Variable Size Block Encryption using Dynamic-key Mechanism (VBEDM)

K.C.Shyamala Bai
M.Tech Student
Dept. of E&C Engineering
Malnad College of Engineering,
Hassan, Karnataka, India

Dr.M.V.Satyanarayana
Principal and Professor
Dept. of E&C Engineering
Malnad College of Engineering,
Hassan, Karnataka, India

Dr. P.A. Vijaya
HOD
Dept. of E&C Engineering
Malnad College of Engineering,
Hassan, Karnataka, India

ABSTRACT
The weak point of the existing block encryption scheme is that the plain text or encryption key could be easily exposed differential cryptanalysis or linear cryptanalysis, which is mostly used for decoding block encryption. This is because the encryption schemes have been designed for the fixed size encryption key. Another weak point of the existing block encryption algorithm is that it has a fixed permutation table and fixed number of encryption rounds.

In order to overcome these weaknesses, an encryption algorithm using unlimited size of key and dynamically changing permutation table should be designed. A new encryption technique called Variable size Block Encryption using Dynamic-key Mechanism (VBEDM), which is designed with unlimited key size, dynamically changing permutation table based on the encryption key and variable block size for each round. To make the cryptanalyst hard to expose the plain text, from the array of compression algorithms the VBEDM uses a compression technique based on key. The compression used is not for compressing the text but for strengthening the encryption method. Because of its dynamic functionality in input block size, key size, permutation, number of rounds and compression it makes the crypt analyst too hard to analyzing the cipher text. This algorithm also uses a compression technique from an array of compression algorithm resulting in more confusion to the analyst.

General Terms
Information Security Cryptographic Algorithms, Block Encryption Mechanism

Keywords
Symmetric Encryption, Variable Size Block Encryption, Cryptographic Algorithms, Dynamic Key Mechanism.

1. INTRODUCTION
The method of encryption and decryption is called a cipher. Some cryptographic methods rely on the secrecy of the encryption algorithms; such algorithms are only of historical interest and are not adequate for real-world needs. Instead of the secrecy of the method itself, all modern algorithms base their security on the usage of a key; a message can be decrypted only if the key used for decryption matches the key used for encryption.

There are two classes of key-based encryption algorithms, symmetric (or secret-key) and asymmetric (or public-key) algorithms. The difference is that symmetric algorithms use the same key for encryption and decryption (or the decryption key is easily derived from the encryption key), whereas asymmetric algorithms use a different key for encryption and decryption, and the decryption key cannot be derived from the encryption key.

Symmetric algorithms can be divided into stream ciphers and block ciphers. Stream ciphers encrypt a single bit of plaintext at a time, whereas block ciphers take a number of bits (typically 64 bits in modern ciphers), and encrypt them as a single unit [5, 8].

Modern cryptographic algorithms are no longer pencil-and-paper ciphers. Strong cryptographic algorithms are designed to be executed by computers or specialized hardware devices. In most applications, cryptography is done in computer software. Generally, symmetric algorithms are much faster to execute on a computer than asymmetric ones. Variable size Block Encryption using Dynamic-key Mechanism (VBEDM) is a symmetric encryption algorithm. The key size [1] is not fixed the crypt analyst has to search the many possibilities. Even though if the crypt analyst gets the some likely key format, the VBEDM produces the more difference for the plaintext because of its unlimited key size generation by using cyclically variable positional reading of bits from the generated key bits stream, encryption is applied for different block size [3] with variable permutation and variable round complex function.

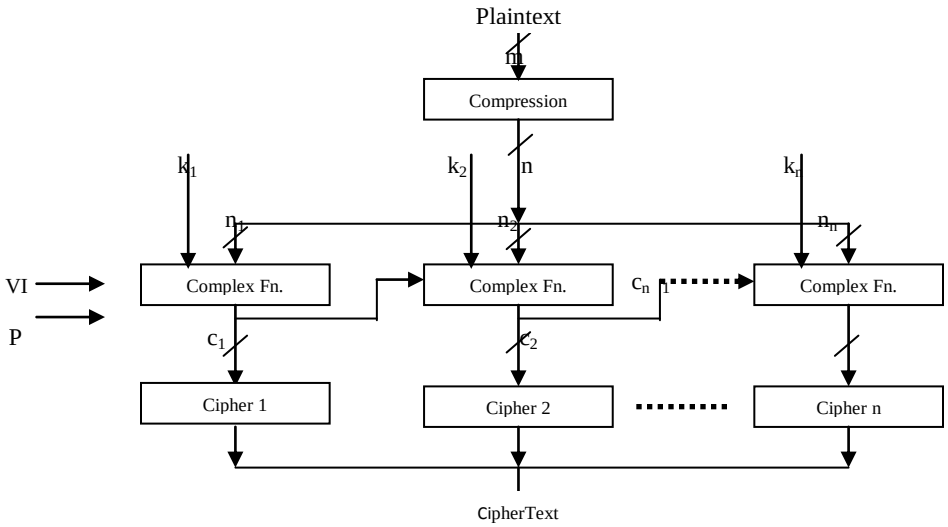

Fig 1: Architectural design

2. SYSTEM DESIGN

m = Size of the plaintext
n= size of the compressed plaintext
$n_1,n_2,..n_n$ be variable size of the block of compressed plaintext
$c_1,c_2,..c_n$ be the block of cipher text
$k_1,k_2 \ldots k_n$ be the keys

