A Research Threshold Efficient Hybrid Encryption Schema for Secure File System

P Penchalaiah, M Vijay Kumar, K Ramesh Reddy

Abstract: ***Electronic technologies have radically changed the way of doing business and the way of storing and sharing information. In the technology revolution, security is more affected with the introduction of distributed environment where data requires accessing from remote computers. When Operating Systems (OS) are running on a device with no file encryption, normally file access is controlled through user authentication mechanisms by its own operating system. However, an invader can easily bypass this fence if he gains physical access to the computer. To protect sensitive files/data even if physically accessed the device, the most widely accepted solution to this is to store the files encrypted on the physical media. Windows family of OS uses an EFS, similarly Gnupg is an encryption tool for operating systems such as FreeBSD, Solaris, MacOS to enforce security measure and much more algorithm which are using short length key or keys for encrypting the file-system. Even though these systems are good, still there is a question about confidentiality and privacy because of explosive growth of computing power and cryptanalysis tools and techniques. In this paper a novel approach for encrypting file system using Rbits cipher with PKC is proposed. In brief, Rbits enforce unlimited sub-keying mechanism in encryption process to avoid any sort of cryptanalysis.***

Index Terms: File-system, Encryption, Rbits, Random bits, Blum integer

I. INTRODUCTION

The limitations of storage space and computing power of devices pare down the ability of file/data sharing between terminals and increases the risks [10]. The techniques used to meet the security requirements like confidentiality and privacy are observed to be quite complex [8]. Some schemas use symmetric encryption which uses a common shared secret key directly in encryption processes. Single-key alone or combination of limited nonce is used to encrypt the entire message which leaves enough evidences to cryptanalyst [11]. Further, if the encryption process is performed using a single key has implicit threshold effect on maximum of data units that can be securely encrypted [5][7].

In general data is maintained in the form of files whose access is controlled by its device OS. But this barrier can be bypass when invader gains physical access to the bare storage device. Physical access to file system is possible when a storage device is removed and put it in another computer as a passive one, or simply booting the device from a separate bootable source like CD, pendrive having an OS. In this case the local filesystem is accessible in full unprotected manner. To protect sensitive files/data even if physically accessed the device, the most widely accepted solution to this is to store the files encrypted on the physical media [4][6].

EFS, Gnupg, CryFS etc are the some of the most predominant encryption tools for operating systems such as Windows, FreeBSD, Solaris, MacOS to enforce security measure. Furthermore various cryptanalysis methods like brute-force, frequency analysis, differential cryptanalysis, liner cryptanalysis etc enable attacker to find the clues about file-system [1] or content with little efforts because of vulnerabilities in the key that is used in encryption. In general most of the encryption algorithm makes use of fixed length key size in encryption process to keep fewer overheads due to key maintenance [3].

Even though these systems are good, still there is a question about confidentiality and privacy because of explosive growth of computing power and cryptanalysis tools and techniques [9]. This paper introduces a novel approach for file system encryption using Rbits cipher with PKC. In brief, Rbits enforce unlimited sub-keying mechanism in encryption process to avoid any sort of cryptanalysis. The main objectives of the novel approach are:

- To propose a novel hybrid encryption scheme that provides confidentiality especially for preserving data sensitivity
- To increase the threshold number of encryptions or even to completely remove the threshold
- Faster in encryption and decryption process
- Easy and light weight computations to encrypt and decrypt the data
- Strong avalanche effect

II. METHODOLOGY

A. Re-Keying

The strength of any cryptosystem depends upon the make use of unpredictable quantities like the continuous supply of key-stream in OTP, the large prime numbers selection in RSA, sub-keys in DES encryption etc. for which the quantities used must be adequate in size and primarily must be “random”. Because the probability of any particular quantity that is being chosen must be adequately enough to prevent adversary effects of cryptanalysis based on any type of estimation. Most of the common attacks success is based on the ability to get lots of clues from the encryptions under a single key.

Revised Manuscript Received on July 10, 2019

P Penchalaiah, Narayana Engineering College, Nellore, A.P, India. (penchal.caliber@gmail.com)

M Vijay Kumar, Rayalaseema University, Kurnool, A.P, India. (vijaykamesh33@gmail.com)

K Ramesh Reddy, Vikrama Simhapuri University, A.P, India. (drkrreddy05@gmail.com)

In order to produce a more resistance ciphertext, a number of randomly generated keys in a cipher can be used [5]. This would help to cover any perceptible patterns in the produced ciphertext. Re-keying in a cryptosystem tremendously increase the threshold on encryptions or even can completely remove the threshold, makes infinite number of encryptions without requiring a new exchange of keys repeatedly. Because different random bits eliminate repeating patterns, consequently avoids framing clues to a cracker that are required to break the key.

