number has to be applied on the key every time before decryption, as it is a different number always, to obtain the subkeys in the correct order. This is illustrated in the operational diagram Fig.3.

2.2 Operational Diagram

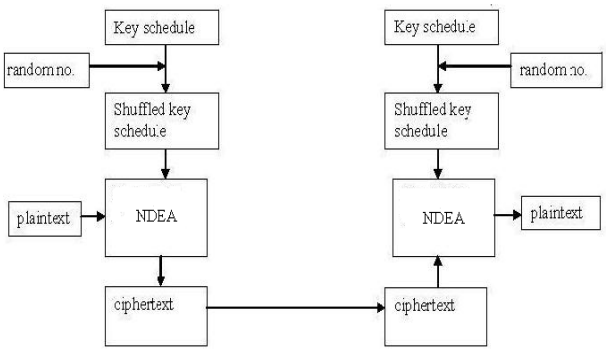

Fig. 3 Operational Diagram

As it is seen in fig. 3, the original key schedule is shuffled by the introduction of the random number. The subkey whose serial number is equal to the random number, becomes the first subkey of the schedule. Now this new ordered schedule is fed into the NDEA along with the plaintext and the ciphertext is generated. While decryptingn the reverse procedure is followed. The same random number used in

encryption is applied to the shufflrd key schedule to get the original schedule. This schedule and ciphertext are fed into the NDEA to obtain the plaintext.

2.3 Secure Usage of Keys

Suying Yang, Hongyan Piao, Li Zhang and Xiaobing Zheng [1] have suggested one method in secure usage of keys which has been incorporated in NDEA. This is also in lieu with the Kerkhoff's Principle.

Keys are kept in a database at a local or a remote host. The user need not carry or remember his key but only a simple key ID. He first needs to validate himself through a username and a password and after successful validation, has to enter the ID of the key. The corresponding key will be available for encryption. In such a system a user can keep several keys with different IDs. The security increases manifold if the sender sends the encrypted data with the key ID instead of the actual key. An eavesdropper's access to the key ID will not raise any security concern. Off course, sharing of username and password by the sender and the receiver is a priori.

Table 2: Users

User Id	User Name	Password
001	ripu	rp1234
002	rajat	gr7890

Table 3: User Keys

ID	User Id	key	Key Value
1	001	k1	abcd1234efgh5678
2	001	k2	1234abcd5678efgh
3	002	k1	abcdefgh12345678
4	002	k2	12345678abcdefgh

3. Discussion and Results

This section has two parts. The first part compares IDEA with NDEA and the second analyses the cipher texts obtained by both the algorithms.

3.1 Comparisons of IDEA and NDEA

Plaintext : abcd1234
Key : abcdefgh12345678
Random number: 8

Table 4: Comparison of time taken

	IDEA	NDEA
Cipher Text	101101000110111 010010110100110 001001101010011 100111000000111 0011	011010000001011 011110100111111 010000011100001 111010110100101 1111
Time Taken in Encryption	1781 ms	1789 ms
Time Taken in Decryption	1609 ms	1640 ms

- There is not much difference in time taken by the new IDEA and the original IDEA for shorter plaintexts. For longer plaintexts it is variable.
- The time calculated also includes the time for printing the result.
- The time varies irregularly with different plaintexts, keys and random numbers. The time consumed increases and decreases by the size of input also. Here, in this application the time calculated also included the time for printing the result.

3.2 Analysis of cipher text of both the algorithms by changing plaintext by a single character

Plaintext : abcd1235
Random number: 28
Key: (same as 3.1)

Table 5: Comparison of ciphertext

	IDEA	NDEA
Cipher Text	101101000110111010 010110100110001001 101010011100000101 1110101010	101001100001100011 000101110001101100 100111001000000011 1101001000

It is clearly observed that that changing a single character in plaintext changes the cipher texts in NDEA drastically (comparing with the result of 3.1) keeping the key same. In IDEA the first 22 characters in both cases (in the example taken) remain the same but in NDEA change in bit pattern is observed from the very first bit. The dynamism introduced is due to the different random numbers generated in both the cases. There is no appreciable difference in time taken for decryption and encryption.

