UDC 621.3.06

Linear transformation properties of ZUC cipher

R. I. Kiyanchuk, R. V. Oliynykov
Kharkiv National University of Radio Electronics, Ukraine

It is presented an analysis of ZUC cipher – a perspective stream cipher considered for use in evolving LTE standard for mobile communications. The finite state machine of ZUC cipher is researched. A linear transformation used in FSM was found to have undesirable impact on its cryptographic properties. And even though the cipher structure does not allow to exploit it, the transformation needs to be improved before been used for development of perspective cryptographic algorithms since it may lead to serious weakness.

Key words: *linear transformation, stream cipher, LTE, finite state machine.*

Представлено аналіз алгоритма ZUC – перспективного поточного шифру для використання в стандарті мобільного зв’язку LTE. Досліджено скінченний автомат, що використовується у шифрі. Виявлено, що лінійне перетворення скінченного автомата має небажаний вплив на його криптографічні властивості. Не зважаючи на те, що структура шифру не дозволяє використати даний недолік для атаки, перетворення необхідно вдосконалити перед використанням у інших перспективних криптоалгоритмах.

Ключові слова: *лінійне перетворення, поточний шифр, LTE, скінченний автомат.*

Представлен анализ алгоритма ZUC – перспективного поточного шифра для использования в стандарте мобильной связи LTE. Исследован конечный автомат, использующийся в шифре. Обнаружено, что линейное преобразование конечного автомата имеет нежелательное влияние на его криптографические свойства. Несмотря на то, что структура шифра не позволяет использовать данный недостаток для атаки, преобразование необходимо усовершенствовать перед использованием в других перспективных криптоалгоритмах.

Ключевые слова: *линейное преобразование, поточный шифр, LTE, конечный автомат.*

1. Introduction

Security of mobile communication systems fell far behind from what was state-of-the-art in modern cryptography. The A5/1 cipher used in GSM can be broken within hours using high-end hardware [1] and A5/1 Security Project uses a combined distributed rainbow table code book to decrypt GSM voice calls and text messages within seconds [2]. Communication over satellite phones has also been shown to be insecure after reverse engineering the proprietary ciphers GMR-1 and GMR-2 [3]. GMR-1 is a variant of A5/2 cipher (which is prohibited for implementation in mobile phones as of July 2007 [4]) and is vulnerable to the known ciphertext-only attack with an average case complexity of 2^{32}. GMR-2 is an original cipher, but its one session key can be recovered with 65 bytes of keystream at a moderate computational complexity.

The need of transition to secure ciphers in mobile communication systems is obvious. However, the adoption and deployment of new cryptographic algorithms in global systems takes a long time. Therefore the 3rd Generation Partnership Project (3GPP) decided to develop promissory cipher suite in case the need of transition for evolving LTE standard. While the first two confidentiality and integrity algorithm sets are based on SNOW 3G and AES, the innovative 128-EEA3 (encryption) and 128-EIA3 (integrity) algorithms are based on ZUC cipher. Its development has been handed over to Data Assurance and Communication Security Research Center of Chinese Academy of Sciences in order for Chinese authorities to permit its use in the country [5].

The cipher has been evaluated by Security Algorithms Group of Experts (SAGE) of European Telecommunications Standards Institute (ETSI) and two teams of independent experts as well. Afterwards it has been opened for public evaluation. Several cryptanalytic attacks on previous two versions of ZUC (January 2010 and June 2010) have been found in [6-7]. The improved version of July 2011 has been opened for additional public evaluation period because of nontrivial changes made to the algorithm.

The finite state machine (FSM) of ZUC is researched. A linear transformation used in FSM was found to have undesirable impact on its cryptographic properties. And even though the cipher structure does not allow to exploit it, the transformation needs to be improved before been used for development of perspective cryptographic algorithms since it may lead to serious weakness.

Analysis of ZUC showed there were several security ideas that help the cipher to resist the disadvantage of linear transformations:

1) combination of modular addition and XOR with different registers;
2) exchanging low and high half-words of registers inside FSM;
3) injection of reorganized LFSR words into FSM output.