B. Rbits

Rbits aims to provide confidentiality especially for large data with the use of random bits. In brief, Rbits algorithm generates random bits which are continuously supplied as sub-keys to encryption process. These random bits (Rbits) are derived based on Ck (core-key) parameters and for re-keying Rbits make use of Blum integers [2][5].

Rbits applies sub-keys in two different modes S2S and S2M [2]. In Single sub-key per single block of data (S2S) mode is each data block DBi is encrypted with its own unique sub-key SKi. For example first data block DB_1 is encrypted using SK_1, second data block DB_2 is encrypted with SK_2, DB_3 with SK_3 and so on. In Single sub-key per a many of block of data (S2M) mode a certain number ‘l’ of data blocks is encrypted using SK_1 and then switch to SK_2. Once “l’ data blocks have been encrypted under SK_2 it switches to SK_3 and so on. A S2S mode is robust than the S2M mode because in S2S every block of data has its own key. But in terms of speed S2M is better than S2S because savings in subkey generation. For the application which requires both forward and backward security S2S mode is recommended.

C. Encryption/Decryption Process

The proposed method is a hybrid technique. The key used to encrypt the file system is symmetric which is encapsulated under asymmetric encryption. Here the symmetric key is core-key ‘Ck’ of Rbits is called as REK (Rbits Encryption Key). The proposed method uses a symmetric encryption technique because it is efficient in terms of time to encrypt/decrypt large amount of data than asymmetric key algorithms. The REK is then encapsulated in its own file after encrypting with public key of the file owner. To decrypt the file, the REK is de-capsulated and then decrypted by its corresponding private key. The decrypted REK used to decrypt the file. The following notations give the mathematical view of the cryptosystem.

Encryption process $C_f = \mathbf{E}(P_f, REK) \| \mathbf{E}(REK, PU_k)$

Decryption process $P_f = \mathbf{D}(C_f, [\mathbf{D}(REK, PR_k)])$

Where

E-Encryption process;	D	-Decryption process
C_f- Encrypted file;	P_f	-Plain file
PU_k-Public key;	PR_k	-Private key;

III. ANALYSIS & RESULTS

A. Cost Analysis

The cost of random bit generation analyzed for hardware implementation of BBS is as follows [5].

No of Bits	*Time (µsec.) @ 100 KHz*
160 bits	1,850
512 bits	5,270

Table: 1 Random bit generation (H/W Implementation)

Further Cost analysis simulation is performed on windows-7 OS, 3.0GHz Processor. The The algorithm is applied on different sizes of file sizes in bytes and various test runs experimental results shows that the time taken for encryption/decryption is 33.1 mille sec.

B. Differential Analysis

For differential effect investigation we developed a Java program which takes actual ciphertext and the resultant ciphertext produced after the key/data tamper as input and finds the number of bits changed. We performed avalanche effect in terms of both data and key sensitivity.The tables 2(a) and 2(b) shows the key sensitivity dependency in terms of percentage change of bits occurred in ciphertext. The tables are as follows.

Text # (s)	**% change in bits**
Test-1	47.28
Test-2	49.37
Test-3	49.09
Test-4	50.62
Test-5	53.82

Table 2(a). Sensitivity dependency of ‘s’

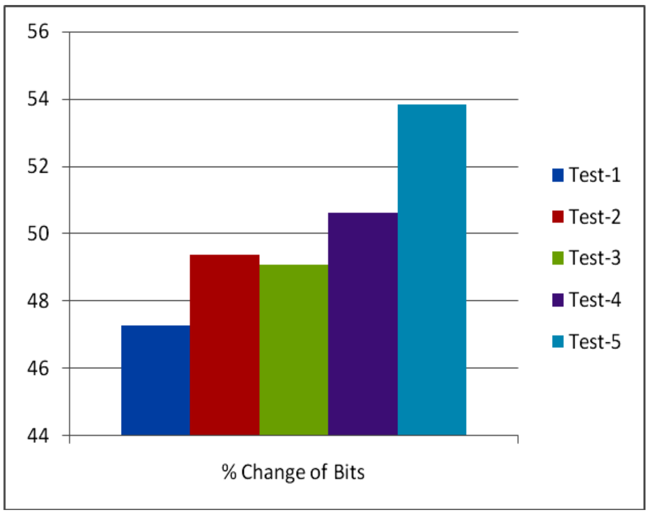

Fig 1.(a) Chart depiction of Sensitivity dependency of *s*

Text # (n)	**% change in bits**
Test-1	48.95
Test-2	51.32
Test-3	50.34
Test-4	48.26
Test-5	48.53

