Int. J. Advance Soft Compu. Appl, Vol. 7, No. 3, November 2015
ISSN 2074-8523

Lightweight Symmetric Encryption Algorithm In Big Data

Majid Bakhtiari[1], Anazida Zainal[2], Saeid Bakhtiari[2] , and Hazinah Kutty Mammi[2]

[1]Advanced Informatics School
Universiti Teknologi Malaysia, Kuala Lumpur, Malaysia
e-mail:bakhtiari@utm.my
[2]Faculty of Computing,
Universiti Teknologi Malaysia, Johor, Malaysia
e-mail: anazida@utm.my, bsaeid3@live.utm.my
and hazinah@utm.my

Abstract

Comprehensive coverage of network has enabled many applications to be online. Growth in technology has produced IoT where almost every gadget is Internet enabled and this has produced massive data. Many data analytics tools and techniques have been developed to mine the data and get meaningful information out of it. With this sophisticated tools, there is a possibility that data are leaked, learned, tampered and shared beyond anticipation of data owner. Therefore, security and privacy issue is critical to big data. Due to its extraordinary scale, traditional encryption algorithms cannot be used to provide privacy protection in big data environment. Big data requires efficient encryption and decryption algorithms, encrypted information retrieval, attribute based encryption to provide data confidentiality and integrity. The focus of this paper is on data encryption to protect user privacy in big data environment. We proposed a symmetric lightweight encryption algorithm for faster encryption and decryption. The algorithm can support encryption for multitude of data in huge shared environment like cloud by providing efficient and fast encryption/decryption processes.

Keywords: *Symmetric Encryption, Lightweight, Big data, Efficient, security, Fast Encryption.*

1 Introduction

Modern technology has enabled people around the world to be connected and do online transactions and activities are aided by sensors. Continuously, there is an increase in the volume and detail of data captured, such as the rise of social media, Internet of Things (IoT), multimedia, has produced an overwhelming flow of data in either structured or unstructured format. Data produced from communications, these transactions and activities happened in every millisecond. Each day, there is about 2.5 quintillion bytes of data created[1]. Big data is described by its velocity, volume and its variety. Data comes in stream, huge quantity and it is in different formats.

Big data is characterized by 3V's; volume (data are numerous), variety (data cannot be categorized into regular relational databases and velocity (data are generated, captured and processed rapidly). The terms, volume, variety and velocity was introduced by Gartner to describe the characteristics of big data. The term big data technologies describes a new generation of technologies and architectures, designed to economically extract value from very large volumes of a wide variety of data, by enabling high-velocity capture, discovery , and /or analysis[2]. Big data brings the benefits of providing more opportunities to mine knowledge from this massive data and therefore, more accurate information can be obtained and predicted. However, big data has brought with it some security challenges[3][4]. The value of information is solely depends on authenticity of the data and the success of mining data will also depends if users are willing to share their data. Sharing the data will not happen if users are worried that their data can easily be leaked or tampered. In general, to achieve authenticity and confidentiality, the basic protection mechanism is signature and encryption[4]. Besides, traditional encryption is not practical to encrypt massive data because encryption will take a long time to encrypt huge data. Therefore, it degrades system performance and may affect users' satisfaction towards the service provided by data storage (i.e in cloud this is referred as SLA-service level agreement). Big data requires a fast encryption algorithm as not to impose lengthy retrieval, processing and storing of data. This paper is focusing on lightweight encryption that can provide efficient and fast encryption/decryption on massive data. The rest of the paper is organized as follows. Section 2 presents related works. Section 3 discusses on our cipher design. Section 4 focuses on cipher operation. Section 5 evaluates and tests the strength and security of our proposed lightweight encryption algorithm. Finally section 6 concludes the paper.

2 Related Works in Data Privacy and Cipher Design

To secure massive data from insiders as well as outsiders is a major challenge in big data. Preventing data leakage during transmission and processing is another challenging task. Therefore, privacy preserving is among the major concern when data becomes massive and demand for better processing and storing.

