Date of publication DEC-31, 2021, date of current version NOV-10, 2021.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.20.4.2436

Elliptic Curve Pseudorandom Bit Generator with Maximum Period Sequences

ALEXANDR A. KUZNETSOV[1,2], YURII I. GORBENKO[1,2], ANASTASIIA S. KIIAN[2], YULIIA V. ULIANOVSKA[3], TATYANA Y. KUZNETSOVA[1]

[1)] V. N. Karazin Kharkiv National University, Svobody sq., 4, Kharkiv, 61022, Ukraine
[2)] JSC "Institute of Information Technologies", Bakulin St., 12, Kharkiv, 61166, Ukraine
[3)] University of Customs and Finance, Street V. Vernadsky, 2/4, Dnipro, 49000, Ukraine

Corresponding author: Alexandr Kuznetsov (e-mail: kuznetsov@karazin.ua).

ABSTRACT Pseudo-random number generator is an important mechanism for cryptographic information protection. It can be used independently to generate special data or as the most important element of security of other mechanisms for cryptographic information protection. The application of transformations in a group of points of elliptic and hyperelliptic curves is an important direction for the designing of cryptographically stable pseudo-random sequences generators. This approach allows us to build the resistant cryptographic algorithms in which the problem of finding a private key is associated with solving the discrete logarithm problem. This paper proposes a method for generating pseudo-random sequences of the maximum period using transformations on the elliptic curves. The maximum sequence period is provided by the use of recurrent transformations with the sequential formation of the elements of the point group of the elliptic curve. In this case, the problem of finding a private key is reduced to solving a theoretically complex discrete logarithm problem. The article also describes the block diagram of the device for generating pseudo-random sequences and the scheme for generating internal states of the generator.

KEYWORDS elliptic curve; discrete logarithm problem; pseudo-random sequence generator; maximum period of sequences; cryptographic strength.

I. INTRODUCTION

RANDOM and pseudo-random number generators are important and extremely powerful cryptographic primitives [1–3]. They are used independently (as a source of primary entropy, key generators, initialization vectors and other special data) or as part of the other mechanisms of cryptographic protection (encryption, cryptographic hashing, zero-knowledge proof protocols, etc.) [1]. Thus, the analysis of known generation techniques, the study of statistical and other properties of the generated sequences is an important task of modern cryptology.

The analysis and comparative studies have shown that the most effective [1,4], in terms of indistinguishability of generated sequences with the implementation of a random process, are methods of generating pseudo-random numbers based on the use of modular transformations [5–7] or transformations in a group of elliptic curve points [1, 4–9]. The most promising are considered to be pseudo-random sequence generators [10–13], which are constructed using transformations in a group of points of an elliptic curve [13–15].

At the same time, as studies have shown in [16, 17], the main disadvantage of such methods is that they do not allow the formation of pseudo-random sequences of the maximum period. For example, when using the well-known Dual Elliptic Curve Deterministic Random Bit Generator with NIST Special Publication 800-90A [18] (in the updated version of the standard this algorithm was excluded [19]), the actual length of the sequence period is much shorter than expected [16]. As the lengths of the parameters increase, this tendency intensifies [16]. Indeed, the application of the operation of scalar multiplication of points of an elliptic curve and the display of the coordinates of the obtained point for the formation of pseudo-random numbers does not provide the maximum period of the formed sequences. The task of this paper is to develop a method for generating pseudo-random numbers sequences, due to the additional introduction of recurrent transformation in combination with transformations in group of points of an elliptic curve it allows generating pseudo-random sequences of the maximum period.

The work is structured as follows. In Section 2, we present the structure and basic transformations of the well-known Dual

Elliptic Curve Deterministic Random Bit Generator. We show that the sequence of internal states of the generator depends on the basic operations of scalar multiplication in the group of elliptic curve points. In fact, there is a looping of internal states, due to which the periods of the sequences formed by the generator are very small. This is the main drawback that we got rid of when developing a new generator. A description of its structure and basic transformations is given in Section 3. Finally, Section 4 is devoted to experimental research. In particular, we consider a simple example with an elliptical curve and the corresponding Dual Elliptic Curve Deterministic Random Bit Generator. We show that the periods of the sequences formed by the generator are critically small. For the same parameters, we investigate the new generator proposed by us and show that the periods of all formed sequences are maximal. In the final part of the work, we summarize and note the benefits gained.

II. DUAL ELLIPTIC CURVE DETERMINISTIC RANDOM BIT GENERATOR

In 2012, the US National Institute of Standards and Technology (NIST) approved the NIST Special Publication 800-90A [19]. This document defines different approaches to generating pseudo-random bits. In particular, Section 10 provides specifications for pseudo-random number generation algorithms using various cryptographic mechanisms: hashing, hash-based message authentication code (HMAC), symmetric block ciphers, and theoretical-numerical problems. The last class includes the Dual Elliptic Curve Deterministic Random Bit Generator, which is given in Section 10.3.1. And although this generator was excluded from the updated version of the standard [20] (due to certain safety deficiencies), its study and research of ways to improve is undoubtedly relevant.

Dual Elliptic Curve Deterministic Random Bit Generator is based on the application of two scalar multiplications of the points of the elliptic curve and the mapping of the corresponding *x*-coordinates of the results to a non-zero integer value [18]. The block diagram of the generator is shown in Fig. 1.

The first scalar multiplication by a fixed (base) point P is performed to generate an intermediate state s_i, which is cyclically updated at each iteration during the operation of the corresponding generator. Thus, the value of the state s_i depends on the value of the previous state s_{i-1} (on the previous iteration) and on the value of the base point P:

$$s_i = \phi(x(s_{i-1} P)), \quad (1)$$

where $x(A)$ is a x-coordinate of the point A, $\phi(x)$ is a function of mapping field elements to non-zero integers.

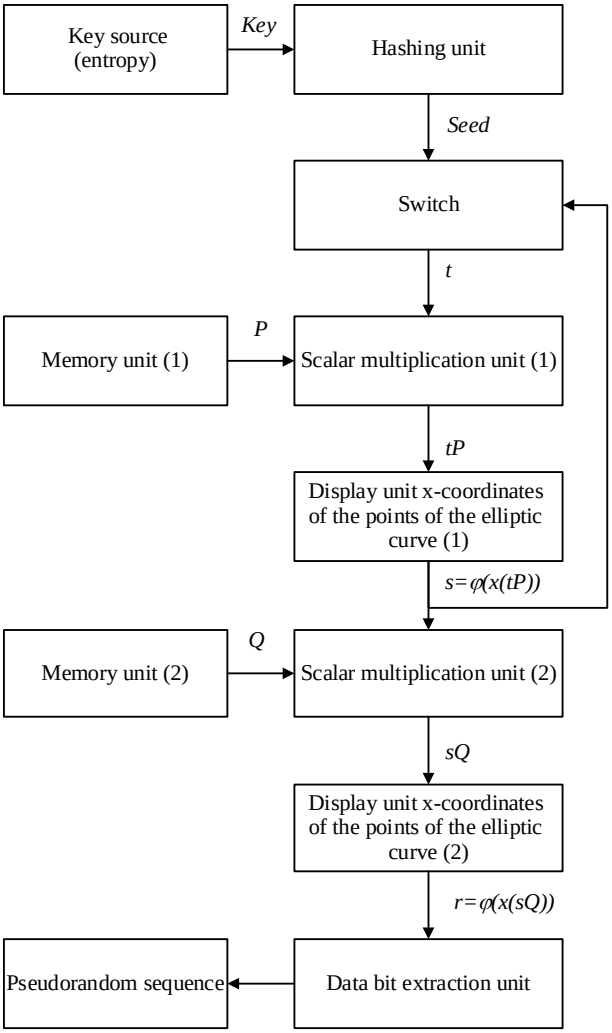

Figure 1. Block diagram Dual Elliptic Curve Deterministic Random Bit Generator.