Table 2(b). Sensitivity dependency of *n*

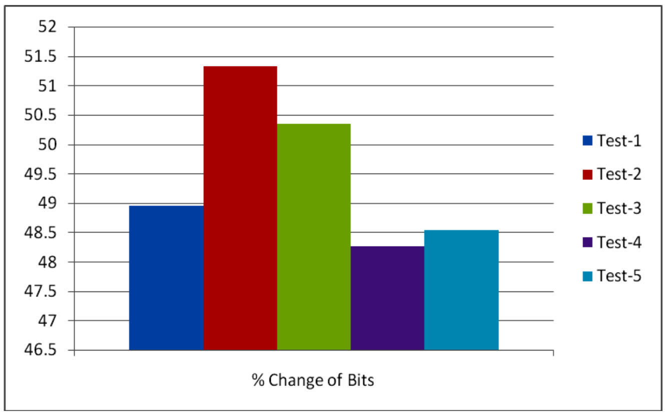

Fig 1.(b) Chart depiction of Sensitivity dependency of *n*

Percentage change of bits in ciphertstream *per_chg* is calculate as follows.

$$per\_chg = \left(\frac{nbc}{tnb}\right) \times 100$$

where *nbc* -no of bits changed ; *tnb* -total number of bits
For instance for Test-1= (340/719)*100=47.28
From the above statistics the average change effect of key and are listed in the table 3 and figure 2.

Seed Element	*Average Change of bits*
s	50.04
n	49.48

Table 3: Average key sensitivity of {s, n}

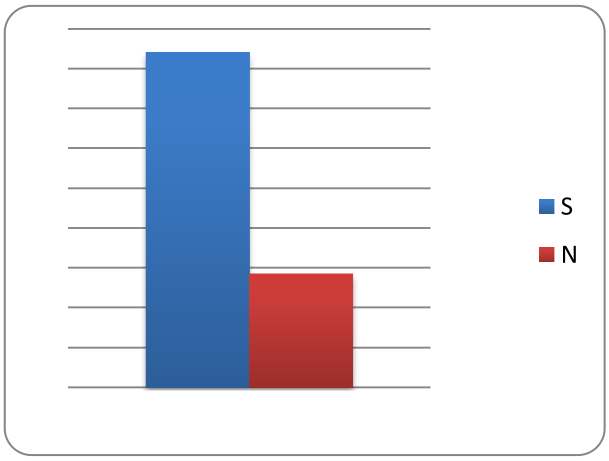

Fig 2: Chart for average of key sensitivity of {s,n}

From the above key sensitivity analysis we can confirm that a small change in the key produces extremely variant ciphertext. Especially from the table 3 show that change of a bit in ‘s’ permitting to produce at least more than 50% of the change in ciphertext. Similarly a change in element ‘n’ also having the same affects as in ‘s’ which is almost 50%. Similarly to key sensitivity test, data sensitivity is also analyzed by changing the actual plaintext in different positions repeatedly and resultant statics are recorded.

Test#	*Percentage of Sensitivity*
Test Run -1	62.59
Test Run -2	30.6
Test Run -3	61.2
Average	51.46

Table 4: Percentage of Data sensitivity

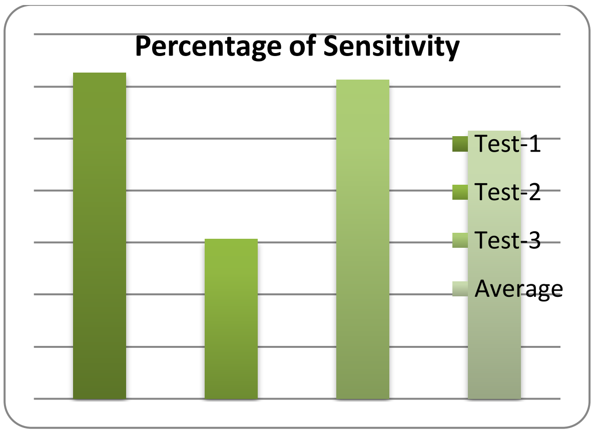

Fig 3. Chart dipiction for the table 4

From this data sensitivity analysis which confirms that known ciphertext attack is infeasible on Rbits by simply knowing the block of cipher blocks. From the above statistical analysis, we observed that Rbits satisfies diffusion and confusion characteristics of cryptographic information theory of Shannon, because the sensitivity is more than 50% for the both the cases of key and data sensitivity.

C. Speed Analysis

Most of the file encryption algorithms like Gnupg, EFS, CryFS uses AES, RC4, DES, 3DES algorithms as core engines to process the plaintext. We performed an encryption speed comparative study on Rbits with RC4, DES, 3DES and the resulted speed statistics are recorded. The test speed runs are recorded for the key of length 64 bits, 128 bits and 256 bits.