2. Description of ZUC

The cipher consists of three layers: linear feedback shift register (LFSR), bit reorganization (BR) and non-linear function F implemented via the finite state machine (fig. 1).

LFSR structure is different from most shift registers used in stream ciphers. It has 16 stages of 31-bit words with feedback defined over $GF(p) = GF(2^{31} - 1)$. Unlike most ciphers based on LFSR, ZUC uses prime finite field instead of extension $GF(2^n)$. Consequently, zero stage is not allowed and is replaced by p. A primitive feedback polynomial is $f(x) = x^{16} - (2^{15}x^{15} + 2^{17}x^{13} + 2^{21}x^{10} + 2^{20}x^4 + (2^8 + 1))$, so the register has its period equal to $p^{16} - 1 \approx 2^{496}$.

Bit reorganization extracts 128 bits from LFSR state by fetching higher and lower 16 bits of certain 31-bit LFSR stages (1). Two of them are consumed by the FSM and the other two are xored with FSM output word.

$$X_0 = s_{15_H} \| s_{14_L}; \quad X_1 = s_{11_L} \| s_{14_H}; \quad X_2 = s_{7_L} \| s_{5_H}; \quad X_3 = s_{2_L} \| s_{0_H}. \qquad (1)$$

Non-linear function F is represented by a finite state machine with two 32-bit registers, a combination of two different $8 \to 8$ S-boxes and linear transformations L_1 and L_2. Words X_1 and X_2 are injected into the state of FSM while words X_0 and X_3 only affect its output. During the update of the FSM state operations XOR and addition modulo 2^{32} are used together with exchanging half-words between two parts of the state, which essentially strengthens the cipher as will be shown further. The FSM is designed to have high non-linearity, good diffusion properties and balanced output sequences.

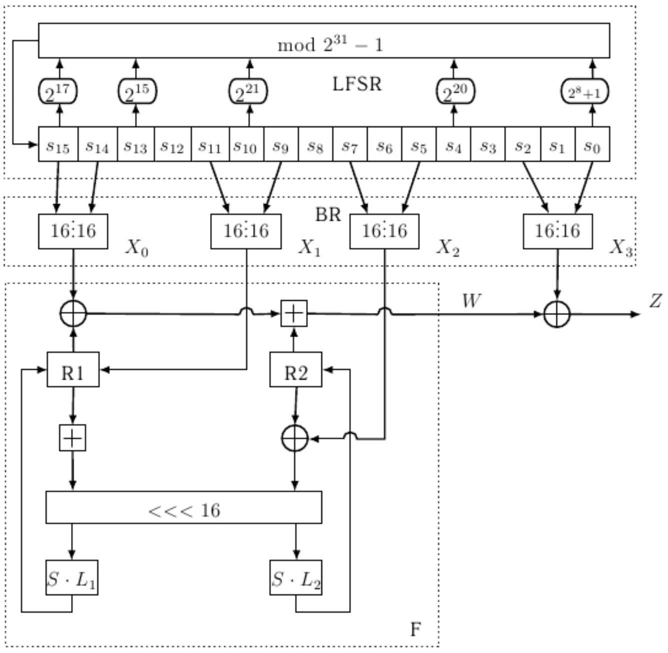

Fig. 1. ZUC structure

In many sources the substitution transformation used in FSM is described as 32×32 S-Box which is not accurate. The specification describes two S-Boxes S_0 and S_1 which are applied to a 32-bit word sequentially:

$$\begin{array}{ll} \text{bits } 0 \ldots 7 & \to S_1 \\ \text{bits } 8 \ldots 15 & \to S_0 \\ \text{bits } 16 \ldots 23 & \to S_1 \\ \text{bits } 24 \ldots 31 & \to S_0 \end{array}$$

Thus one can say that S-box S is composed of S-boxes S_0 and S_1: $S=(S_0,S_1,S_0,S_1)$. Therefore bits are diffused only within separate bytes, not the whole word. S-Box S_0 is designed using Feistel structure with three permutations (fig. 2). S-Box S_1 is generated using affine transformation equivalent to that of AES, but with different polynomial and constants [5].