$$Compressed\ msg = n1 \,||\, n2 \,||\, \ldots. \,||\, nn$$

$$Cipher\ Text = Cipher\ text1 \,||\, Cipher\ text2 \,||\, \ldots. \,||Cipher\ textn$$

$$Cipher\ Text(i) = F(ki, ni)$$

$$Cipher\ Text = F(k1, n1+VI)||F(k2, n2+C1)||\ldots. \\ ||F(kn, nn+Cn\text{-}1)$$

Fig 1 shows the plaintext is compressed by an algorithm. Compressed data the data is taken for encryption and it is divided into variable size block by using key. Each block is encrypted based its size and present key (ki). The encryption round is changed based on its block size for example if the block size 16 bit then there are 8 rounds. The current block plain text is XOR with the previous block cipher text to avoid the reputations. Where the size of two blocks may differs so that, If size of current block size is greater than that of previous block then whole bits from previous cipher text is taken for processing the XOR is taken up to size of previous cipher text [2] and remaining bits are unchanged. If size of current block size is less than that of previous block then required number of bits from previous cipher text is taken for processing. This process is continued until end of the plaintext for encryption. The out of XOR of each block is given for complex function [4] where the encryption is performed based on block size and key the pseudo function, function table and dynamic permutation are performed.

3. EXPERIMENTAL RESULTS

3.1 Variable Block Encryption

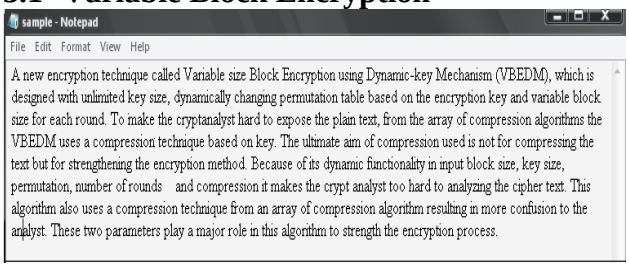

Fig 2: Snapshot showing the Plaintext

Fig 2 shows the file to be encrypted with the data; the plaintext is compressed by an algorithm. Compressed data the data is taken for encryption and it is divided into variable size block by using key. In this scheme, the length of the key size is unlimited i.e. user can select the key size accordance to their requirements. If very high security is needed, then lengthy key size could be selected. In key generation, first each character in the key is converted into binary code.

3.2 Key Management

A key check value for the combined components should also be available as a final check when the last component is entered. A problem that occurs with depressing regularity in the real world is when it is necessary to re-enter a key from its components. This is always an emergency situation, and it is usually found that one or more of the key component holders cannot be found. For this reason it is prudent to arrange matters so that the components are distributed among the key holders in such a way that not all of them need to be present. For example, if there are three components (C1, C2, C3) and three key holders (H1, H2, H3) then H1 could have (C2, C3), H2 could have (C1, C3) and H3 could have (C1, C2). In this arrangement any two out of the three key holders would be sufficient. In more sophisticated systems the components may be held on smart cards.

3.3 Key Generation

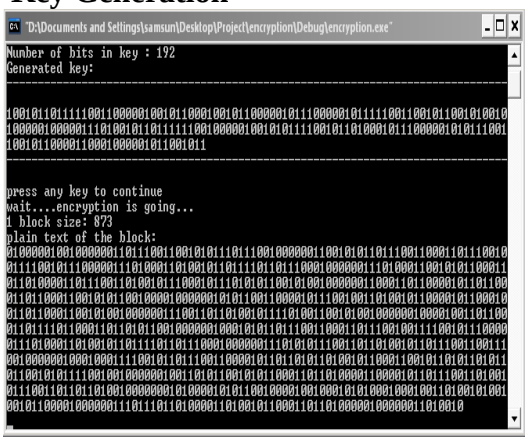

Fig 3: Snapshot showing the Compressed Plaintext

Fig 3 shows the length of the key size is unlimited; user can select the key size accordance to their requirements [7]. If very high security is needed, then lengthy key size could be selected. In key generation, first each character in the key is converted into binary code. The binary value of first character is NOT XOR with second character and second character with third character and so on. The same process is repeated cyclically for all characters of the key. If n is the number of characters in the key, then after NOT XOR operation, we get 2n set of binary values. The resultant 2n set of binary values is again XOR cyclically to get another 2n set of binary value. Finally 4n set of binary values. A transposition of these 4n set of binary values result in the encryption key.