Wei *et al* [4] solved the confidentiality and authenticity in big data by proposing a new identity based generalized signcryption scheme. The scheme offers encryption, signature and both as big data may require different security services.

The authors claim that the proposed scheme is suitable to be used in big data environment since it does not have complicated certificate management compared to traditional cryptographic schemes. Unfortunately the scheme utilizes asymmetric encryption algorithm. Most of the public key infrastructure encryption algorithms suffer from lengthy encryption or decryption process. Apparently this is undesirable when handling massive data. Besides, by adopting Diffie –Hellman key exchange algorithm, it is susceptible to man-in-the-middle attack. Meanwhile, Cheng *et al.*[5] opts for using hash values and argue that encryption and decryption processes are time consuming and not suitable to be used in big data environment. The authors claim that their work focuses on protecting the privacy for cloud big data tenant. The proposed a big data preprocessing model that decompose data into *n*-parts according to some criteria (i.e data type). Each part will be hashed. Data parts will be packed into data blocks then uploaded to the cloud storage centers. These hash values are used to claim the data parts from the cloud storage. This approach of protecting the privacy is focusing on just providing data integrity but it doesn't provide privacy and it cannot prevent data leakage. Addressing the issue of privacy from different angle, Shrivastva *et al* [6] use differential privacy which is a noise based approach in providing privacy for big data. Differential privacy is defined as an approach where the probability of output of two different data sets will be nearly be the same. To create similar output, their approach perturbs the output. The amount of noise embedded into the output is proportional to the sensitivity level of the data set. Therefore, adversary cannot determine the targeted data set by any quasi identifier.

Finally, popular public key cryptographic systems like ECC and RSA suffer from lengthy encryption process and decryption process respectively.

Meanwhile, Stream ciphers are very popular due to their many attractive features: they are generally fast, can typically be efficiently implemented in hardware, have no error propagation, and are particularly suitable for use in environments where no buffering is available and/or plaintext elements need to be processed individually. Recent years have witnessed an increase in the research of design and analysis of stream ciphers, primarily motivated by eSTREAM, the ECRYPT Stream Cipher Project[7]. eSTREAM was a multi-year project, which started in 2004, and had the objective of selecting a portfolio of promising stream cipher designs. According to specific applications identified for stream ciphers of algorithm design, there are classified in two usage portfolio as follows:

i) Profile 1: stream ciphers for software applications with high throughput.

ii) Profile 2: stream ciphers for hardware applications with highly restricted resources.

The project received 34 submissions, of which 16 were selected to the final phase[8]. The final portfolio was announced in April 2008, containing eight ciphers; four in profile 1 and four in profile 2 [9]. However, the portfolio was later revised, due to new cryptanalytic results [10] against one of the selected ciphers in profile 2 (namely, the F-FSCR-H stream cipher). Despite the end of the eSTREAM project, the research area of analysis and design of stream ciphers remains active, with particularly eSTREAM portfolio ciphers continuing to attract much attention of the cryptographic community [11].

