Improved Gentry's Fully Homomorphic Encryption Scheme: Design, Implementation and Performance Evaluation

Svitlana Kazmirchuk [1[0000-0001-2222-251X]], Anna Ilyenko [1[0000-0001-8565-1117]], Sergii Ilyenko [1[0000-0002-0437-0995]], Yakovenko Olesya [1[0000-0003-2998-9767]], Marharyta Herasymenko [1[0000-0003-3736-8776]] and Maksim Iavich [2[0000-0003-3731-4276]]

[1] National Aviation University, Kyiv, Ukraine
{sv.kazmirchuk,ilyenko.a.v,ilyenko.s.s,yak_olesya} @nau.edu.ua
[2] Scientific Cyber Security Association, Tbilisi, Georgia
m.iavich@scsa.ge

Abstract. Cryptographic homomorphic encryption algorithms provides secure communication between users. Homomorphic encryption guarantee message integrity and confidentiality of information about the origin of a message. In the present paper we describe existing cryptographic homomorphic encryption and decryption algorithms. The conducted studies made it possible to determine the ways of the improve Gentry cryptosystem. In this paper, we define criteria and requirements for the formation of modern homomorphic encryption systems. In this paper, we present a new Gentry scheme with additional encryption scheme. The main difference between the proposed scheme was the replacement additional encryption of the session key which minimizes the time for software encryption and decryption operations and increases cryptographic stability, software performance assessment and reliability of an algorithm as for cryptoanalysis. Improved algorithm will be a perfect tool for ensuring the confidentiality of information, using "cloud" computing, because protecting information from unauthorized access is one of the most pressing problems.

Keywords: homomorphic encryption, integrity, confidentiality, cryptographic stability, software performance assessment.

1 Introduction

The rapid increase in the volume of information flows in modern information and communication systems (hereinafter – ICSM) and networks places increased demands on the technical characteristics of communication networks and also on the introduction of new modern methods of cryptographic protection in order to ensure the confidentiality, integrity, and accessibility of information resources and the information system as a whole [7-10].

Today, various cryptographic encryption methods are usually used to protect the information resources, which are transferred, processed and stored in modern ICSM. Encryption methods allow you reliably and efficiently to protect information from unauthorized access and its familiarizationThe use of cryptographic protection, which refers to the use of text encryption procedures through complex mathematical algorithms is becoming increasingly popular. Homomorphic information encryption techniques is one of such methods.

The concept of “homomorphic encryption” was first used in 1978 after the development of the well-known asymmetric RSA algorithm . But their first attempts to justify the necessity and possibility of the practical application of homomorphic encryption were unsuccessful. In 2009, IBM Craig Gentry proposed a model of a fully homomorphic cryptographic system by means of which it became possible to implement addition and multiplication over the encrypted data without their preliminary decryption [2-5].

Currently, cryptographic homomorphic encryption algorithms are widely used in automated systems, cloud computing, and implemented as hardware, software, and/or software-hardware methods. This article will describe how to improve Gentry 's cryptosystem and evaluate the performance of programs that perform clear-text cryptographic operations, using homomorphic encryption algorithms based on ensuring the integrity and confidentiality of information.

To achieve this objective, the following has been done: various methods of homomorphic encryption are analyzed, and the advantages and disadvantages of completely and partly homomorphic systems are determined; identified further ways to improve Gentry 's algorithm, for their further use in the procedure of encryption and decryption of information; software implementations have been written to perform an encryption and decryption procedure, using the RSA, El Gamal, Gentry algorithm, and the advanced Gentry algorithm to conduct further productivity assessment according to ISO/IEC 14756:2010; based on the results of the experiments, conclusions were drawn about the efficiency of using the improved Gentry algorithm to solve the task of ensuring the integrity and confidentiality of information [1-3].

The purpose of this article is to analyze and consider the practical bases of the homomorphic encryption and decryption system. In this paper, we propose a improved Gentry's cryptosystem. The main difference between the proposed scheme was the replacement additional encryption of the session key which minimizes the time for software encryption and decryption operations and increases cryptographic resistance.This approach allows solving the problem of information protection, including stored information, processed and transmitted in modern information networks on the basis of confidentiality and integrity.