The initial value of the parameter s_0 is formed using the initialization procedure, which includes entering a private key (Key), which specifies the initial entropy, and hashing the entered key with the formatting of the result to a certain bit length. The value $Seed$ obtained in this way defines the initial value of the parameter: $s_0 = Seed$. The second scalar multiplication by a fixed (base) point Q is performed to generate an intermediate state r_i, which after the appropriate transformation sets the value of the generated pseudo-random bits. The value of the parameter r_i depends on the parameter generated as a result of the first scalar multiplication s_i and on the value of the base point Q:

$$r_i = \phi(x(s_i Q)). \quad (2)$$

The obtained value r_i is the source for the formation of pseudo-random bits, which are generated by reading a block of the least significant (right) bits of the number r_i. The pseudo-

random sequence is generated by concatenation of the read bits of the formed numbers r_i. The values of fixed (base) points are set as constants and do not change during the formation of the pseudo-random sequence.

Thus, the considered method of generation of pseudo-random sequences applies the transformation in the group of points of the elliptic curve to the generation of the intermediate states s_i and r_i. Moreover, the reverse action, i.e., the generation of s_{i-1} by the known s_i, and / or the generation of s_i by the known r_i is associated with the solution to a theoretically complex elliptic curve discrete logarithm problem. The scheme of generation of the intermediate states of the generator is given in Fig. 2.

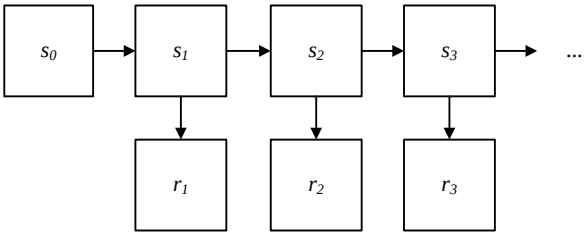

Figure 2. Scheme of generation of the intermediate states of the generator.

According to Fig. 2 the sequence of states $\dots s_{i-1}$, s_i, $\dots s_{i+1} \dots$ is generated from the initial value $s_0 = Seed$, which is formed from the private key data. Each subsequent value s_i depends on the previous value s_{i-1} and is formed by scalar multiplication of the base point of the elliptic curve by formula (1). Individual bits of the pseudo-random sequence are formed by reading the bits of the sequence of numbers $\dots r_{i-1}$, r_i, r_{i+1} ,…, i.e., by reading the data obtained by scalar multiplication of another base point by the corresponding values of the states $\dots s_{i-1}$, s_i, $\dots s_{i+1} \dots$ by formula (2).

Since the private key (Key), which sets the rule of sequence generation, after certain transformations determines the initial value of the parameter s_0, the corresponding stability of the generator is based on reducing the problem of recovering private key data to solve a well-known and extremely complex mathematical elliptic curve discrete logarithm problem. In addition, individual fragments of a pseudo-random sequence are also interconnected by scalar multiplication of an elliptic curve point, i.e., in order to reconstruct any fragment of a pseudo-random sequence by some other known fragment, it is necessary to solve elliptic curve discrete logarithm problem. If for the considered generator with the known fragment of a pseudo-random sequence it is possible to recover another, any unknown fragment, or it is possible to recover value of a secret key (or at least values of elements of sequence $\dots s_{i-1}$, s_i, $\dots s_{i+1} \dots$) it means that it is possible to solve discrete logarithm in a group of elliptic curve points, i.e., an inverted function (1) or (2).

The study of the periodic properties of the Dual Elliptic Curve Deterministic Random Bit Generator was performed in [16]. In particular, this paper shows that the actual length of the periods of the formed sequences is much shorter than the maximum period. The problem is the early looping of the sequence of states. In Fig. 3 the periodicity of the sequence of states is schematically shown.

$$s_0, s_1, \dots, s_{i-1}, s_i, \dots s_{i+1}, \dots, s_{L-1}, s_0, s_1, \dots,$$

and the resulting periodicity of the sequence

$$r_0, r_1, \dots, r_{i-1}, r_i, \dots r_{i+1}, \dots, r_{L-1}, r_0, r_1, \dots,$$

where r_0 is a value that can be obtained by expression (2) when substituting as an argument to the function of scalar multiplication of points of a number $s_0 = Seed$.

The states sequence s_i formed by scalar multiplication of the base point in formula (1) is a periodic sequence (non-periodicity in this case will mean the infinity of the set of states and the corresponding infinity of the set of points of the elliptic curve, which contradicts the finiteness of the group). In other words, in the set of possible scalars, to which the base point P is multiplied in formula (1), there will be such a value $k = s_{l=L-1}$ that $s_{L-1} = s_0$ and all subsequent values of states s_{l+j}, $j = 1, 2, \dots$ will begin to repeat with the corresponding values s_j. The problem is that such looping starts too early. In [16] an example was given when the real period is an order of magnitude less than expected and, according to our estimates, this tendency also intensifies with increasing key lengths. Free from this disadvantage is the proposed new generator on elliptical curves.

III. NEW PSEUDO-RANDOM SEQUENCE GENERATOR ON ELLIPTIC CURVES

The generation of sequences of the maximum period is solved by the additional introduction of recurrent transformations. The block diagram of the new generator is shown in Fig. 4. The basis of our proposal is the above Dual Elliptic Curve Deterministic Random Bit Generator (from the previous version of the standard NIST SP 800-90). The newly introduced elements are highlighted in Fig. 4.

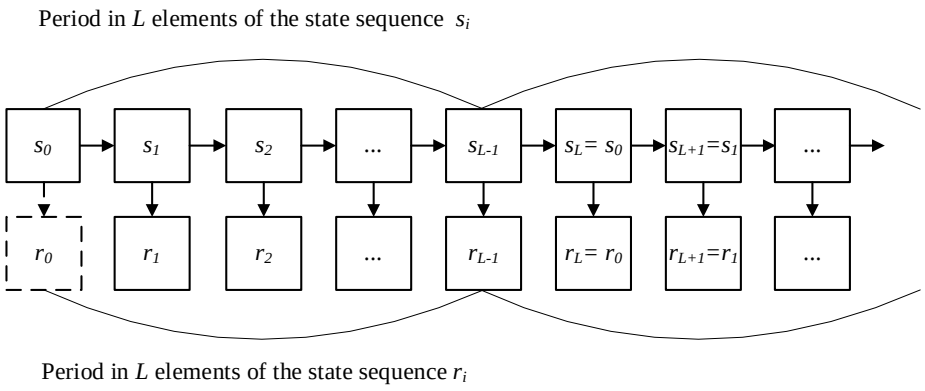

Figure 3. Scheme of formation of periodic sequences of the states of the generator