The main idea behind the linear transformation in ZUC is high suitability for software and hardware implementations and good diffusion properties. Therefore, the transformation was defined using the quotient ring $GF(2)[x]/(x^{32}+1)$. A 32-bit word $a=a_{31}a_{30}a_{29}\ldots a_0$ matches an element $a(x)$ according to the bijection $\ddot{}$ from binary field extension $GF(2^{32})$ to quotient ring $GF(2)[x]/(x^{32}+1)$ as follows:

$$a = a_{31}a_{30}a_{29}\dots a_0 \rightarrow a(x) = \sum_{i=0}^{31} a_i x^i . \qquad (2)$$

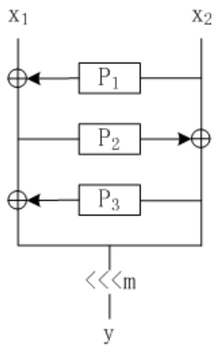

Fig. 2. Feistel network for S-Box construction

So the elements of linear transformation are represented by polynomials over the quotient ring. Two polynomials define linear transformations used in ZUC:

$$L_1 = x^{24} + x^{18} + x^{10} + x^2 + 1; \quad L_2 = x^{30} + x^{22} + x^{14} + x^8 + 1. \qquad (3)$$

A transformation is performed by multiplying the input polynomial by L_1 or L_2. Such transformations may be implemented by using cyclic shift and xor operations which are highly efficient both in software and hardware. ZUC developers also noticed that the matrix representation of $L_1(x)$ and $L_2(x)$ over $GF(2)$ happen to be transpose matrices of each other.

2.1. Initialization

Cipher key and initialization vector are split into bytes and combined with 15-bit constants defined in ZUC specification in order to load the key material into LFSR stages which are 31 bits long. Each 15-bit constant substring is an m-sequence generated by a primitive polynomial of degree 4 over the binary field $GF(2)$ [8].

Initialization is done by clocking the cipher for 32 ticks. During initialization the FSM output word W is consumed by the LFSR update. Since stages of LFSR are 31 bits long, the least significant bit of W is removed by right shift. If the resulting feedback value is 0, it is replaced by $p = 2^{31} - 1$ as zero stage is not allowed in $GF(2^{31} - 1)$.

2.2. Keystream generation

Keystream mode is similar to initialization, but the LFSR does not receive any input. The output W of non-linear function (FSM) is xored with X_3 (word extracted on bit reorganization layer) producing keystream word Z. The very first word produced by FSM after initialization is discarded. According to ZUC specification the maximum length of keystream generated from a single key is 65504 bits (or 8188 bytes). Such limitation is enforced by the LTE Service Data Unit (SDU) size [9].

3. Implication of the Finite State Machine

Analysis of the FSM revealed a drawback in linear transformations L_1 and L_2. Given a symmetric input word (that is its lower 16 bits are equal to higher 16 bits) the output of the transformation is also a symmetric word, thereby the power of outputs set is reduced to 2^{16}:

$$L_1\,(\text{0x34BC34BC}) = \text{0x785A785A};$$

$$L_2\,(\text{0xABCDABCD}) = \text{0x50C950C9}.$$

Equality in lower and higher half-words means that if the polynomial has a coefficient x^a, than it also has a coefficient x^{a+16}. Such behaviour of the linear transformation can now be explained using polynomial arithmetic in quotient ring. Consider an input polynomial of the form $x^a + x^{a+16}$ for transformation L_1. Any polynomial satisfying the described condition of symmetry may be chosen here but for the sake of clarity only two coefficients are considered. The transformation applies as follows:

$$\begin{aligned}
&L_1(x^a + x^{a+16}) = (x^a + x^a \cdot x^{16}) \cdot (x^{24} + x^{18} + x^{10} + x^2 + 1) = \\
&= x^a \cdot ((x^{24} + x^{18} + x^{10} + x^2 + 1) + x^{16} \cdot (x^{24} + x^{18} + x^{10} + x^2 + 1)) = \\
&= x^a \cdot (x^{24} + x^{18} + x^{10} + x^2 + 1 + x^{24+16} + x^{18+16} + x^{10+16} + x^{2+16} + x^{16}) = \\
&= x^a \cdot (x^{24} + x^{22} + x^{14} + x^8 + 1 + x^{40} + x^{34} + x^{26} + x^{18} + x^{16}) = \\
&= x^a \cdot (x^{26} + x^{24} + x^{16} + x^{10} + x^8 + 1).
\end{aligned} \tag{4}$$