2 Related Works

There are different classifications of modern homomorphic encryption and decryption systems. By the concept of homomorphic encryption, we consider a model of

encryption, which allows us to perform certain mathematical actions with encrypted text and to obtain an encrypted result, which corresponds to the result of a similar operation carried out with clear text [11,12]. Modern homomorphic encryption systems are divided into two classes: partly homomorphic systems (RSA Cryptosystem, El Gamal, Paillier) and fully homomorphic encryption systems [13].

By the concept of partly homomorphic systems, we mean cryptosystems, which are homomorphic relatively only one mathematical function (addition or multiplication). The most basic and effective partly homomorphic algorithms are described below [5,6].

RSA cryptosystem. The asymmetric encryption algorithm is one of the best known and most efficient encryption algorithms for the information data flow. The algorithm is partly homomorphic because it has the property of homomorphic concerning the operation of multiplying open texts [10].

Let's take N as the algorithm module, $N = p \cdot q$, where p and q are mutually prime numbers, m_1 is open exponent, mutually simple with $\varphi(n)$, m_1 is open text, k is a public key, encryption function is: $E[(N,e),m_1] = m^e \ mod \ N$

At the same time for any values of m_1 and m_2 the homomorphic condition, concerning multiplication operation is fulfilled:

$$E(k,m1)\cdot E(k,m2) = m1^e \cdot m2^e \cdot mod\ N = E(k,m1\cdot m2)$$

ElGamal cryptosystem. It is the simplest encryption algorithm based on a discrete logarithm. Some open parameters can be a majority for users of information exchange, unlike RSA in the El Gamal algorithm. These are called domain (key) settings. In the ElGamal Cryptosystem in a cyclic group, if the public key is, where, is the private key, the encryption function is as follows: $E(m) = \left(g^r, m\cdot h^r\right), r \in (0....,q-1)$

At the same time for any values of m_1 and m_2 the homomorphic condition concerning multiplication operation is fulfilled:

$$E(m1)\cdot E(m2) = \left(g^{r1}, m1\cdot h^{r1}\right)\cdot\left(g^{r2}, m2\cdot h^{r2}\right) = \left(g^{r1+r2}, (m1\cdot m2)\cdot h^{r1+r2}\right) = E(m1\cdot m2)$$

Thus, it can be argued that the El Gamal cryptosystem is homomorphic for the multiplication operation [15, 21].

Paillier cryptosystem. This cryptographic algorithm is based on the principle of large number factorization, which is the product of two primes.

The system is homomorphic for the addition operation since we know the public key and ciphertext m_1 and m_2, corresponding to the public texts, we can calculate the open text ciphertext $(m_1 + m_2)$.

This can be proved as follows. Let p and q be mutually prime numbers, $N = p \cdot q$. Next, we calculate M : $M = \left(L \cdot (g^L \bmod n^2)\right)^{-1} \bmod n$

A number g and r is selected from the Z space, the public key is a pair of numbers (n,g), the private key is a pair of numbers (L,M). To perform a open text encryption operation, the following calculation is performed: $C = g^m \cdot r^n \, mod \, n^2$.

In this case, for any values of m_1 and m_2, the homomorphic condition is fulfilled

$$E(m1) \cdot E(m2) = \left(g^{m_1} \cdot r1^n\right) \cdot \left(g^{m_2} \cdot r2^n\right) = \left(g^{m_1+m_2} \cdot r_1 \cdot r_2{}^n\right) = E(m_1 + m_2) mod \, n^2$$

In this case, the homomorphic property is not described as in the above-mentioned algorithms, since the result of two encrypted digits is their sum, that is $E(k,m_1) \cdot E(k,m_2) mod \, n^2$, after decryption we will get $(m_1 + m_2) mod \, n^2$.