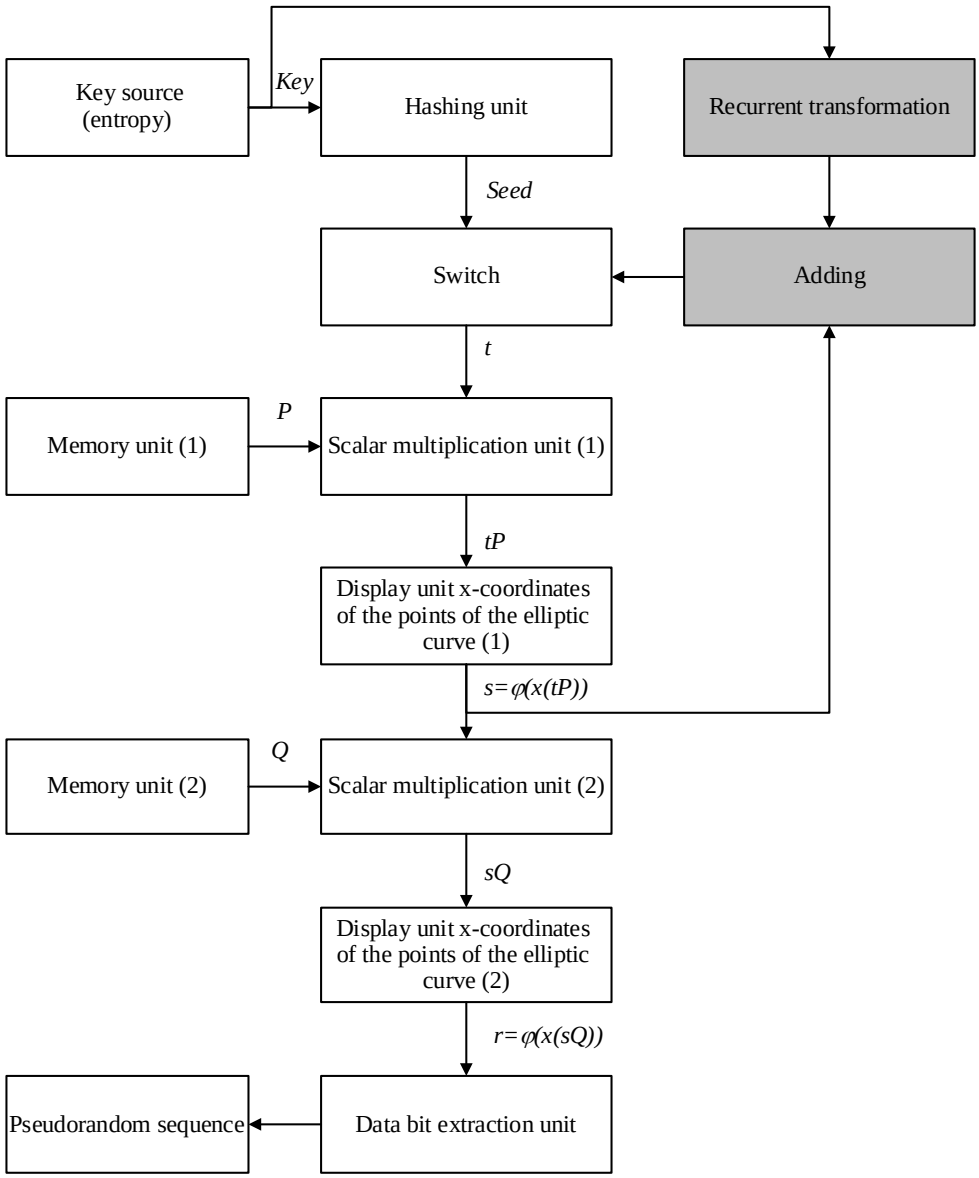

Figure 4. Block diagram of the new generator.

The first scalar multiplication to a fixed (base) point, P as in the Dual Elliptic Curve Deterministic Random Bit Generator, is performed to generate an intermediate state s_i,

which is cyclically updated at each iteration during the operation of the corresponding generator. But the fundamental difference is the process of generating this intermediate state.

To ensure the maximum period of the sequences

$$\dots s_{i-1},\ s_i, \dots s_{i+1} \dots$$

in our method it is proposed to use a recurrent transformation, which is initiated by the entered secret key (Key).

Thus, each subsequent value of the state depends not only on the value of the previous state s_{i-1} (on the previous iteration) and on the value of the base point P, but also on the result of the recurrent transformation (denote it by $LRR(y)$), i.e.,

$$s_i = \phi(x((s_{i-1} + LRR(y))P)),$$

where $x(A)$ is the x-coordinate of the point A, $\phi(x)$ is a function of mapping field elements to non-zero integers.

The recurrent transformation can be constructed in different known ways, in particular, through the simplest variant using a chain of linear recurrent registers (LRR) with feedback (see Fig. 5) [20, 21], the taps of which are given by the polynomial coefficients

$$g(x) = g_0 + g_1 x + g_2 x^2 + \dots + b_m x^m .$$

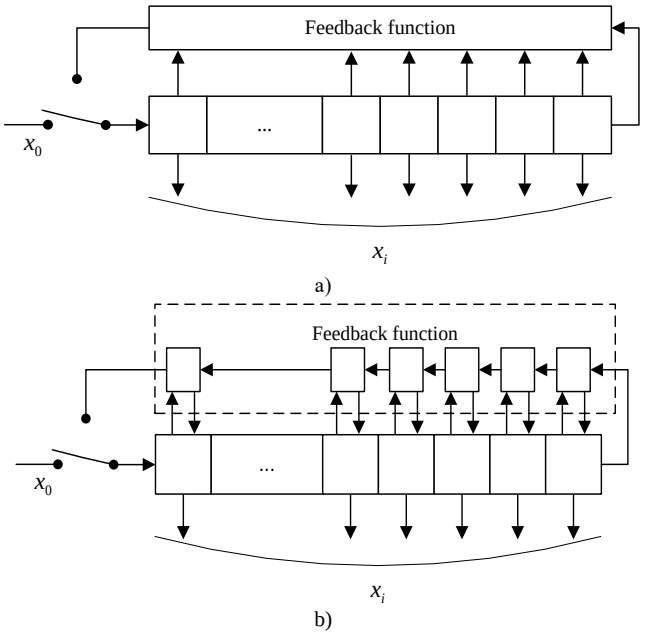

Figure 5. Structural linear registers in Fibonacci configuration (a) and in Galois configuration (b)

If the polynomial $g(x)$ is primitive over a finite field $GF(2^m)$, then the sequence formed by the LRR with the corresponding feedback logic has a maximum period equal to $2^m - 1$. The value of the private key (Key), which initiates the work $LRR(y)$, is written in the LRR as the initial value of the register.

Fig. 5 shows the general scheme of construction of such devices in the Fibonacci configuration (Fig. 5.a) and in the Galois configuration (Fig. 5.b) [20, 21].