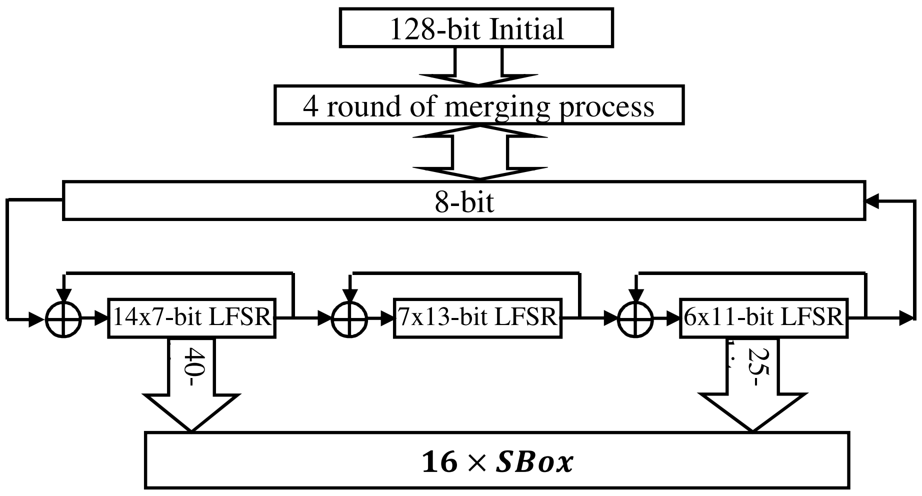
Big data paradigm demands for faster and efficient encryption/decryption process. Following the trend, this paper is proposing a new lightweight stream cipher. The algorithm presents a simple and potentially scalable design, and is particularly suitable for big data environment. In brief, this proposed algorithm takes a 263- bit secret key and a 128-bit IV (if desired) as input and generates for each iteration an output block of 32 random bits from a combination of 65 bits of parallel random number generator. Encryption/decryption is done by XOR the random data with the plaintext/ciphertext. This following section will discuss on Cipher Design and a brief overview of the main component (Design criteria, Parallel Random Number Generator, Substitution box, Initial Vector) will be given.

3 The Proposed Cipher Design

In this section, the main design criteria of lightweight stream cipher is explained. The major components of the design are; Parallel Random Number Generator, Substitution Box, Initial Vector and as well as the detail of the cipher specification.

3.1 Design Criteria

This algorithm is designed in two basic stages. There are parallel random number generator (PRNG) and cascaded substitution boxes. Fig. 1 shows basic diagram of the proposed algorithm.



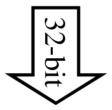


Fig. 1: Basic diagram of lightweight stream cipher algorithm

3.2 Parallel Random Number Generator

This part of the algorithm generates 65 bits stream from one linear feedback shift register, as shown in Fig 2. It should be noticed that an important advantage of this kind of LFSR configuration is isolation of each random bit streams from each other. While in normal LFSR, the sequence length of random stream bit is very close to each other, which is susceptible to correlation, and algebraic attacks of algorithm.

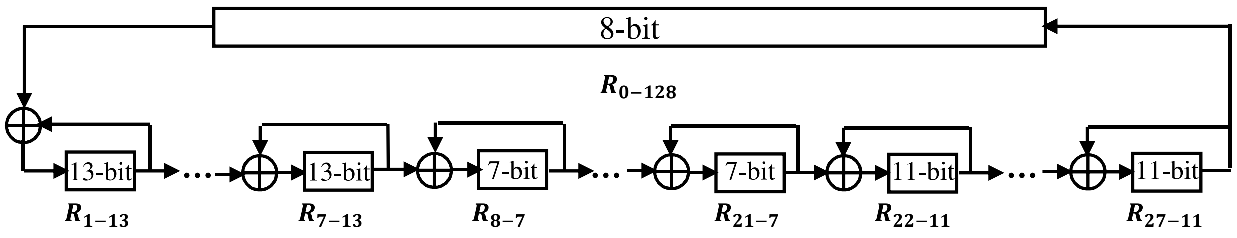


Fig. 2: Parallel Random Number Generator

Equation 1 shows the primitive polynomial equation of Fig 2, which is the main seed of parallel random number generator.

$$f(x) = x^{263} + (x^{13} + 1)^7 (x^7 + 1)^{14} (x^{11} + 1)^6 \tag{1}$$

3.3 Substitution Box

In this algorithm we implemented a total of 16 S-Boxes as shown in Figure 3. In fact, all of S-Boxes play a role of $\mathbb{F}_2^{65} \rightarrow \mathbb{F}_2^{32}$. With consider that each two bit of output generated by one S-Box and one extra bit (LSB) passed to next S-Box. Therefore, for each S-Box, four bit directly feed from PRNG. Fig 3 shows the combination of S-Boxes are put together to generate 32-bit random key stream vector. Table 1 shows the transfer mapping of each S-Box $\mathbb{F}_2^5 \rightarrow \mathbb{F}_2^3$.