By fully homomorphic systems we consider cryptosystems, which allow to perform operations "+" and " * " on encrypted data in such a way that the result of decryption matches with the result of performing the same operation on unencrypted data. The most common is the Gentry cryptosystem [11,12,14].

Gentry cryptosystem. Let 's take a look at their proposed schema for using the example of calculations in Z space.Let p be an odd number,. This number is a secret parameter. Suppose that binary bits are encrypted, so m is plain text, 0, or 1. Then let's pick a number $z = 2 \cdot r + m$, from here $z = m \, mod \, 2$.

The encryption procedure is as follows:

For each value of M the following function is calculated: $C = z + p \cdot q$,where q is an arbitrary number [2-5].

According to the encryption function is as follows: $C = 2 \cdot r + m + (2 \cdot k + 1) \cdot q$.

Then the decryption procedure consists of the following mathematical procedures [6]. Let's take the numbers *c* and *p* as a known ones, where *c* is an encrypted number, and *p* is a secret parameter. The decryption procedure includes:

$$r = c \, mod \, p = (z + p \cdot q) mod \, p = z \, mod + p \cdot q \, mod \, p \, .$$

The parameter is called a noise, which possible values ranging from $(-p/2; p/2)$. Next, we get an open text: $m = r \bmod 2$

This algorithm is completely homomorphic. This is possible to prove it as follows.

Suppose there are 2 numbers m_1 and m_2. Let's compare them with a pair of numbers Z_1 i Z_2:

$$Z_1 = 2r + m_1; Z_2 = 2r + m_2 \, . \qquad (1)$$

Secret parameter $p = (2 \cdot k + 1)$ is odd number. The encryption functions are as follows:

$$c_1 = z + pq_1; c_2 = z + pq_2 \, . \qquad (2)$$

Then their sum and product will be equal:

$$c_1 + c_2 = z_1 + z_2 + p(q_1 + q_2) = 2r_1 + m_1 + 2r_2 + m_2 + p(q_1 + q_2) = \\ = 2(r_1 + r_2) + m1 + m2 + (2k + 1)(q_1 + q_2) \quad (5)$$

$$c_1 c_2 = z_1 z_2 + p(z_1 q_2 + z_2 q_1) + p^2 q_1 q_2 = \\ = (2r_1 + m_1)(2r_2 + m_2) + 2(kz_1 q_2 + z_2 q_1) + z_1 q_2 + z_2 q_1 = \\ = 4r_1 r_2 + 2(r_1 m_2 + r_2 m_1) + m_1 m_2 + 2k(z_1 q_2 + z_2 q_1) + \\ + 2r_1 q_2 + 2r_2 q_1 + m_1 q_2 + m_2 q_1 \quad (4)$$

Using the decryption procedure will produce the following result:

$$(c_1 + c_2) mod\ 2 = [2(r_1 + r_2) + m_1 + m_2] mod\ 2 = (m_1 + m_2) \quad (6)$$

The result cannot be decrypted without knowing the secret parameter p:

$$(c_1 + c_2) mod\ 2 = [m_1 + m_2 + q_1 + q_2] \quad (7)$$

Using formula (4) for decryption, a similar result is obtained:

$$(c_1 c_2) \bmod 2 = [2(r_1 + r_2) + m_1 m_2] \bmod 2 = (m_1 m_2) \quad (8)$$

Thus, Gentry 's algorithm has been proven to be completely homomorphic encryption [4, 5, 12-16].

Gentry's cryptosystem is a complete homomorphic encryption algorithm, which allows to perform mathematical operations with the encrypted text and obtain an encrypted result, that corresponds to the result of a similar operation carried out on open text. The fact that such operations can be carried out is, undoubtedly, the main advantage of the algorithm and distinguishes it among others. However, the mathematical implementation of the algorithm itself has certain disadvantages, which allow crackers to obtain the open text, based on a known key and ciphertext, using attacks on algorithms and cryptanalysis tools. Of course, apart from the possibility of cryptanalysis by attackers, the algorithm would be a perfect tool for ensuring the confidentiality of information, using “cloud” computing, but nowadays, protecting information from unauthorized access is one of the most pressing problems. So, it is important to identify “weaknesses” in the algorithm and possibly eliminate them by introducing additional parameters and functions into the algorithm.