The initial value of the parameter s_0, as in Dual Elliptic Curve Deterministic Random Bit Generator, is formed using the initialization procedure, which includes entering a private key (Key), which sets the initial entropy (uncertainty), and hashing the entered key with formatting the result to a certain bit length. The value obtained in this way sows (initiates) the initial value of the parameter: $s_0 = Seed$.

The second scalar multiplication by a fixed (base) point Q is performed to form an intermediate state r_i, which after the appropriate transformation sets the value of the generated pseudo-random bits.

Since each subsequent state value s_i depends on the result of the recurrent transformation $LRR(y)$, which provides the maximum period of the generated sequences, then the value of the parameter r_i depends on the parameter s_i and the value of the base point Q:

$$r_i = \phi(x(s_i Q))$$

will depend on the result of the recurrent transformation $LRR(y)$, i.e., the generated sequence of states $\dots r_{i-1}$, r_i, r_{i+1},… will have a maximum period.

The obtained value r_i is a source for generation of pseudo-random bits, which are formed by reading the block from the least significant (right) bits of the number r_i. The pseudo-random sequence is generated by concatenation of the bits of the formed numbers r_i. The values of fixed (base) points are set as constants and they do not change during the formation of the pseudo-random sequence.

Thus, the periodic properties of the states of the proposed generator are determined by the periodic properties of the additionally introduced recurrent transformation $LRR(y)$.

Let us denote the original sequence of transformation $LRR(y)$ by $\dots y_{i-1}$, y_i, $\dots y_{i+1} \dots$ and schematically show the influence of the periodicity of this sequence on the periodicity of the sequences $\dots s_{i-1}$, s_i, $\dots s_{i+1} \dots$ and $\dots r_{i-1}$, r_i, r_{i+1},…. We use the scheme shown in Fig. 6.

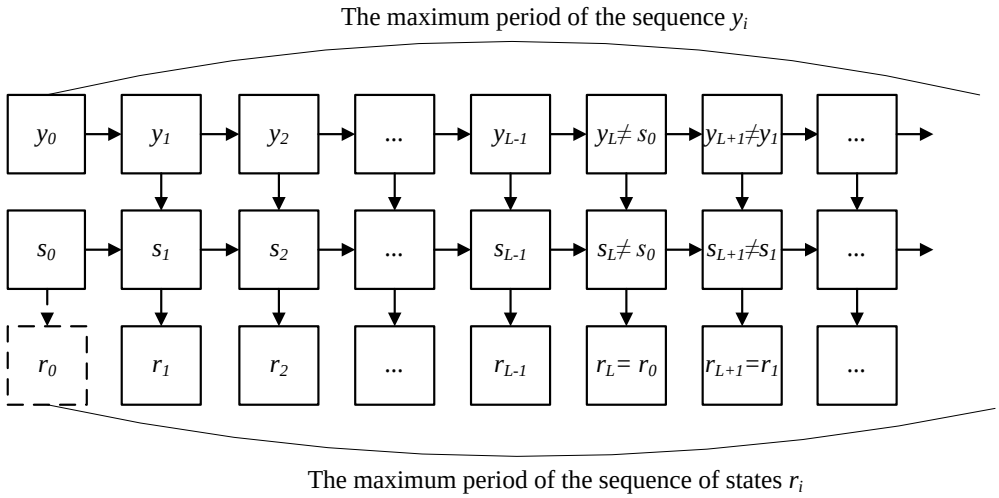

Figure 6. Scheme of generation of the sequences of the generator states with the maximum period

Because the periodic properties of the sequences $\dots s_{i-1}$, s_i, $\dots s_{i+1} \dots$ and $\dots r_{i-1}$, r_i, $r_{i+1}, \dots$ directly depend on the properties of the sequence $\dots y_{i-1}$, y_i, $\dots y_{i+1} \dots$, the use of recurrent transformation $LRR(y)$ with the maximum period of the original sequences provides the maximum period of the original sequence.

Generation of pseudo-random sequences using linear recurrent registers (denoted by LRR) can be represented as follows.

Private key: *Key*;

Constants: *P*, *Q* are points of the EC with order n; Initial state:

$$x_0 = Key, y_0 = Key;$$

Cycle function:

$$\phi(f(x + LRR(y))) = \phi((x + LRR(y))P),$$

$$LRR(y = \{u_1, u_2, \dots, u_m\}): u_i = -\sum_j a_j u_{i-j} + u_i,$$

where $\{u_1, u_2, \dots, u_m\}$ is a LRR state, $\{a_1, a_2, \dots, a_m\}$ are the coefficients that define the LRR feedback function; $\phi(P'_i)$ is a conversion of point coordinates $P'_i \in EC_n$ (for example, reading the value of one of the point P'_i coordinates). Generated pseudo-random sequence: $(b_0, b_1, \dots, b_i, \dots)$,

where b_i is the least significant bit (parity bit) of the number z_i,

$$z_i = \phi(f'(\phi((x_{i-1} + LRR(y_{i-1}))P))) = \phi(\phi((x_{i-1} + LRR(y_{i-1}))P)Q), \quad y_i = LRR(y_{i-1}).$$

Thus, due to the additional introduction of recurrent transformations, which are implemented, for example, by LRR, it is possible to generate pseudo-random sequences of the maximum period.

IV. EXPERIMENTAL STUDY OF PERIODIC PROPERTIES

To confirm the conclusions about the periodic properties of pseudo-random sequence generators, we conduct the following experimental studies.

A. DUAL ELLIPTIC CURVE DETERMINISTIC RANDOM BIT GENERATOR

According to the specification of the generator, which is given in the standard NIST SP 800-90 [19], it is recommended to use an elliptic curve defined over a finite prime field $GF(p)$, i.e., a set of pairs of numbers (x, y), $x, y \in GF(p)$ that satisfy the identities:

$$y^2 \equiv x^3 + ax + b \,(mod\ p),$$

where $a, b \in GF(p)$ and the condition is met

$$4a^3 + 27b^2 \neq 0 \,(mod\ p).$$

According to the NIST SP 800-90 [19] specification, it is recommended to use $a = -3$ both parameters b, p and values of coordinates of points P and Q from the corresponding field $GF(p)$, the bit size of the base of which is 256, 384 or 521 bits. To simplify, consider the case of using an elliptic curve over $GF(7)$, which is given by the equation

$$y^2 \equiv x^3 - 3x + 4 \,(mod\ 7),$$

and the condition is fulfilled

$$4a^3 + 27b^2 = 2 (mod\ 7) \neq 0 \,(mod\ 7).$$

Substituting into the equation of the curve all possible pairs of numbers $(x, y), x, y \in GF(7)$ we choose those that satisfy identities. We obtain a set of solutions to the equation – a set of nonzero points of the curve (Table 1). Thus, the order of the elliptic curve E (the number of all points of the curve, together with the zero point O) is equal to m = 10. It should be noted that the points listed in Table 1 have the following properties: point P_1 (0,2) is a negation of point P_2 (0,5), and vice versa. Similarly, points P_3 (1,3) and P_4 (1,4), P_5 (3,1) and P_6 (3,6), P_8 (5,3) and P_9 (5,4) are negations of each other. Thus, we have:

$$P_1(0,2) = -P_2(0,-5),$$
$$P_3(1,3) = -P_4(1,-4),$$
$$P_5(3,1) = -P_6(3,-6),$$
$$P_8(5,3) = -P_9(5,-4).$$

Point P_7 (4,0) is a self-negation, i.e., P_7 (4,0) = – P_7 (4,0). In accordance with the above provisions we have:

$$\phi(x(P_1(0,2))) = \phi(x(P_2(0,5)));$$
$$\phi(x(P_3(1,3))) = \phi(x(P_4(1,4)));$$
$$\phi(x(P_5(3,1))) = \phi(x(P_6(3,6)));$$
$$\phi(x(P_8(5,3))) = \phi(x(P_9(5,4))).$$

We will use formulas for implementation of operations of addition and doubling of points, we will construct the corresponding Cayley table for all elements of group H_{EC} (see Table 2).