Weak keys are those keys with a certain value for which the block cipher exhibits poor level of encryption. For instance encrypting twice with the same key yields the plaintext itself. In DES, there are four such weak keys. IDEA also has such weak keys whose numbers are found up to 2^{64}. The presence of weak keys have an obvious impact on the security of the block cipher and this issue as illustrated in [4], [5] and [7] can also be successfully addressed to by NDEA. If a weak key is used in NDEA then the original key schedule will no longer remain the same as it will be modified by the random number which is generated differently each time the algorithm is run and hence, the same key which is weak in IDEA may not remain so in NDEA.

4. Conclusion and Future Work

NDEA is an algorithm which holds the properties of the block ciphers and aimed to boost up the security of various real life applications. In this work we have tried to incorporate the goodness of IDEA and go even beyond it by introducing randomness and at the same time keeping the time and space constraints to a minimum. Initial experiments show that time taken for encryption and decryption depends on the length of the plaintext. An appreciable security enhancement is observed in the cipher texts obtained by the NDEA while making slight modification in the plaintext. This is in lieu with the confusion and diffusion properties of cryptography. The security can be further increased by using the key ID approach as the user need not remember his keys or carry them. In future intensive analysis is required for its use as an application. The algorithm may be used in encrypting an entire document and in e-mail based applications.

5. References:

1. Jia Chen, Dongyue Xue and Xuejia Lai: "An Analysis of IDEA Security Against Differential Cryptanalysis"
2. Suying Yang, Hongyan Piao, Li Zhang and Xiaobing Zheng: "An Improved IDEA Algorithm Based on USB Security Key", IEEE, 2007
3. Bruce Schneier: "Applied Cryptography", 2005
4. Alex Biryukov, Jorge Nakahara Jr, Bart Preneel, Joos Vandewalle: "New Weak-Key Classes of IDEA", 4th International Conference, ICICS, 2002
5. Joe Daemen, Rene Govaerts and Joos Vandewalle: " Weak Keys for IDEA", Advances in Cryptology, 1993
6. Kelsey, Bruce Schneier, David Wagner, "K ey-Schedule Cryptanalysis of IDEA, G-DES, GOST, SAFER, and Triple-DES", 1997
7. Philip Hawkes, "Differential-Linear Weak Key Classes of IDEA", Springer-Verlag, 1998
8. Nick Hoffman, "A Simplified IDEA", Journal: Cryptologia, 2007
9. Wikipedia

6. Authors

Rajat Goel is pursuing M.Tech. in Computer Science & Engineering, with specialization in information security, from Central University of Rajasthan, India. He obtained his B.E. degree from Seedling Academy of Design, Technology and Management, University of Rajasthan in 2008. Then for two years he gained a varied experience in industry and academia. He worked as an IT professional at IBM India, Chennai. He is an OCA and has one research paper to his credit. His areas of interest include database, MIS and cryptography.

Ripu R. Sinha is currently associated as an Assistant Professor with School of Computer and System Sciences at Jaipur National University, Jaipur and pursuing PhD Computer Science from NIMS University. Prior to joining JNU Jaipur, he was a SAP Consultant at ERP Technologies, India. Sinha has an extensive industrial experience and has participated, presented and published research papers in conferences at global level and he is holding diversified knowledge in the field of information technology management. His expertise and research interest focuses in the optimization of ERP, Knowledge management Practices, Cryptography, Cloud computing, Web intelligent System and Database technology for Industry.

Dr. Om Prakash Rishi was born on August30, 1971, in a small village of Rajasthan (India). Dr. Rishi completed his secondary examinations from the board of secondary education Rajasthan in 1986 and graduation from University of Rajasthan in 1991. He earned his First Class M.Tech degree in Computer Science in 2000 from Birla Institute of Technology, Mesra, Ranchi (India), and Doctor of Philosophy (PhD) in Computer Science in March 2009. Dr. Rishi's academic credentials were burnished by the years he spent on the faculty of Birla Institute of Technology, Mesra, Ranchi (India), Banasthali Vidhyapeeth, Banasthali Rajasthan (India) and c urrently working with the Central University of Rajasthan (India) which is one of the most prestigious Universities in India and established by MHRD, Govt. of India. Dr. Rishi's area of interest is Artificial Intelligence, Intelligent Systems, Cloud Computing and Information Security. He is member of IEEE, Computer Society of India, and several other National and International professional bodies.