3 Proposed homomorphic encryption system with additional parameters

The previous sections provided the advantages and reasonable choice of algorithms of the homomorphic encryption system. As noted above, one of the weaknesses of the algorithm is that the same session key is used for encryption and decryption. Therefore, attention should be paid to the possibility of optimizing the algorithm to

improve the reliability of the generated key. To increase the cryptographic resistance of a given algorithm, it is better to use an encryption scheme in which the session key is further encrypted using an asymmetric RSA algorithm and transmitted to the communication channel in an encrypted form. It will provide cryptofirmness and reliability of an algorithm as for cryptoanalysis and interpretation of a key the attacker needs to solve a problem of *n* parameter decomposition on simple factors of p and q, and at successful selection of these parameters (not less than 1024 bits) such a problem is almost impossible under the current conditions.

3.1. Mathematical foundations of the improved Gentry algorithm

Algorithm proposes to use a key pair (public and private) to encrypt the secret session key.

Let's use additional parameters *n, p, q,* with $n = pq$, where *p, q* are integer numbers, which will be generated using generator of taken numbers. Next, the Euler function $\varphi(n)$, that is equal to the product of the numbers, will be calculated. After calculating this parameter, the selected number *e* is such that $1 < e < \varphi(n)$, and using the Euclid algorithm, the number d will be calculated such that $ed \equiv 1(mod(\varphi(n)))$.

So, Gentry's upgraded algorithm will be as follows. To begin with, we will perform a procedure to find an open and secret asymmetric RSA algorithm, using a cryptographic-resistant prime generator. Next, we will perform a procedure to find the number and calculate the Euler function.

Next, we will select and calculate the public key *e*, where $3 \le e < \varphi(n)$, and also the secret key *d* according to the Euclid algorithm $ed \equiv 1(\mathrm{mod}(\varphi(n)))$.

The next step is to find the secret parameter. Let *x* be an odd number. It is also a secret parameter. Suppose that binary bits are encrypted, so *m*, open text, accepts values 0 or 1. Then let's pick a number $z = 2 \cdot r + m$, from here $z = m \bmod 2$.

The encryption procedure is as follows:

For each value of M the following function is calculated:

$$C = z + x \cdot h, \tag{9}$$

where *h* is an arbitrary number.

According to the encryption function, it is as follows:

$$C = 2 \cdot r + m + (2 \cdot k + 1) \cdot h \tag{10}$$

To encrypt a session key (secret parameter *x*), using an asymmetric RSA algorithm: $X = x^e \ mod \ n$, where *(e,n)* is the public key of the RSA cryptosystem.

To decrypt a session key (secret parameter *x*) using an asymmetric RSA algorithm $x = X^d \ mod \ n$, where *(d,n)* is the secret key of the RSA cryptosystem.

Then the procedure for decrypting the cryptogram consists of the following mathematical procedures (formulas 1-8): $r = c \ mod \ x$ is decoding function; $m = r \ mod \ 2$ is clear text.

The advanced Gentry algorithm is more reliable and cryptographic-resistant than the conventional method because it doesn't allow an attacker to decrypt a message based on a public key, which is transmitted over communication channels, since session key decryption is only possible with parameter d, which is not transmitted over the channel and can be protected by additional means. At the same time, with the correct selection of parameters p and q (that have length at least 1024 bits), it is almost impossible for the attacker to determine p and q, based on a known n (a problem of factorization of composite number). Also, due to the modernization of the algorithm, it will be achieved that the attacker will not be able to effectively implement the attack on the known ciphertext, as for its successful implementation it will be necessary to obtain the parameter d.

3.2. Prospects for the practical implementation of the improved Gentry algorithm

A completely homomorphic cryptographic system of information protection with sufficient speed of operation is a necessary condition for ensuring the operability of many modern application programs. This system allows statistical and mathematical calculations, data retrieval, electronic voting, commitment schemes, multilateral secret calculations and any other operations with encrypted data. It also has to guarantee both high data processing speed and data confidentiality [7, 8, 17].