Find all cyclic subgroups generated by the elements of the group and calculate the order of these elements (each point of the elliptic curve) and the corresponding cycles. To do this, take in turn each of the non-zero elements of the group, and begin to perform a group operation on this element with itself, that is, we begin to perform the addition of each point P_i with itself any number of times.

The set of formed elements

$$P_i,\ 2P_i, 3P_i, \ldots, k_iP_i = O$$

is a subgroup (loop) in the group of points H_{EC} generated by the element P_i, the number k_i is the order of the element P_i, and the corresponding subgroup. The results are summarized in Table 3.

In Table 3 it is seen that the group of elliptic curve points H_{EC} of order $m = 10$ contains three cycles of order 10, which are generated by points P_1, P_5 and P_6, accordingly. These cyclic subgroups are equal to the group H_{EC}.

But the group H_{EC} also contains four cycles of order 5, which are generated by points P_3, P_4, P_8 and P_9 accordingly, and two cycles of order 2, which are generated by points P_2 and P_7, respectively. Obviously, the condition of dividing the whole order of the group H_{EC} into the orders of its cyclic subgroups is also fulfilled.

Let us analyze the operation of a pseudo-random sequence generator that uses transformations in the considered group H_{EC}. Assume that points of maximum order are used as basis points P and Q, for example, points $P = P_5$ and $Q = P_6$. Construct a sequence of internal states (1) and (2) and estimate the periodicity of these sequences. For simplicity, we assume that the function $\phi(x)$ of mapping field elements x to nonzero integers is given as. $\phi(x) = x + 1$.

This assumption does not impose certain restrictions on the number of possible different display results, because by definition we have a functional relationship of the argument (field elements) and the value of the function $\phi(x)$ (some integer), i.e., the map is bijective and can be represented as a regular permutation of field elements.

Adding a unit eliminates the formation of a zero value, the occurrence of which translates the operation of the generator into a degenerate state (a deterministic sequence of only zero values is formed).

The obtained results of the generator (values of internal states) for all possible initial values $s_0 = Seed$ are given in Table 4.

The values of the states are given before the first iteration, because the remaining values are a cycle of the values given in the table. The last column shows the period of the generated state sequences, i.e., the smallest number of sequence elements through which the repetition begins.

As it can be seen in Table 4 data periodic properties of the pseudo-random sequence generator are unsatisfactory. Indeed, the resulting sequences have very small values of periods, in most sequences the period is equal to $L = 2$, one of the sequences has a period $L = 1$, i.e., at the output of the generator the same value is formed.

Even for such a simple example, the shortcomings of the considered generator are obviously revealed. It is unsatisfactory periodic properties.

Thus, in the course of research the necessity of improving the methods of pseudo-random sequences generation for keys formation at maintenance of safety of telecommunication systems and technologies is proved.

Table 1. The set of the elliptic curve points

i	1	2	3	4	5	6	7	8	9
(x_i, y_i)	$P_1(0,2)$	$P_2(0,5)$	$P_3(1,3)$	$P_4(1,4)$	$P_5(3,1)$	$P_6(3,6)$	$P_7(4,0)$	$P_8(5,3)$	$P_9(5,4)$

Table 2. Cayley table for the operation of adding elements in a group of the elliptic curve points

+	**O**	**P_1**	**P_2**	**P_3**	**P_4**	**P_5**	**P_6**	**P_7**	**P_8**	**P_9**
O	O	P_1	P_2	P_3	P_4	P_5	P_6	P_7	P_8	P_9
P_1	P_1	P_4	O	P_2	P_6	P_3	P_8	P_9	P_7	P_5
P_2	P_2	O	P_3	P_5	P_1	P_9	P_4	P_8	P_6	P_7
P_3	P_3	P_2	P_5	P_9	O	P_7	P_1	P_6	P_4	P_8
P_4	P_4	P_6	P_1	O	P_8	P_2	P_7	P_5	P_9	P_3
P_5	P_5	P_3	P_9	P_7	P_2	P_8	O	P_4	P_1	P_6
P_6	P_6	P_8	P_4	P_1	P_7	O	P_9	P_3	P_5	P_2
P_7	P_7	P_9	P_8	P_6	P_5	P_4	P_3	O	P_2	P_1
P_8	P_8	P_7	P_6	P_4	P_9	P_1	P_5	P_2	P_3	O
P_9	P_9	P_5	P_7	P_8	P_3	P_6	P_2	P_1	O	P_4

Table 3. Cyclic subgroups of a group of elliptic curve points and the corresponding orders of the curve points

k	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**Orders of points**
kP_1	P_1	P_4	P_6	P_8	P_7	P_9	P_5	P_3	P_2	O	10
kP_2	P_2	O									2
kP_3	P_3	P_9	P_8	P_4	O						5
kP_4	P_4	P_8	P_9	P_3	O						5
kP_5	P_5	P_8	P_1	P_3	P_7	P_4	P_2	P_9	P_6	O	10
kP_6	P_6	P_9	P_2	P_4	P_7	P_3	P_1	P_8	P_5	O	10
kP_7	P_7	O									2
kP_8	P_8	P_3	P_4	P_9	O						5
kP_9	P_9	P_4	P_3	P_8	O						5

Table 4. Internal states of the generator on the elliptic curves

$s_0 \setminus i$	i	**1**	**2**	**3**	**4**	**5**		**L**
$s_0 = 1$	s_i	4	2	6	2			2
	r_i	2	6	2	6			2
$s_0 = 2$	s_i	6	2	6				2
	r_i	2	6	2				2
$s_0 = 3$	s_i	1	4	2	6	2		2
	r_i	4	2	6	2	6		2
$s_0 = 4$	s_i	2	6	2				2
	r_i	6	2	6				2
$s_0 = 5$	s_i	5	5					1
	r_i	5	5					1
$s_0 = 6$	s_i	2	6	2				2
	r_i	6	2	6				2
$s_0 = 7$	s_i	1	4	2	6	2		2
	r_i	4	2	6	2	6		2
$s_0 = 8$	s_i	6	2	6				2
	r_i	2	6	2				2
$s_0 = 9$	s_i	4	2	6	2			2
	r_i	2	6	2	6			2

B. NEW PSEUDO-RANDOM SEQUENCE GENERATOR ON ELLIPTIC CURVES

When setting up the experiment, we use the same initial data as in Section 4.1. The main element of the new generator, which distinguishes it from the prototype, is the introduced recurrent transformation (see Fig. 4). It is used to generate a sequence of internal states y_i. Also in the scheme in Fig. 4, the operation of addition is introduced.

