References

1. Shevtsova L.N. Project workshop: textbook / L.N. Shevtsova; Krasnoyarsk State Agrarian University. - Krasnoyarsk, 2016. - 107p.

THE AVALANCHE EFFECT OF A NOVEL STREAM CIPHER WITH A 3D SPONGE STRUCTURE

Ikramov A. [1,2], Juraev G. [2]

[1]*Institute of Mathematics of Academy of Sciences of Uzbekistan, Tashkent, Uzbekistan*

[2]*National University of Uzbekistan, Tashkent, Uzbekistan*

E-mail: a.ikramov@mathinst.uz

Abstract

We develop the special stream algorithm based on the SPONGE structure in order to obtain a secure and fast symmetric encryption algorithm. We research the avalanche effect to decide the number of rounds of the algorithm. The algorithm uses a secret key of 512 or 1024 bits and produces the key stream consisting of blocks of 2048 bits each.

Introduction

Both the Republic of Kazakhstan and the Republic of Uzbekistan does not have their own standardized stream cipher and relies on methods provided by manufacturers of hardware. Block symmetric encryption algorithms are not good with transmission of large data in real time. For example, streaming video or audio in encrypted mode is only possible with stream ciphers. Thus, the development of a new stream cipher is an actual problem for our countries.

The perfect stream cipher should act as random number generator. Pseudo-random number generators built with algorithms such as RC4 [2] are generally significantly faster than those based on block ciphers. The RC4 algorithm is widely used in various information security systems, in computer networks (for example, in the SSL protocol, for encrypting passwords, etc.). The development of a new approach to hash functions introduced with Keccak (or SHA-3) has led to an increasing interest in using SPONGE structures in other cryptographic applications [1]. The popular stream cipher RC4 was modified to use SPONGE structure and was called Spritz [2]. The resulting algorithm is more robust than the initial RC4. While non-linear operations in Keccak are simple, we focused on already established S-box used in AES.

The SPONGE structure itself represents a large array (usually 2-D or 3-D) that consumes data gradually and returns a small piece of stored information. The size of the array is designed to be so large that it is practically impossible to brute-force it. As the returning data is not enough to reconstruct the internal state the whole structure becomes practically irreversible for intruders. We continue our research of SPONGE structured stream cipher [4].

Design of a new stream cipher

We designed our stream cipher as a SPONGE structure with internal state's shape of $17 \times 16 \times 32$ bits. We address each bit within internal state using $S_{i,j,t}$, where $i \in \{0,1,2,\dots,16\}, j \in \{0,1,2,\dots,15\}, t \in \{0,1,2,\dots,31\}$. Another representation of the same internal state is $B_{i,j,p} \in \{0,1,2,\dots,255\}^{17\times16\times4}$ where each number is stored in exactly one byte.

Each round consists of the following operations in the given order:

1. Adding Input. We use XOR (exclusive OR) to add an array of the same size as the internal state:

$$S = S \oplus Input$$

2. Substitution. As was mentioned above, we selected AES substitution table as Sbox for our cipher. This is the only non-linear operation of the cipher. This substitution table provides security against linear and differential cryptanalysis [8]. We previously analyzed other substitution tables [9, 10, 11].

Each byte in the internal state is replaced with its corresponding substituted value:

$$B_{i,j,p} = Sbox(B_{i,j,p})$$

*3. Multiplication*For each t we take matrix $A_t = \{S_{i,j,t}\}_{i,j}$ and perform multiplication $A_t = A_t \times M$. Next, we replace old values in S with new values formed by all A_t.

We fixed matrix M such that it has non-zero determinant and used as many bits as possible.

$$M = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

*4. Rotation.*This linear operation allows to spread bytes across the internal state. We do not change j indices as operation #3 is responsible for mixture of columns.

$$B_{i+j+p \bmod 17, j, p+j \bmod 4} = B_{i,j,p}$$

5. Additional mixing To increase mixture of different slices (third index of the internal state) we introduce adding operation:

If $(i + j)$ is odd, then we add to $B_{i,j,p}$ value of $B_{i,j,p-1}$ using XOR. We put $B_{i,j,-1} = 0xFF$.

If $(i + j)$ is even, then we add to $B_{i,j,p}$ value of $B_{i,j,p+1}$ using XOR. We put $B_{i,j,4} = 0x00$.

The algorithm uses 512 or 1024 encryption key to produce the key stream. We put $key_{j\times 4+p}$ values into $Input_{0,j,p}$ if key is 512 bit long. If it has 1024 bits, we put next 512 bits into $Input_{1,j,p}$.

If we use stream cipher in synchronized mode, other bytes of Input (starting from $Input_{2,0,0}$) are filled with R, N, where R is the number of rounds from start, N is the serial number of the key block to be produced (starting with 0). When $R = N = 0$, we put Initialization Vector instead. The inserted data can be padded using a tweak [5].

Output Key stream is formed after each 6 rounds using values $B_{i,j,p}$ where $i \in \{0,1,\dots,15\}$, $j \in \{12,13,14,15\}, p \in \{0,1,2,3\}$. This produces exactly 2048 bits of key stream. If the mode is synchronized, rounds continue, and the algorithm does not reset the internal state.