Table 1: Transfer mapping of each S-Box

5-bit Input	**3-bit Output**
From 00000 to 11111	[07, 02, 01, 06, 04, 00, 03, 05, 00, 07, 06, 01, 05, 03, 02, 04, 03, 00, 04, 05, 07, 02, 06, 01, 00, 07, 01, 06, 05, 04, 02, 03]

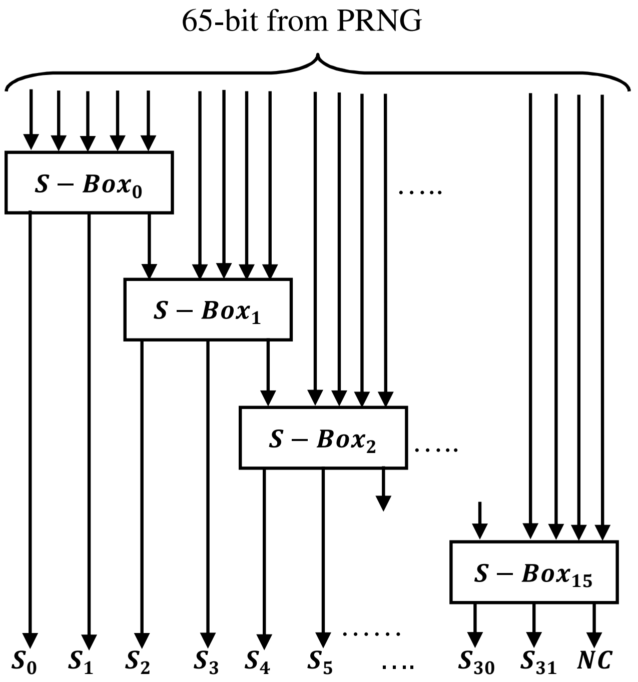

65-bit from PRNG
$S-Box_0$
$S-Box_1$
$S-Box_2$
$S-Box_{15}$
S_0 S_1 S_2 S_3 S_4 S_5 .... S_{30} S_{31} NC

Fig. 3: The S-Box configuration

3.4 Initial Vector

Stream cipher cryptosystems suffer from key management. It is because most of stream ciphers use XOR function for encrypting and decrypting. Therefore, if two massages are encrypted with one secret key, eavesdropper can obtain XOR'ed of two plain text as shows in Equation 2.

$$C_1 \oplus C_2 = (P_1 \oplus K_1) \oplus (P_2 \oplus K_1) = P_1 \oplus P_2 \qquad (2)$$

Therefore, initial vector (IV) and the method of merging it to secret key in stream ciphering are very important. In this regard, most stream ciphers use two keys to generate a key-stream sequence. They are secret key and an additional parameter named the initial value (IV), which generally broadcast in public. The initial vector is a block of bits that is required to allow one cryptosystem to be executed in several operational modes to produce a unique key-stream, independent from other key-streams that are produced by the same secret key, without changing secret key. However, the size of the IV is important. Usually it depends on the encryption algorithm in use.

Shannon published the concepts of confusion and diffusion as fundamental concepts for achieving security in cryptosystems [12]. Confusion is reflected in the nonlinearity of cryptosystem parts, with notice that the linear systems are generally easy to break. Diffusion is achieved by ensuring that a small change in the input is spread out to make a large change in the output. This part of algorithm engaged with secret key in founds of processing

3.5 Specification of the Proposed Lightweight Algorithm

This stream cipher algorithm has designed on the base of key variety equal to 2^{263} and Initial Vector 2^{128}. The encryption/decryption speed is >70 mega bit per second (mb/s) on normal personal computer. However, it can be increased until 100mb/s if this algorithm implemented on FPGA. The main structure of algorithm has designed on two major parts, which are parallel random number generator and cascaded substitution boxes.

4 Cipher Operation

In this section the key stream generation and initialization process will be described.