So applying the linear transformation to any polynomial results to multiplication of x^a by $L_1 \cdot x^{16} + L_1 = x^{26} + x^{24} + x^{16} + x^{10} + x^8 + 1$. This polynomial is symmetric as clearly seen from its binary representation:

$$(x^{26} + x^{24} + x^{16} + x^{10} + x^8 + 1) = 0000\ 0101\ 0000\ 0001\ 0000\ 0101\ 0000\ 0001.$$

In fact the result of operation $L_1 \cdot x^a + L_1$ will always be symmetric in the chosen quotient ring not depending on L_1 or x^a. Such behaviour is easy to explain considering operations with binary representation of the polynomial. Multiplication by x^{16} in the quotient ring means left cyclic shift by 16 bits which exchanges halves of the word. Addition of polynomials in the quotient ring is just xoring their binary representation. It is now clear from figure 3 that such transformation will always lead to symmetric output since left and right parts of the word are xored with each other.

Following sections will consider the influence of such property on the cipher security.

4. Impact on Cryptographic Properties

The S-Boxes that follow linear transformation do not affect the symmetry of words since they diffuse bits only within separate bytes. However the analysis showed that there are still several obstacles on using the property for cryptanalysis. The most effective transformations that corrupt symmetric words inside the FSM are addition modulo 2^{32} and half-words exchanging.

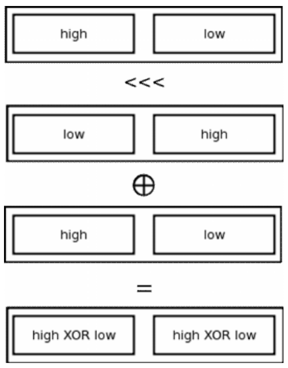

Fig. 3. Linear transformation of symmetric input

4.1. Symmetric Words in Bit Reorganization Layer

With the possibility to choose both the cipher key and the IV it is possible to make the words in Bit Reorganization layer symmetric. The following initialization:

s15	4D47AC**00**
s14	**00**789A8F
s13	**00**3C4D35
s12	**00**5E26D7
s11	269AF15E
s10	136BC49A
s9	78AF1313
s8	624D78E2
s7	0989AF6B
s6	3C7135AF
s5	57B5E226
s4	1AD789C4
s3	71135E4D
s2	44E26B89
s1	2F26BC**00**
s0	35C4D7**00**

leads to symmetry in word X_0 for 1 tick, word X_1 for 2 ticks, X_2 for 7 ticks and X_3 for 12 ticks. The highlighted bytes indicate that they may obtain any value and do not affect the symmetry in bit reorganization layer words. Yet only the first output word of FSM is symmetric because of the half-word exchanging, so registers are initialized by confused values after the first tick.

If addition modulo 2^{32} is replaced by xor and half-word exchanging is excluded, the Finite State Machine is filled with symmetric words. However, the cipher still resists to propagation of the linear transformation drawback due to injecting X_0 into FSM. Symmetry in X_0 is destroyed with the second tick because of the LFSR feedback which updates s_{15}. The effect of such initialization on generated keystream is analyzed further.

4.2. Randomness of Cipher States on Initialization

Several statistical properties of LFSR state and keystream are considered. Cross correlation between LFSR states after key loading and after initialization indicates if any non-random relations remain after the initialization procedure. Likewise, the cross correlation between LFSR state after initialization and the first 496 bits of keystream show if the starting keystream bits depend on the initial LFSR state. Correlation of random sequences oscillates near 0.5 value (shown on graphs with green line), so the larger deviation of correlation peaks from value 0.5, the higher dependency between them.