1. Let us consider the key which contains three characters x_1, x_2, x_3. These three characters are converted into ASCII binary value $x_{1,8}$, $x_{2,8}$, $x_{3,8}$ and bit wise NOT XOR operation is performed cyclically to obtain $y_{1,8}$, $y_{2,8}$, $y_{3,8}$

$$y1,8 = x1,8 \; NOT \; XOR \; x2,8$$
$$y2,8 = x2,8 \; NOT \; XOR \; x3,8$$
$$y3,8 = x3,8 \; NOT \; XOR \; x1,8$$

2. The result is rearranged in the form $x_{1,8}$ $y_{1,8}$ $x_{2,8}$ $y_{2,8}$ $x_{3,8}$ $y_{3,8}$
3. Bit wise XOR operation is performed cyclically

$$z1,8 = XOR(x1,8, y1,8)$$
$$z2,8 = XOR(y1,8, x2,8)$$
$$z3,8 = XOR(x2,8, y2,8)$$
$$z4,8 = XOR(y2,8, x3,8)$$
$$z5,8 = XOR(x3,8, y3,8)$$
$$z6,8 = XOR(y3,8, x1,8)$$

4. Bit wise transposition is performed on the resultant 12 set of binary values in the following way

$$(z1,1, x1,1)\,(z1,2, x1,2)\,(z1,3, x1,3)\,(z1,4, x1,4)$$
$$(z1,5, x1,5)\,(z1,6, x1,6)\,(z1,7, x1,7)\,(z1,8, x1,8)$$

$$(z2,1, y1,1)\,(z2,2, y1,2)\,(z2,3, y1,3)\,(z2,4, y1,4)$$
$$(z2,5, y1,5)\,(z2,6, y1,6)\,(z2,7, y1,7)\,(z2,8, y1,8)$$

$$(z3,1, x2,1)\,(z3,2, x2,2)\,(z3,3, x2,3)\,(z3,4, x2,4)$$
$$(z3,5, x2,5)\,(z3,6, x2,6)\,(z3,7, x2,7)\,(z3,8, x2,8)$$

$$(z4,1, y2,1)\,(z3,2, x2,2)\,(z3,3, x2,3)\,(z3,4, x2,4)$$
$$(z3,5, x2,5,)\,(z3,6, x2,6)\,(z3,7, x2,7)\,(z3,8, x2,8)$$

$$(z5,1, x3,1)\,(z3,2, x2,2)\,(z3,3, x2,3)\,(z3,4, x2,4)$$
$$(z3,5, x2,5)\,(z3,6, x2,6)\,(z3,7, x2,7)\,(z3,8, x2,8)$$

$$(z6,1, y3,1)\,(z3,2, x2,2)\,(z3,3, x2,3)\,(z3,4, x2,4)$$
$$(z3,5, x2,5)\,(z3,6, x2,6)\,(z3,7, x2,7)\,(z3,8, x2,8)$$

3.4 Key Scheduling

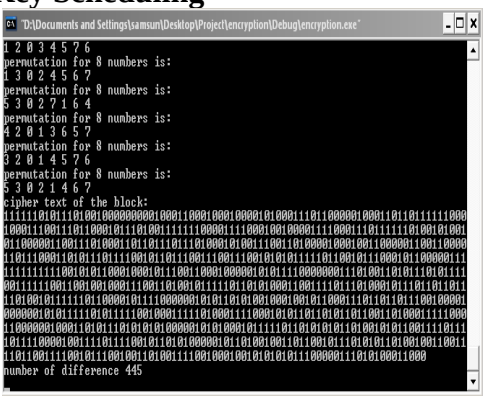

Fig 4: Snapshot showing the Compressed Cipher Text

Fig 4 shows the each block is encrypted based its size and present key (ki). The encryption round is changed based on its block size for example if the block size 16 bit then there are 8 rounds. The current block plain text is XOR with the previous block cipher text to avoid the reputations. Where the size of two blocks may differs so that, If size of current block size is greater than that of previous block then whole bits from previous cipher text is taken for processing the XOR is taken up to size of previous cipher text and remaining bits are unchanged. If size of current block size is less than that of previous block then required number of bits from previous cipher text is taken for processing. This process is continued until end of the plaintext for encryption. The out of XOR of each block is given for complex function where the encryption is performed based on block size and key the pseudo function, function table and

dynamic permutation are performed. it makes the crypt analyst too hard to analyzing the cipher text [6,9,10, 12].