4.1 Key Stream Generation

This algorithm generates 32-bit random bit as key stream, in each iteration. In fact two bits of key stream are generating from one S-Box. Each S-Box is generating 3-bit which one least significant bit (LSB) is using by next S-Box. In other word, each 5-bit input data for a S-Box are equal to one bit from former S-Box and 4-bit directly from PRNG.

4.2 Initialization Process

Since the impact of initial vector (*IV*) is regarded as high, the initialization of the proposed lightweight algorithm is started with initialization of secret key with the length of 263-bit and the 128-bit initial vector (*IV*). The pseudo-code of the initialization algorithm is shown in Fig 4.

```
Input-1 : 263-bit of initial Secret Key
Input-2 : 128-bit of Initial Vector
Output : 32-bit key stream
Start of Initialization
    1. Divide IV to four 32-bit as IV1, IV2, IV3 and IV4
    2. Fill 263-bit of secret key to PRNG
    3. Run PRNG 13 clock pulses
    4. First 32 bit of PRNG = IV1 $\oplus$ (32 bits of latest
    generated key stream)
    5. Second 32 bit of PRNG =IV2 $\oplus$ ( second 32 bits of PRNG)
    6. Run PRNG 19 clock pulses.
    7. Third 32 bit of PRNG = IV3 $\oplus$ (32 bits of latest
    generated key stream)
```

```
8. Fourth 32 bit of PRNG =IV4 $\oplus$ ( fourth 32 bits of PRNG)
9. Run PRNG 127 clock pulses.
10. Fifth 32 bit of PRNG = IV4 $\oplus$ (32 bits of latest
generated key stream)
11.Run PRNG 127 clock pulses.

End of Initialization
```

Fig. 4: Initialization Algorithm

5 Security Evaluation and Test

For testing and evaluation, we used two well-known statistical package tests on the proposed algorithm by generating 100 samples of key streams. There are Diehard Suite and NIST Suite of statistical tests. These tests have been performed with key stream size of 10^8 - byte each, where each one was generated with different secret key. Afterward, we tested each output of key stream. However, different parts of algorithm were also tested by correlation immunity test, non-linearity test, algebraic test and autocorrelation test.

5.1 Statistical Test

The proposed algorithm has successfully passed all the 16 NIST Suite tests. Furthermore, each of S-Box bit have been tested separately. Table 2 shows the result of each test.

Table 2: Result of statistical test

Test Name	Result
Non-Linearity	=12 (Maximum level for 5-bit)
Correlation Immunity	Passed successfully
Algebraic Degree	= 4 (Maximum level for 5-bit)
Result of Statistical NIST tests	Passed successfully
Confusion and Diffusion test on IV & secret key	Passed successfully

5.2 Diehart Suite Tests

The algorithm has successfully passed all of Diehard Suite Tests. Diehard suite tests are the latest cryptography package test which consists of 15 tests proposed by George Marsaglia/Florida State University.

5.3 Comparison with Other Stream Ciphers

Some ciphers aim for high speed in software or hardware. The aim of the proposed algorithm is to provide high security level in comparison with other stream cipher algorithms in big data environment. Table 3 shows a bigger key length used by the proposed algorithm, can provide higher security while keeping internal state small. It is important to note that internal state bit is the number of memory bit cells used by the algorithm. As much as lower internal state has direct relationship to processing speed of the algorithm. Meanwhile, the encryption speed of the proposed algorithm is >70 mb/s in normal PC.

Table 3: Comparison between the proposed algorithm and other well-known stream cipher algorithms

Algorithm	**Key Length bit**	**IV Length bit**	**Internal State**
A5/1	54	114	64
A5/2	54	114	64
E0	$8 \leq K \leq 128$	48	202
RC4	$8 < K < 256$	24	2064
Grain v1 [13]	80	64	160
Trivium [13]	80	80	288
MICKEY-128 [13]	128	128	320
Grain-128 [14]	128	96	256
HC-128	128	128	N/A
Rabbit	128	64	513
Salsa-20	128	64	512
Sosemanuk	128	128	384
Proposed algorithm	*263*	*128*	*263*