Serial correlation coefficient measures the dependency of bits in the sequence itself and obviously should be close to zero. Entropy shows the information density of the binary data (maximum possible entropy in this case is 1 bit of information per 1 bit of data). Expected value (or arithmetic mean) is the result of summing all the bits and dividing by the length of data. The value should converge to 0.5 in case of random data [10].

4.3. LFSR state after initialization

Results of statistical testing with corresponding values of keys and IVs are presented in tables 1-4.

Table 1. Statistical properties of LFSR state after initialization

Key	all bytes equal 0x00
IV	all bytes equal 0x00
Entropy	0.999295 per bit
Arithmetic mean	0.4844
Serial correlation coefficient	-0.047898
Max cross correlation peak deviation	0.0678
Min cross correlation peak deviation	0.2081

Cross correlation tests indicate noticeable deviations from random values, so it is possible to distinguish keys and IVs with long series of zeroes from other random initialization values (fig. 4). However, the corresponding keystream shows better results (table 2) and random cross correlation (fig. 5).

The correlation deviation is observed again (fig. 6) for a different non-random key and IV (table 3). But the corresponding keystream yet shows to be random.

In the following testing sample (table 4) the key and IV were chosen to generate symmetric words in Bit Reorganization layer. No correlation deviations or non-randomness signs have been found which indicates that the disadvantage of linear transformations is annihilated by the cipher structure.

Table 2. Statistical properties of first 496 keystream bits

Key	all bytes equal 0x00
IV	all bytes equal 0x00
Entropy	0.999840 per bit
Arithmetic mean	0.4926
Serial correlation coefficient	0.025578
Max cross correlation peak deviation	0.0616
Min cross correlation peak deviation	0.0596

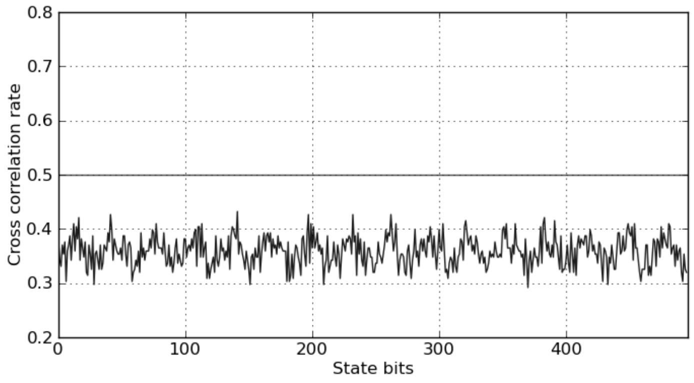

Fig. 4. Cross correlation between LFSR initial state and after initialization

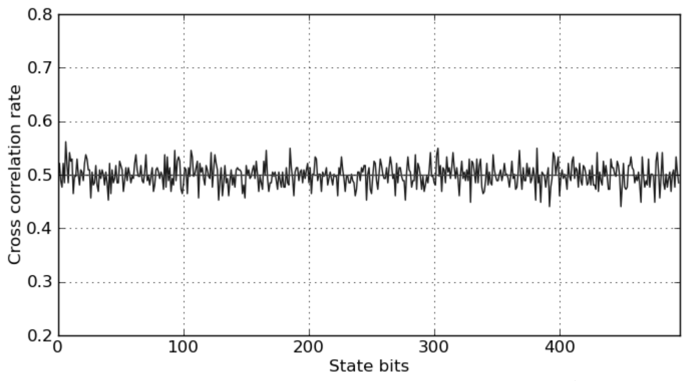

Fig. 5. Cross correlation between initialized LFSR state and the first 496 bits of keystream

Table 3. Statistical properties of LFSR state after initialization

Key	all bytes equal 0xFF
IV	all bytes equal 0xFF
Entropy	0.991353 per bit
Arithmetic mean	0.4453
Serial correlation coefficient	0.019521
Max cross correlation peak deviation	0.1387
Min cross correlation peak deviation	0.0509