In addition to research in homomorphic encryption, developments are also underway based on other schemes and in other areas. For example, one of the last achievements in the information security sphere in cloud computing is the development of the program CryptDB complex, that uses mechanisms of homomorphic enciphering. The CryptDB provides encryption support in which data on the database management systems (DMS) side never appears in the public form, and all requests, transmitted to the DMS, contain only encrypted data, including conditional blocks. All actions are carried out only with encrypted data by using CryptDB in the process of executing SQL queries. So, the user can send the SQL query to the DBMS and obtain the result without decrypting the information on the server-side (the data will be decrypted with the client equipment). To ensure the confidentiality of information, a multilevel encryption system is used in which different data is placed on different nested cryptographic layers, each of the layers has its key and supports a limited set of simple operations on encrypted data. Each layer uses its homomorphic encryption methods to hide data. However, only fully homomorphic encryption can eliminate the need for even partial data decryption to perform calculations on them. Having solved this problem, homomorphic encryption is also not the most optimal encryption scheme, as it is fundamentally vulnerable to attacks on the selected ciphertext. Unfortunately, at the moment, there is no quality information security system based on a homomorphic encryption scheme that addresses both privacy and usability, computation speed, and performance issues.

Therefore, in the future, effective use of such a security system requires implementation that meets the following conditions: possibility of using the complete set of mathematical functions in the procedure of encryption and decryption; the accuracy and speed of calculations should be constant at all stages of encryption and

decryption; the key tuple must be so large that it is not possible to completely overrun all possible keys; the size of the encrypted data and the length of the key have no significant impact on system performance.

3.3. The directions of performance evaluation of homomorphic encryption algorithms according to ISO/IEC 14756:2010

The term "software performance assessment" will further be understood as an activity that includes all methods of evaluating data processing performance in information systems where performance is expressed by numerical characteristics, and may also include an assessment of how qualitatively the performance meets user requirements.

For an information system user, the critical question is whether its performance will be sufficient to perform the necessary calculations. There are currently a large number of methods for describing and measuring software performance. Each method was designed for a certain type of data processing system and use in a specific environment. To solve these problems, ISO has developed a new method that can be applied to a wide range of data processing systems types and applications. ISO/IEC 14756:2010 presents the modern principles of measuring computer performance and measuring the efficiency of software startup and execution times. This standard allows to evaluate software performance. ISO/IEC 14756:2010 defines 3 classes of software performance characteristics: the first-class describes what the computer can do. It is a set of measures, the correctness of work and calculated results, that is, characterizes convenience; the second class describes how stable and consistent the computer is during operation. This applies to the reliability of operation in the broadest sense; the third class relates to the speed of operation. One of the most important aspects is the speed of tasks, that is, the time to deliver the results of tasks. Another aspect is a number of tasks that can be performed at a given time [1, 18].

The evaluation of the performance of developed software implementations of homomorphic encryption and decryption algorithms has been reduced to determining the program execution time of encryption/decryption functions, key generation and determining factors affecting these indicators to determine the "weaknesses" of the program and algorithm. That is, performance indicators of the algorithm were determined. This procedure will be performed in several steps. A necessary condition for this is to conduct a pilot study, to determine the time of execution of program functions after changing the factors, which affect the performance of the program implementation change [19, 20].

The software performance evaluation method defined in ISO/IEC 14756 is based on the principle under which the evaluated system is considered as a "black box". The system consists of hardware, software and network components. This set is considered as a black box, which is connected through a set of its interfaces for its users. Users are usually people or machines who submit tasks to the system via an interface. By linking this item in software implementations to be evaluated, it can be argued that the tool is a data processing system with a priori unknown quality and performance indicators. The estimated system consists of the equipment (the personal computer on where the program is installed), software and users.