The sequence of internal states y_i has a maximum period; it is generated by a recurrence rule, for example, using a linear recurrence register with feedback. This should provide the maximum period of the initial sequences, i.e., the maximum period of states r_i. This position will be checked during experimental research.

We set the rule for the formation of a recurrent transformation in the form of a linear register in Galois configuration (see Fig. 5.b). The feedback function, which defines the taps of the shift register, is given by the coefficients of the primitive polynomial $g(x) = x^4 + x + 1$. Such a polynomial generates a sequence of the maximum period, i.e., the states of the register give a cycle of length 15:

{1, 2, 4. 8, 3, 6, 12, 11, 5, 10, 7, 14, 15, 13, 9}.

The initiation vector y_0 determines the initial value entered in the shift register.

According to the block diagram of the new generator (Fig. 4), the other transformations are not changed, i.e., when conducting experimental calculations we use the same basic points of maximum order, i.e.,

$$P = P_5 \text{ and } Q = P_6.$$

The obtained results of the generator (values of internal states) for all possible initial values $s_0 = Seed$ are given in Table 5 – 13.

Table 5. Initial values: $s_0 = 1$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	***1***	***1***	***2***	***P(5,3)***	***P(3,6)***	***4***
1	*6*	*2*	*8*	*P(5,4)*	*P(1,3)*	*2*
2	*6*	*4*	*10*	*P(0,0)*	*P(1,3)*	*2*
3	*1*	*8*	*9*	*P(3,6)*	*P(3,6)*	*4*
4	*4*	*3*	*7*	*P(0,5)*	*P(1,4)*	*2*
5	*1*	*6*	*7*	*P(0,5)*	*P(3,6)*	*4*
6	*1*	*12*	*13*	*P(0,2)*	*P(3,6)*	*4*
7	*1*	*11*	*12*	*P(5,3)*	*P(3,6)*	*4*
8	*6*	*5*	*11*	*P(3,1)*	*P(1,3)*	*2*
9	*4*	*10*	*14*	*P(1,3)*	*P(1,4)*	*2*
10	*2*	*7*	*9*	*P(3,6)*	*P(5,4)*	*6*
11	*4*	*14*	*18*	*P(5,4)*	*P(1,4)*	*2*
12	*6*	*15*	*21*	*P(3,1)*	*P(1,3)*	*2*
13	*4*	*13*	*17*	*P(0,5)*	*P(1,4)*	*2*
14	*1*	*9*	*10*	*P(0,0)*	*P(3,6)*	*4*
15	***1***	***1***	***2***	***P(5,3)***	***P(3,6)***	***4***
The sequence period is equal to *L = 15*						

Table 6. Initial values: $s_0 = 2$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	*2*	*2*	*4*	*P(1,3)*	*P(5,4)*	*6*
1	*2*	*4*	*6*	*P(1,4)*	*P(5,4)*	*6*
2	*2*	*8*	*10*	*P(0,0)*	*P(5,4)*	*6*
3	***1***	***3***	***4***	***P(1,3)***	***P(3,6)***	***4***
4	*2*	*6*	*8*	*P(5,4)*	*P(5,4)*	*6*
5	*6*	*12*	*18*	*P(5,4)*	*P(1,3)*	*2*
6	*6*	*11*	*17*	*P(0,5)*	*P(1,3)*	*2*
7	*1*	*5*	*6*	*P(1,4)*	*P(3,6)*	*4*
8	*2*	*10*	*12*	*P(5,3)*	*P(5,4)*	*6*
9	*6*	*7*	*13*	*P(0,2)*	*P(1,3)*	*2*
10	*1*	*14*	*15*	*P(4,0)*	*P(3,6)*	*4*
11	*5*	*15*	*20*	*P(0,0)*	*P(4,0)*	*5*
12	*1*	*13*	*14*	*P(1,3)*	*P(3,6)*	*4*
13	*2*	*9*	*11*	*P(3,1)*	*P(5,4)*	*6*
14	*4*	*1*	*5*	*P(4,0)*	*P(1,4)*	*2*
15	*5*	*2*	*7*	*P(0,5)*	*P(4,0)*	*5*
16	*1*	*4*	*5*	*P(4,0)*	*P(3,6)*	*4*
17	*5*	*8*	*13*	*P(0,2)*	*P(4,0)*	*5*
18	***1***	***3***	***4***	***P(1,3)***	***P(3,6)***	***4***
The sequence period is equal to *L = 15*						

Table 7. Initial values: $s_0 = 3$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	3	3	6	P(1,4)	P(0,5)	1
1	**2**	**6**	**8**	**P(5,4)**	**P(5,4)**	**6**
2	6	12	18	P(5,4)	P(1,3)	2
3	6	11	17	P(0,5)	P(1,3)	2
4	1	5	6	P(1,4)	P(3,6)	4
5	2	10	12	P(5,3)	P(5,4)	6
6	6	7	13	P(0,2)	P(1,3)	2
7	1	14	15	P(4,0)	P(3,6)	4
8	5	15	20	P(0,0)	P(4,0)	5
9	1	13	14	P(1,3)	P(3,6)	4
10	2	9	11	P(3,1)	P(5,4)	6
11	4	1	5	P(4,0)	P(1,4)	2
12	5	2	7	P(0,5)	P(4,0)	5
13	1	4	5	P(4,0)	P(3,6)	4
14	5	8	13	P(0,2)	P(4,0)	5
15	1	3	4	P(1,3)	P(3,6)	4
16	**2**	**6**	**8**	**P(5,4)**	**P(5,4)**	**6**
The sequence period is equal to L = 15						

Table 8. Initial values: $s_0 = 4$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	4	4	8	P(5,4)	P(1,4)	2
1	6	8	14	P(1,3)	P(1,3)	2
2	2	3	5	P(4,0)	P(5,4)	6
3	5	6	11	P(3,1)	P(4,0)	5
4	4	12	16	P(1,4)	P(1,4)	2
5	2	11	13	P(0,2)	P(5,4)	6
6	**1**	**5**	**6**	**P(1,4)**	**P(3,6)**	**4**
7	2	10	12	P(5,3)	P(5,4)	6
8	6	7	13	P(0,2)	P(1,3)	2
9	1	14	15	P(4,0)	P(3,6)	4
10	5	15	20	P(0,0)	P(4,0)	5
11	1	13	14	P(1,3)	P(3,6)	4
12	2	9	11	P(3,1)	P(5,4)	6
13	4	1	5	P(4,0)	P(1,4)	2
14	5	2	7	P(0,5)	P(4,0)	5
15	1	4	5	P(4,0)	P(3,6)	4
16	5	8	13	P(0,2)	P(4,0)	5
17	1	3	4	P(1,3)	P(3,6)	4
18	2	6	8	P(5,4)	P(5,4)	6
19	6	12	18	P(5,4)	P(1,3)	2
20	6	11	17	P(0,5)	P(1,3)	2
21	**1**	**5**	**6**	**P(1,4)**	**P(3,6)**	**4**
The sequence period is equal to L = 15						