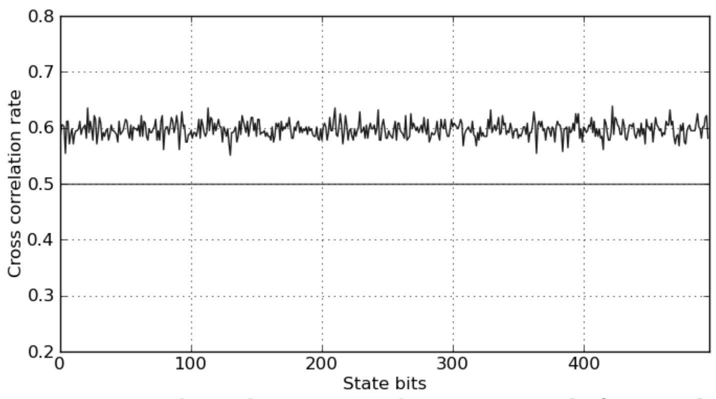

Fig. 6. Cross correlation between initial LFSR state and after initialization

Table 4. Statistical properties of LFSR state after initialization

Key	generates symmetric words in BR
IV	generates symmetric words in BR
Entropy	0.999725 per bit
Arithmetic mean	0.5098
Serial correlation coefficient	0.007434

Further testing showed the corresponding keystream is random as well. All tested random keys and IVs also resulted to good randomness of LFSR state and generated keystream.

If the key equals 0xFF and IV equals 0x0, the correlation graphic shows normal deviation (table 5) but periodic oscillation is observed (fig. 7).

Table 5. Statistical properties of LFSR state after initialization

Key	all bytes equal 0xFF
IV	all bytes equal 0x00
Max cross correlation peak deviation	0.0603
Min cross correlation peak deviation	0.0737

Even though entropy, arithmetic mean and serial correlation tests show good results, cross correlation testing of LFSR shows that it is possible to distinguish some weak keys analysing the LFSR state after initialization procedure. Non-random keys lead to bad randomization of LFSR state, but do not noticeably influence generated keystreams. The property of the linear transformations could not be used for an attack since the cipher structure diffuses bits and destroys symmetry in output words within 2 ticks at worst.

No evidence of non-randomness in keystream bits has been found, but the testing samples were too small for gathering enough statistics. Further the severe randomness testing of ZUC keystream is performed using the set of tests provided by NIST Statistical Test Suite [11].

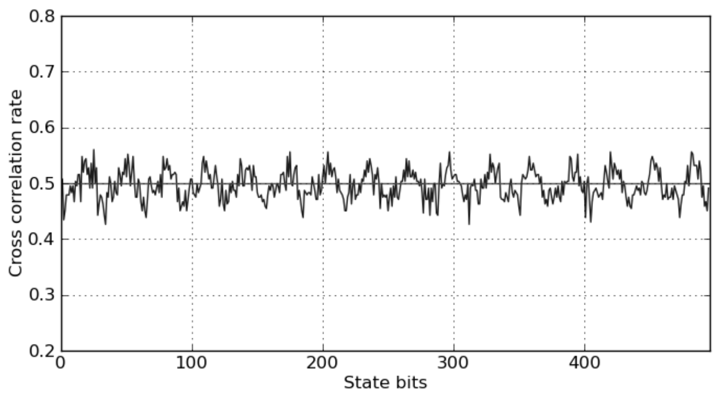

Fig. 7. Cross correlation between initial LFSR state and after initialization

4.4. Keystream Randomness

The cipher keystream has been analysed using test for randomness provided by NIST. In order to get correct statistical results one needs to provide at least 55 data samples and each sample should be at least 1000000 bits long. According to ZUC specification, the maximum length of keystream generated on a single key is 65504 bits, so there is lack of data for performing all 15 tests from NIST STS. Random Excursions and Random Excursions Variant tests are impossible to perform with the specified amount of keystream data and therefore have been excluded from evaluation. These tests actually represent 26 tests with different parameters so the total amount of executed NIST STS tests is 163 instead of all 189.