Results and discussion

Robust ciphers must meet several conditions. One of them is an avalanche effect when a change in one bit of initial data leads to change of half of bits in output in average.

To test the effect in the designed cipher we put Input to only zeros, the encryption key to zeros and run the cipher for different number of rounds. We also run cipher using the internal states that had only one bit equal to 1 for each of 8704 bits for the same number of rounds. Then we compared how many bits are different in the resulting internal states. The results of average percentages of number of bits that changed are presented in Table 1.

Table 1. Average percentage of number of bits that changed given one different bit in inputs

Number of rounds	Percentage of total bits changed, %
1	1.17
2	34.54
3	49.67
4	49.99

5	50

According to the results, 5 rounds are enough to guarantee avalanche effect. Therefore, we chose number of rounds of the algorithm equal to 6.

This work was supported in part by the project UZB-Ind-2021-98 — “Research and development of stream encryption algorithm”. We are grateful to Timur Abdullaev for his help in the development of the new stream algorithm.

References

1. *Aleksander B. Vavrenyuk, Victor V. Makarov, Victor A. Shurygin.* Synchronous Stream Encryption Using an Additional Channel to Set the Key, Procedia Computer Science, Volume 190, 2021, Pages 797-802, ISSN 1877- 0509.

2. *Ronald L. Rivest, Jacob C. N. Schuldt.* Spritz — a spongy RC4-like stream cipher and hash function. IACR Cryptol. ePrint Arch. (2016): 856.

3. *Bo Qu, Dawu Gu, Zheng Guo, Junrong Liu.* Differential power analysis of stream ciphers with LFSRs. Computers & Mathematics with Applications. Volume 65, Issue 9, p. 1291-1299 (2013).

4. *Alisher Ikramov, MirsaidAripov, GayratJuraev.* SPONGE structure in the basis of a new stream cipher. Conference: Modern problems of applied mathematics and information technologies al-Khwarizmi 2021: abstracts of the international scientific conference (15-17 November, Fergana, Uzbekistan). FTY Fergana. 2021. p.188

5. *A. Chakraborti, N. Datta, A. Jha, C. Mancillas-Lypez, M. Nandi, Y. Sasaki.* Elastic-Tweak: A Framework for Short Tweak Tweakable Block Cipher. In: Adhikari, A., Kusters, R., Preneel, B. (eds) Progress in Cryptology INDOCRYPT 2021. INDOCRYPT 2021. Lecture Notes in Computer Science, vol 13143. Springer, Cham. https://doi.org/10.1007/978-3-030-92518-5_6.

6. *D. Chung, S. Lee, D. Choi, J. Lee.* Alternative Tower Field Construction for Quantum Implementation of the AES S-box. Cryptology ePrint Archive, Report 2020/941 (2020).

7. *A. Makalesi, et al.* A New Pseudo Random Number Generator Design with LFSR Based 32-Bit Floating Point with High-Speed FPGA. Int. J. Adv. Eng. Pure Sci. 2020, 32(3): 219-228.

8. *H. Kruppa, SUA Shahy.* Differential and Linear Cryptanalysis in Evaluating AES Candidate Algorithms. Technical report, National Institute of Standards and Technology. 1998.

9. *Juraev G.U., Marakhimov A.R.* Representation of the block data encryption algorithm in an analytical form for differential cryptanalysis. International Journal of Innovative Research in Information Security (IJIRIS). 2019, Issue 03, Volume VI, p.38-42. DOI: 10.26562/IJIRIS.2019. MRIS10081.

10. *Juraev G.U., Ikramov A.A., Marakhimov A.R.* About differential cryptanalysis algorithm of block encryption “Kuznyechik”. International Journal of Advanced Research in Science, Engineering and Technology (IJARSET). Vol. 6, Issue 2, Feb. 2019. P.8164-8169.

11. *Timur Abdullaev, and GayratJuraev.* Development of a method for generating substitution tables for binary and ternary number systems. AIP Conference Proceedings 2365, 040003 (2021); DOI: 10.1063/5.0056841 Published Online: 16 July 2021.

12. *Timur Abdullaev, and GayratJuraev.* Selection of the optimal type of the gamming function for symmetric encryption algorithms. AIP Conference Proceedings 2365, 040004 (2021); DOI: 10.1063/5.0056843 Published Online: 16 July 2021.

MAIN FEATURES OF THE PYTHON PROGRAMMING LANGUAGE

Seitimbetova A.B., Tohmetova K.M.

Karaganda Buketov University, Karaganda, Kazakhstan

Karaganda Technical University named after Abylkas Saginov,Karaganda, Kazakhstan

E-mail:s_b_aigerim@mail.ru, kuralay_tokhmetova@mail.ru

In connection with the currently observed rapid development of personal computing, there is a gradual change in the requirements for programming languages. Interpreted languages are beginning to play an increasingly important role, as the increasing power of personal computers begins to provide sufficient speed for the execution of interpreted programs. And the only significant advantage of compiled programming languages is the high-speed code they create. When the speed of program execution is not critical, the most appropriate choice is an interpreted language, as a simpler and more flexible programming tool.

In this regard, it is of particular interest to consider the relatively new programming language Python, which was created by its author Guido van Rossum in the early 90s. The author of the