Table 9. Initial values: $s_0 = 5$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	5	5	10	P(0,0)	P(4,0)	5
1	1	10	11	P(3,1)	P(3,6)	4
2	4	7	11	P(3,1)	P(1,4)	2
3	**4**	**14**	**18**	**P(5,4)**	**P(1,4)**	**2**
4	6	15	21	P(3,1)	P(1,3)	2
5	4	13	17	P(0,5)	P(1,4)	2
6	1	9	10	P(0,0)	P(3,6)	4
7	1	1	2	P(5,3)	P(3,6)	4
8	6	2	8	P(5,4)	P(1,3)	2
9	6	4	10	P(0,0)	P(1,3)	2
10	1	8	9	P(3,6)	P(3,6)	4
11	4	3	7	P(0,5)	P(1,4)	2
12	1	6	7	P(0,5)	P(3,6)	4
13	1	12	13	P(0,2)	P(3,6)	4
14	1	11	12	P(5,3)	P(3,6)	4
15	6	5	11	P(3,1)	P(1,3)	2
16	4	10	14	P(1,3)	P(1,4)	2
17	2	7	9	P(3,6)	P(5,4)	6
18	**4**	**14**	**18**	**P(5,4)**	**P(1,4)**	**2**
The sequence period is equal to L = 15						

Table 10 Initial values: $s_0 = 6$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	6	6	12	P(5,3)	P(1,3)	2
1	**6**	**12**	**18**	**P(5,4)**	**P(1,3)**	**2**
2	6	11	17	P(0,5)	P(1,3)	2
3	1	5	6	P(1,4)	P(3,6)	4
4	2	10	12	P(5,3)	P(5,4)	6
5	6	7	13	P(0,2)	P(1,3)	2
6	1	14	15	P(4,0)	P(3,6)	4
7	5	15	20	P(0,0)	P(4,0)	5
8	1	13	14	P(1,3)	P(3,6)	4
9	2	9	11	P(3,1)	P(5,4)	6
10	4	1	5	P(4,0)	P(1,4)	2
11	5	2	7	P(0,5)	P(4,0)	5
12	1	4	5	P(4,0)	P(3,6)	4
13	5	8	13	P(0,2)	P(4,0)	5
14	1	3	4	P(1,3)	P(3,6)	4
15	2	6	8	P(5,4)	P(5,4)	6
16	**6**	**12**	**18**	**P(5,4)**	**P(1,3)**	**2**
The sequence period is equal to L = 15						

Table 11. Initial values: $s_0 = 7$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	7	7	14	P(1,3)	P(0,2)	1
1	2	14	16	P(1,4)	P(5,4)	6
2	2	15	17	P(0,5)	P(5,4)	6
3	**1**	**13**	**14**	**P(1,3)**	**P(3,6)**	**4**
4	2	9	11	P(3,1)	P(5,4)	6
5	4	1	5	P(4,0)	P(1,4)	2
6	5	2	7	P(0,5)	P(4,0)	5
7	1	4	5	P(4,0)	P(3,6)	4
8	5	8	13	P(0,2)	P(4,0)	5
9	1	3	4	P(1,3)	P(3,6)	4
10	2	6	8	P(5,4)	P(5,4)	6
11	6	12	18	P(5,4)	P(1,3)	2
12	6	11	17	P(0,5)	P(1,3)	2
13	1	5	6	P(1,4)	P(3,6)	4
14	2	10	12	P(5,3)	P(5,4)	6
15	6	7	13	P(0,2)	P(1,3)	2
16	1	14	15	P(4,0)	P(3,6)	4
17	5	15	20	P(0,0)	P(4,0)	5
18	**1**	**13**	**14**	**P(1,3)**	**P(3,6)**	**4**
The sequence period is equal to L = 15						

Table 12. Initial values: $s_0 = 8$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	8	8	16	P(1,4)	P(5,3)	6
1	2	3	5	P(4,0)	P(5,4)	6
2	5	6	11	P(3,1)	P(4,0)	5
3	4	12	16	P(1,4)	P(1,4)	2
4	2	11	13	P(0,2)	P(5,4)	6
5	**1**	**5**	**6**	**P(1,4)**	**P(3,6)**	**4**
6	2	10	12	P(5,3)	P(5,4)	6
7	6	7	13	P(0,2)	P(1,3)	2
8	1	14	15	P(4,0)	P(3,6)	4
9	5	15	20	P(0,0)	P(4,0)	5
10	1	13	14	P(1,3)	P(3,6)	4
11	2	9	11	P(3,1)	P(5,4)	6
12	4	1	5	P(4,0)	P(1,4)	2
13	5	2	7	P(0,5)	P(4,0)	5
14	1	4	5	P(4,0)	P(3,6)	4
15	5	8	13	P(0,2)	P(4,0)	5
16	1	3	4	P(1,3)	P(3,6)	4
17	2	6	8	P(5,4)	P(5,4)	6
18	6	12	18	P(5,4)	P(1,3)	2
19	6	11	17	P(0,5)	P(1,3)	2
20	**1**	**5**	**6**	**P(1,4)**	**P(3,6)**	**4**
The sequence period is equal to L = 15						

Table 13. Initial values: $s_0 = 9$, $P = P_5$, $Q = P_6$

i	The value of internal states					
	s_i	y_i	s_i+y_i	$(s_i+y_i)P$	s_iQ	r_i
0	9	9	18	P(5,4)	P(3,1)	4
1	6	1	7	P(0,5)	P(1,3)	2
2	1	2	3	P(0,2)	P(3,6)	4
3	**1**	**4**	**5**	**P(4,0)**	**P(3,6)**	**4**
4	5	8	13	P(0,2)	P(4,0)	5
5	1	3	4	P(1,3)	P(3,6)	4
6	2	6	8	P(5,4)	P(5,4)	6
7	6	12	18	P(5,4)	P(1,3)	2
8	6	11	17	P(0,5)	P(1,3)	2
9	1	5	6	P(1,4)	P(3,6)	4
10	2	10	12	P(5,3)	P(5,4)	6
11	6	7	13	P(0,2)	P(1,3)	2
12	1	14	15	P(4,0)	P(3,6)	4
13	5	15	20	P(0,0)	P(4,0)	5
14	1	13	14	P(1,3)	P(3,6)	4
15	2	9	11	P(3,1)	P(5,4)	6
16	4	1	5	P(4,0)	P(1,4)	2
17	5	2	7	P(0,5)	P(4,0)	5
18	**1**	**4**	**5**	**P(4,0)**	**P(3,6)**	**4**
The sequence period is equal to L = 15						

The values of the states in Tables 5-13 are given for the first repetition, because the remaining values are a cycle of values given in the table. The last line shows the period of formed states sequences, i.e., the smallest number of elements of sequences, through which the duplicates begin.

As can be seen from the above data, the periodic properties of the new generator are significantly improved compared to the prototype (see Table 4). Indeed, all the resulting sequences have the maximum order $L = 15$. This order is determined by the periodic properties of the additionally introduced sequences of internal states y_i. In comparison with the prototype, an increase of almost an order of magnitude in the length of the generated source sequences is achieved.

Thus, in the course of experimental research, the general theoretical conclusion is confirmed that the pseudo-random sequences formed by the new generator have improved periodic properties. Even for the given simple example there is a significant increase in the length of the period and with increasing length of the seed vector this improvement increases.