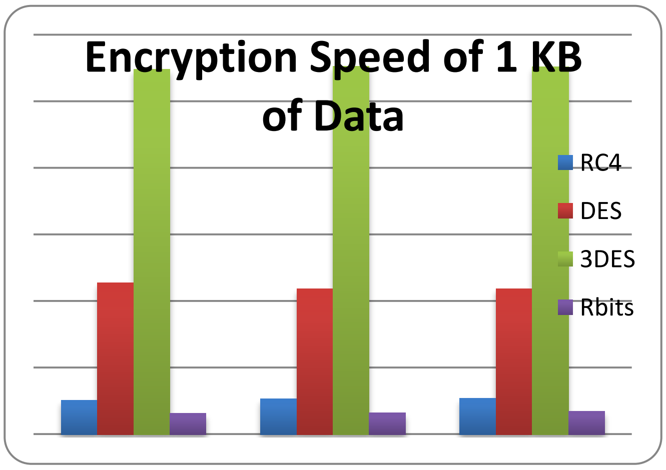

Fig 4. Encryption Speed for 64 bits, 128 bits and 256 bits

From the analysis, the speed (time taken) of the algorithm in ascending order as

$$Rbits < RC4 < DES < 3DES$$

Furthermore, Rbits is faster in encryption by 1.6 times than RC4, 6.7 times than DES and 16.6 times than 3DES, that speed of Rbits is efficient than RC4, DES, 3DES.

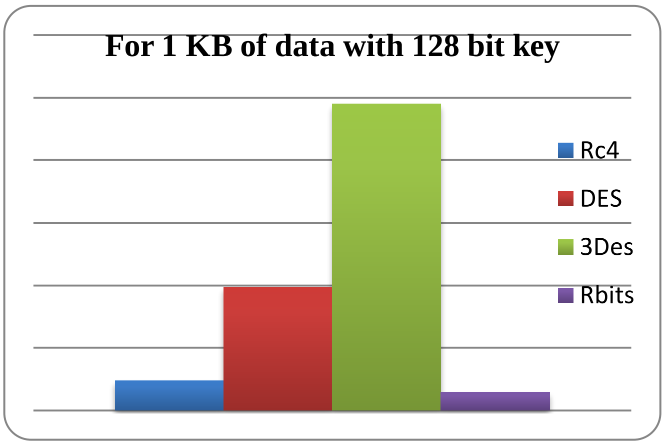

Fig 5. Avg. Speed with 128 bits key

CONCLUSION

The proposed hybrid schema Rbits enhances the security level of the confidential files even if the source storage device passively connected. This hybrid method achieves enhanced confidentiality and privacy by introducing an encapsulated master key which rekeys continuously to entire file. This would help to cover any perceptible patterns in the resultant ciphertext in the file, because re-keying eliminates repeating patterns. Further the proposed method is faster in terms of speed, has strong avalanche effect (key and data sensitive) and robust and secure.

REFERENCES

1. Ajay Kumar, Ankit Kumar, "File Encryption System Based on Symmetric Key Cryptography", *International Journal of Engineering and Computer Science*" Vol 5 Issue 03, pp. 16050-16057, 2016.
2. Penchalaiah P, Ramesh Reddy K, "Random Multiple Key Streams for Encryption with Added CBC Mode of Operation", *Perspectives in Science journal, (ELSEVIER)*, Vol 8, pp.57-62, April 2016.
3. Abhishek Joshi et al. "An Efficient Cryptographic Scheme for Text Message Protection against Brute Force and Cryptanalytic Attacks", ICCC 2015, *ELSEVIER*, pp. 360-366, 2015.
4. Eoghan Casey, et al. "The growing impact of full disk encryption on digital forensics", *Digital Investigation, ELSEVIER*, Volume 8, pp. 129–134, Nov- 2011,
5. Penchalaiah P, Ramesh Reddy K, "Secure and Cost Effective Cryptosystem Design Based on Random Multiple Key Streams", *Journal of Information Security Research, DIFR Publisher,* Volume 7, Number 1, pp. 29-40, March 2016.
6. Guy-Armand Yandji et al. "Research on a Normal File Encryption and Decryption" *International Conference on Computer and Management* (CAMAN), IEEE, 2011.
7. Gang Hu, "Study of file encryption and decryption system using security key", *International Conference on Computer Engineering and Technology (ICCET)*, IEEE, 2010.
8. Punam V. Maitri et al. "Low latency for file encryption and decryption using BRA algorithm in network security", *International Conference on Pervasive Computing (ICPC)*,IEEE 2015.
9. Dominik Leibenger, "EncFS goes multi-user: Adding access control to an encrypted file system", *Communications and Network Security* (CNS), IEEE, 2016.
10. Li Yang, Ziyi Han, "A remotely keyed file encryption scheme under mobile cloud computing", *Journal of Network and Computer Applications* (ELSEVIER), Volume 106, 15 March 2018, Pages 90-99.
11. LiuYepeng, RenYongjun etl, "A CCA-secure multi-conditional proxy broadcast re-encryption scheme for cloud storage system", *Journal of Information Security and Applications*, Volume 47, Pages 125-131, 2019