6 Conclusion

Security and privacy issue is important for big data. It's 3Vs criteria demands a new encryption algorithm that can provide efficient and fast encryption/decryption process. This paper has proposed a lightweight stream cipher for big data encryption which can support >70 mb/s with security of 2^{263}. With this speed, encrypting data to be transmitted, processing and data at rest in big data environment will provide seamless data privacy protection. Finally, the proposed lightweight stream cipher has been tested with the standard testing procedure and it has outperformed other stream cipher algorithms in terms of security and it is suitable to be used in big data environment where users are fully confident that data leakage is hardly happen if not impossible.

ACKNOWLEDGEMENTS.
The authors would like to express their appreciation for the support of the sponsors of the Research Grant University (Project Vot No. PY/2014/03207) entitled "Copyright Protection of Digital Campus" from the Ministry of Education, Malaysia and Universiti Teknologi Malaysia, managed by the Research Management Center, Universiti Teknologi Malaysia.
We would also like to express our gratitude to the editor and anonymous reviewers who reviewed this paper.

References

[1] Cloud Security Alliance (CSA) 2012. Top Ten Big Data Security and Provacy Challenges.https://downloads.cloudsecurityalliance.org/initiatives/bdwg/Big_Data_Top_Ten_v1.pdf

[2] Cavoukian, A. and Jonas, J. 2012. Privacy by Design in the Age of Big Data. https://privacybydesign.ca/content/uploads/2012/06/pbd-big_data.pdf

[3] Indrajit, R, Srinath, T, V,S, Kilzer, A, Shmatikov, V, and Witchel, E. 2010. Airavat: Security and Privacy for Mapreduce. *NSDI*, 297-312

[4] Wei, G., Shao, J., Xiang, Y. Zhu, P. and Lu, R. 2015. Obtain Confidentiality or/and Authenticity in Big Data by ID-based Generalized Signcryption. *Information Sciences*.Vol. 318,11-122

[5] Cheng, H., Wang, W. and Rong, C. 2014. Privacy Protection Beyond Encryption for Cloud Big Data. *IEEE 2nd International Conference on Information Technology and Electronic Commerce (ICITEC 2014)* Dalian, China.188-191.

[6] Shrivasta, K., M., Rizvi, M., A. and Singh, S. 2014. Big Data Privacy based on Differential Provacy a Hope for Big Data. *IEEE 6th International Conference on Computational Intelligence and Communication Networks*. 776-781.

[7] Smart, N. 2010. *ECRYPT II yearly report on algorithms and keysizes (2009-2010). Framework*, www.ecrypt.eu.org/documents/D.SPA.13.pdf

[8] Robshaw, M. and Billet, O. 2008. *New stream cipher designs: the eSTREAM finalists*, Springer-Verlag New York Inc

[9] Babbage S.and Dodd, M. 2008 The MICKEY Stream Ciphers. In New Stream Cipher Designs: The eSTREAM Finalists, number 4986 in LNCS, Berlin, Heidelberg. Springer-Verlag. 191-209

[10]Hell, M. and Johansson, T. 2008. Breaking the F-FCSR-H stream cipher in real time. *Advances in Cryptology-ASIACRYPT 2008*, 557-569.

[11]Cid, C. and Robshaw, M. 2009. The eSTREAM portfolio 2009 annual update, eSTREAM, ECRYPT Stream Cipher Project, Tech. Rep., Jul 2009.

[12]Cusick, T. W. and Stănică, P. 2009. Cryptographic Boolean functions and applications, Academic Press.

[13]Gaj, K., Southern, G. and Bachimanchi, R. 2009. Comparison of hardware performance of selected Phase II eSTREAM candidates, eSTREAM, ECRYPT Stream Cipher Project, Report (2007)

[14]Hwang, D., Chaney, M., Karanam, S., Ton, N., and Gaj, K. 2008. Comparison of FPGA targeted hardware implementations of eSTREAM stream cipher candidates. *The State of the Art of Stream Ciphers*, 151–162.