Three keystream data sets have been tested for randomness: generated from random keys, from keys with long bit series, and from LFSR state that injects symmetric words into BR layer. Results of testing the keystream randomness are shown on figures 8-10 and total statistics are shown in table 6. Success rate shows how much samples passed certain test (1 means all samples passed). The red line marks minimum success rate in order to consider the sequence to be random.

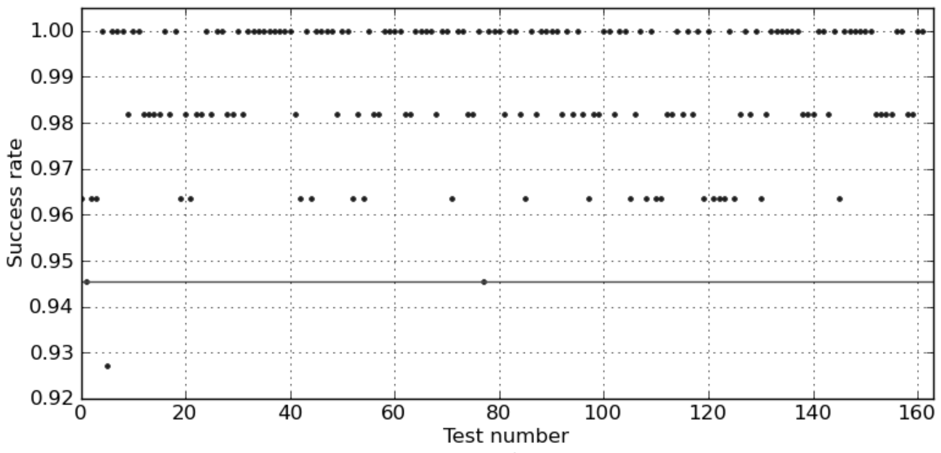

Fig. 8. Testing results for random keys

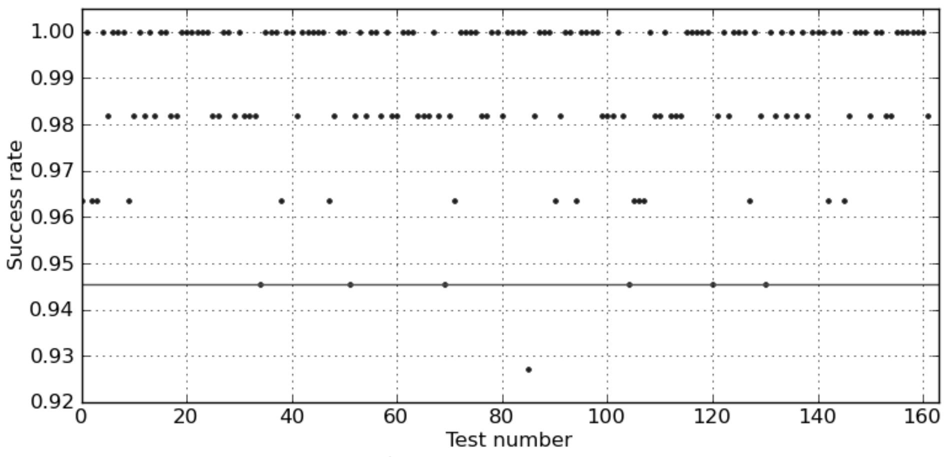

Fig. 9. Testing results for non-random keys with long bit series

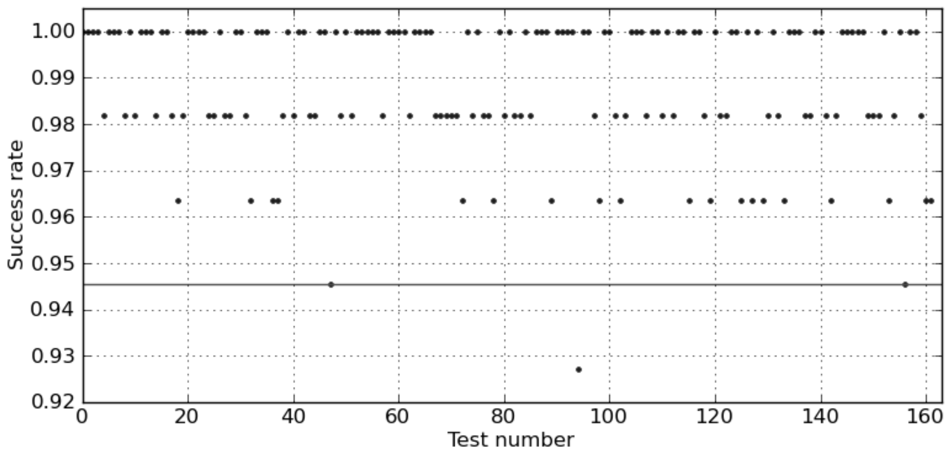

Fig. 10. Testing results for LFSR state leading to symmetric words in BR

Table 6. Keystream randomness test summary

Type of initialization	Tests passed
Random keys	162/163 (99%)
Keys with long bit series	162/163 (99%)
State that generates symmetric words in BR	162/163 (99%)

According to NIST STS in order for the keystream to be random 52 out of 55 samples should pass all statistical tests (in case of level of significance to be 0.01). Keystream data set generated from random keys failed the Longest Run test (51/55 samples passed). The other two keystream data sets failed the Non-Overlapping Template Matching test for 1 of the 147 provided templates (51/55 samples passed).

Even though the maximum keystream length used for encryption in LTE network on single key is 65504 bits, the algorithm itself can produce any amount of data. In order to make the testing more reliable two more data sets were produced to satisfy Random Excursions and Random Excursions Variant test requirements. Both keystreams generated from random and non-random keys successfully passed the tests. Such results still indicate that the provided keystreams are indistinguishable from truly random sequence and do not depend on the initial key. Initializing the cipher with the state that injects symmetric words into Bit Reorganization layer did not influence the keystream randomness.

5. Conclusions