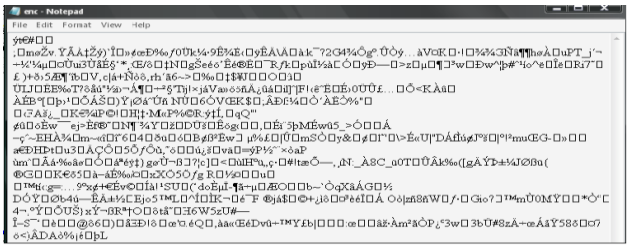

Fig 5: Snapshot showing the Cipher Text

Fig 5 shows the variable size block encryption cipher text using the dynamic key mechanism.

4. CONCLUSION

The plaintext is compressed by an algorithm. Compressed data the data is taken for encryption and it is divided into variable size block by using key. In this scheme, the length of the key size is unlimited each block is encrypted based its size and present key (ki). The encryption round is changed based on its block size for example if the block size 16 bit then there are 8 rounds. The current block plain text is XOR with the previous block cipher text to avoid the reputations. Where the size of two blocks may differs so that, If size of current block size is greater than that of previous block then whole bits from previous cipher text is taken for processing the XOR is taken up to size of previous cipher text and remaining bits are unchanged. If size of current block size is less than that of previous block then required number of bits from previous cipher text is taken for processing. This process is continued until end of the plaintext for encryption. The out of XOR of each block is given for complex function where the encryption is performed based on block size [11] and key the pseudo function, function table and dynamic permutation are performed.

5. FUTURE SCOPE

In the future work, the effectiveness of modified Variable Size Block encryption using dynamic Key Mechanism can be assessed under more complex information security environments.

Information requires high security, the number of bits changed for the each character during the encryption can be increased so that, it becomes very difficult for the cryptanalyst to break the cipher text to plaintext, as variable size block encryption using the dynamic key mechanism is influenced by a one-time padding technique.

6. ACKNOWLEDGMENTS

I acknowledge and express sincere thanks to our beloved principal and also my guide for his resourceful guidance, timely assistance and graceful gesture **Dr. M.V.Sathyanarayana,** Professor & Principal, Department of Electronics & Communication Engineering, for having supported us in our academic endeavors, **Dr.P.A.Vijaya,** Professor and Head of Department, Department of Electronics & Communication Engineering, for the facilities and support extended towards me and **Mrs. Sujatha B. R,** Asst Professor**,** Department of Electronics & Communication Engineering, for her kindly assistance throughout my academic endeavors.

7. REFERENCES

[1] Hamid Mirvaziri, Kasmiran Jumari Mahamod Ismail and Zurina Mohd Hanapi, "Message Based Random Variable Length Key Encryption Algorithm" Journal of Computer Science 5 (8): 573-578, 2009.

[2] Howard M. Heys, "Analysis of the Statistical Cipher FeedbackMode of Block Ciphers", IEEE TRANSACTIONS ON COMPUTERS, VOL. 52, NO. 1, JANUARY 2003.

[3] Zhu Shun-le , Wang Ya-xiang , Li Xin , "Design and Analysis of Variable-length Block Encryption Based on Chaos Particle Swarm" , Hefei, China. August 24–27, 2010.

[4] Marc Joye, "On White-Box Cryptography", Security of Information and Networks, pp. 7-12, Trafford Publishing, 2008.

[5] John B. Lacy, Donald P. Mitchell, William M. Schell CryptoLib: Cryptography in Software, AT&T Bell Laboratories.

[6] Luke O'Connor, A Differential Cryptanalysis of Tree-Structure Substitution-Permutation Networks.

[7] Chang-Doo Lee, Bong-Jun Choi, Kyoo-Seok Park, Design and evaluation of a block encryption algorithmusing dynamic-key mechanism,Future Generation Computer Systems 20 (2004) 327–338, http:// www.elsevier.com/locate/future/.

[8] Alfred J Menezes, Paul C van Oorschot, Scot A Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.

[9] E Biham, New Types of Cryptanalytic Attack using Related Keys, Journal of Cryptology, vol.7 No. 4, 1994, pp.229-2

[10] E Biham and A Shimir, Differential cryptanalysis of DES-like cryptosystem, Journal of Cryptology, 4 (1991), 3-72.

[11] Wu Wenling, Feng Dengguo, Zhang Wentao. Design and Analysis of Block Cipher (Second Edition) [M].Tsinghua university Press, 2009.

[12] Wil Michiels, Paul Gorissen, and Henk D.L Hollmann, "Cryptanalysis of a Generic Class of White-Box Implementation", SAC 2008, LNCS 5381, pp. 414-428, 2009.