V. CONCLUSIONS

Using cryptographic transformations in a group of elliptic curves points, it is possible to design efficient generators of pseudo-random sequences. However, it is known that the deterministic random number generator of the double elliptic curve (which is described in the standard NIST SP 800-90) has a significant disadvantage. The cyclic function of the generator does not provide the maximum period of the generated sequence of internal states and the corresponding points of the elliptic curves. This is due to the existence of an early cycle. Because of this, the real period of pseudo-random sequences is much smaller than expected.

This article describes a new method that allows you to generate sequences of generator states with the maximum period. It uses recurrent transformations (implemented, for example, using linear recurrent registers with feedback). The generated sequences of the elliptic curve points also have a maximum period. Thus, the proposed method eliminates the

disadvantages of the deterministic random bit generator of the double elliptic curve with respect to the periodic properties of the generated pseudo-random sequences.

References

[1] A. J. Menezes, P. C. van Oorschot, S. A. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 2018. https://doi.org/10.1201/9780429466335.

[2] I. V. Chugunkov, M. A. Ivanov, E. A. Gridneva, N. Y. Shestakova, "Classification of pseudo-random number generators applied to information security," *Proceedings of the 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus)*, 2017, pp. 370–373. https://doi.org/10.1109/EIConRus.2017.7910569.

[3] J. Chi, L. Dong, Y. Zeng, "Reconfigurable pseudo-random number generator based on cellular automata," *Proceedings of the 2019 International Conference on Networking and Network Applications (NaNA)*, 2019, pp. 268–273. https://doi.org/10.1109/NaNA.2019.00054.

[4] H. Delfs, H. Knebl, *Introduction to Cryptography*, Berlin, Heidelberg: Springer, 2015. https://doi.org/10.1007/978-3-662-47974-2.

[5] L. Blum, M. Blum, M. Shub, "A simple unpredictable pseudo-random number generator," *SIAM J Comput*, vol. 15, pp. 364–383, 1986. https://doi.org/10.1137/0215025.

[6] M. Blum, S. Micali, "How to generate cryptographically strong sequences of pseudo random bits," *Proceedings of the 23rd Annual Symposium on Foundations of Computer Science (SFCS'1982)*, 1982, pp. 112–117. https://doi.org/10.1109/SFCS.1982.72.

[7] M. Blum, S. Micali, "How to generate cryptographically strong sequences of pseudo-random bits," *SIAM J Comput*, vol. 13, pp. 850–864, 1984. https://doi.org/10.1137/0213053.

[8] S. Rubinstein-Salzedo, *Cryptography*, Cham: Springer International Publishing, 2018. https://doi.org/10.1007/978-3-319-94818-8.

[9] A. Shamir, "On the generation of cryptographically strong pseudo-random sequences," In: Even S, Kariv O, editors. Automata, Languages and Programming, Berlin, Heidelberg: Springer; 1981, p. 544–550. https://doi.org/10.1007/3-540-10843-2_43.

[10] O. Reyad, M. E. Karar, K. Hamed, *Random Bit Generator Mechanism Based on Elliptic Curves and Secure Hash Function*. ArXiv:200209239 [Cs] 2020. https://doi.org/10.1109/AECT47998.2020.9194180.

[11] J. Payingat, D. P. Pattathil, "Pseudorandom bit sequence generator for stream cipher based on elliptic curves," *Mathematical Problems in Engineering*, vol. 2015, e257904, 2015. https://doi.org/10.1155/2015/257904.

[12] M. Benssalah, M. Djeddou, K. Drouiche, "Pseudo-random sequence generator based on random selection of an elliptic curve," *Proceedings of the 2015 International Conference on Computer, Information and Telecommunication Systems (CITS)*, 2015, p. 1–5. https://doi.org/10.1109/CITS.2015.7297719.

[13] L.-P. Lee, K.-W. Wong, "A random number generator based on elliptic curve operations," *Computers & Mathematics with Applications*, vol. 47, pp. 217–226, 2004. https://doi.org/10.1016/S0898-1221(04)90018-1.

[14] R. Steinmetz, J. Dittmann, M. Steinebach, editors, "Communications and Multimedia Security Issues of the New Century," *Proceedings of the IFIP TC6 / TC11 Fifth Joint Working Conference on Communications and Multimedia Security (CMS'01)*, May 21–22, 2001, Darmstadt, Germany, Springer US, 2001. https://doi.org/10.1007/978-0-387-35413-2.

[15] V. Chevardin, "Deterministic random bit generator on elliptic curve transformations," Proceedings *of International Conference on Modern Problem of Radio Engineering, Telecommunications and Computer Science*, 2012, зр. 468–468.

[16] A. Kuznetsov, S. Kavun, V. Panchenko, D. Prokopovych-Tkachenko, F. Kurinniy, V. Shoiko, "Periodic properties of cryptographically strong pseudorandom sequences," *Proceedings of the 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T)*, 2018, pp. 129–134. https://doi.org/10.1109/INFOCOMMST.2018.8632021.

[17] A. Kuznetsov, A. Kiian, O. Smirnov, A. Cherep, M. Kanabekova, I. Chepurko, "Testing of code-based pseudorandom number generators for post-quantum applicationm," *Proceedings of the 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, 2020, pp. 172–177. https://doi.org/10.1109/DESSERT50317.2020.9125045.

[18] E. Barker, J. Kelsey, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, National Institute of Standards and Technology, 2012. https://doi.org/10.6028/NIST.SP.800-90a.

[19] E. Barker, J. Kelsey, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, National Institute of Standards and Technology, 2015. https://doi.org/10.6028/NIST.SP.800-90Ar1.

[20] A. Canteaut, Linear Feedback Shift Register, In: van Tilborg H.C.A., Jajodia S., editors, *Encyclopedia of Cryptography and Security*, Boston, MA: Springer US; 2011, pp. 726–729. https://doi.org/10.1007/978-1-4419-5906-5_357.

[21] R. E. Blahut, *Theory and Practice of Error Control Codes*. Reprint with corr edition, Reading, MA: Addison-Wesley, 1983.

ALEXANDR A. KUZNETSOV, Doctor of Sciences (Engineering), Full Professor, Professor of the Department security information systems and technologies of the V. N. Karazin Kharkiv National University. Areas of scientific interests: cryptography and authentication, steganography, cybersecurity.
Email: kuznetsov@karazin.ua

YURII I. GORBENKO, Candidate of Sciences (Engineering), Academician of the Academy of Applied Radioelectronics Sciences, Researcher of the Department security information systems and technologies of the V. N. Karazin Kharkiv National University. Areas of scientific interests: applied cryptology.
Email: gorbenkou@iit.kharkov.ua

ANASTASIIA S. KIIAN, Researcher of the Department security information systems and technologies of the V. N. Karazin Kharkiv National University. Areas of scientific interests: cryptography, information theory and coding.
Email: nastyak931@gmail.com

YULIIA V. ULIANOVSKA, PhD, Department of Computer Science and Software Engineering University of Customs and Finance (Dnipro, Ukraine). Research interests: artificial intelligence systems, data processing methods.
Email: yuliyauyv@gmail.com

TATYANA Y. KUZNETSOVA, Researcher of the Department security information systems and technologies of the V. N. Karazin Kharkiv National University. Areas of scientific interests: computer systems and components, security information systems and technologies.