Detailed mathematical and statistical analysis of ZUC cipher revealed the negligible defect in its linear transformation. Even though the structure of ZUC effectively annihilates any consequences of the found property, such linear transformations need improvement before been used for development of future cryptographic algorithms in order to prevent possible weaknesses and losses of entropy. As for ZUC, no evidence of weakness in the cipher caused by the linear transformations in FSM could be found. The cipher has some set of weak keys that cause bad randomization of LFSR state during initialization. Particularly all non-random keys with long series of zeroes and ones result to non-random state after initialization. However, such initialization states did not affect the randomness of generated keystream.

REFERENCES

1. A Real-World Attack Breaking A5/1 within Hours (Timo Gendrullis, Martin Novotn´y, Andy Rupp).
2. A5/1 Security Project http://reflextor.com/trac/a51.
3. Eavesdropping on Satellite Telecommunication Systems (draft) by Benedikt Driessen Horst-Goertz Institute for IT Security, Ruhr-University Bochum, Germany, 2012. http://eprint.iacr.org/2012/051.pdf.
4. Prohibiting A5/2 in mobile stations and other clarifications regarding A5 algorithm support, 3GPP TSG-SA WG3 (Security) Meeting #48, Montreal, Canada, 10-13 July 2007.
5. ETSI/SAGE Task Force. Specification of the 3GPP Confidentiality and Integrity Algorithms 128-EEA3 & 128-EIA3. Document 4: Design and Evaluation Report. Technical report, 3GPP, 2011. Version: 1.3.
6. Bing Sun, Xuehai Tang and Chao Li, Preliminary Cryptanalysis Results of ZUC, appear in the First International Workshop on ZUC Algorithm, 12, 2010.
7. Hongjun Wu, Cryptanalysis of the Stream Cipher ZUC in the 3GPP Confidentiality & Integrity Algorithms 128-EEA3 & 128-EIA3, appear at the sump session in ASIACRYPT 2010.
8. ZUC discussion forum (http://zucalg.forumotion.net).
9. Official ZUC cipher page on DACAS (http://zuc.dacas.cn/Default.aspx).
10. ENT — A Pseudorandom Number Sequence Test Program, John Walker, 1998 http://www.fourmilab.ch/random.
11. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, National Institute of Standards and Technology (NIST), Gaithersburg, MD, USA, 2001.

Надійшла – 14.03.2012.