The results of the performance determination, whether it is done by measurement or forecasting, are the values of performance, that is, physical quantities. The overall score is not numeric, ("bad", "enough" or "excessive"). We must determine which ranges of performance values correspond to each of these three numeric values.

Performance *P* is defined as a set of the following indicators: *P = (B, TME, E)*, where *B* is the pass vector, *TME* is the average time of vector *B*, *E* is the regression vector. The measured performance *P* is a set of physical quantities. The user of the system is interested in whether they fully meet the user's requirements. After the measured performance values, it is possible to conclude how much these indicators suit the user of the system, from the obtained values of coefficients it is possible to determine which of them have the greatest impact on the performance of the program, how it is possible to increase or reduce the impact of certain coefficients on performance. Performing experimental studies in the work, the results of determining the performance of the software can indicate "weaknesses" of the program or cryptographic algorithm, which is implemented in software form. At the same time, upgrading these shortcomings will improve the performance and efficiency of the software.

During the performance evaluation mechanism of software and cryptographic algorithms, which were implemented in them, was used and the following was taken into account: the following parameters of productivity were calculated for software productivity assessment according to ISO 14756:2010 [1, 21, 22]: total bandwidth (throughput vector), average time of tasks performance; regression coefficients; to evaluate performance, all the functions, that the program implements and the factors that are most influential in software performance have been identified; by calculating the regression coefficients, it can be further stated which of the following factors are the most influential on the performance of the software and, based on these calculations, algorithms will be upgraded to improve the performance of cryptographic functions and algorithms.

4 Results

As a result of the proposed scheme, a software implementation of the procedure for encryption and decryption in C # was obtained using the CryptoLib library. This library fully supports cryptography algoritms and is certified for use in Ukraine at the state level. The algorithms were tested in the Crypto ++ 5.6.0 software environment on a dual-core Intel Core 1.83 GHz processor running Windows 8 32 bit x86 The comparative characteristic of the results of the program studies and the performance evaluation of the software implementing the RSA, El-Gamal, Paillier, Gentry algorithms and the improved Gentry algorithm, respectively, occurred according to the following parameters: the number of program executable functions, the total bandwidth vector B (M), the average execution time of the program functions , the ratio of time spent to complete all program features and additional settings. Below is a summary table of comparison (Table 1). So you can draw a conclusion about the changes in the encryption and decryption procedure. Firstly, considering the performance of the algorithm, namely the speed of encryption and decryption

operations, the improved Gentry algorithm provides much higher cryptocurrency due to the additional encryption of the session key, but at the same time due to the complexity of mathematical calculations, the time to perform software functions is reduced. The quantitative value of the average execution time of the encryption/decryption process for all experiments is much smaller than in the known methods (reduction of time from 1.17 to 1.31 times, depending on the method used). Secondly, there is a factor that can significantly reduce the speed of encryption and decryption operations. This factor is the exponent *e*. The execution time increases with the number of non-zero bits in the binary representation of the open exponent *e*. To increase the encryption speed, *e* is often required to be 17, 257, or 65 537 prime numbers whose binary representation contains only two units: 17 (10) = 10001 (2), 257 (10) = 100000001 (2), 65537 (10) = +100000000000000012 (prime numbers). This fact was verified with performing experiments to measure the speed of encryption and decryption operations. It should be noted that it is important to use of pseudorandom number generator using in forming the algorithm parameters to increase the cryptocurrency. For selecting the parameters of the algorithm, we should also take into account that if the index $d < n^{1/4}$, crypto analysts can successfully execute a Wiener attack to find *d*, based on the theory of continuous fractions.

Table 1. Comparison of modern algorithms for the homomorphic encryption and decryption

Algorithm	**Open key, bit**	**Number of functions performed by the program**	**Total throughput vector B *(m)***	**The ratio of time that has been spent to perform all program**	**Average time to execute soft-ware functions of all experiment,*s***
RSA	512-4096	7	0,6193	21,79%	4,69
El Gamal	1024-4096	8	0,5768	20,57%	4,35
Paillier	1024-4096	7	0,7245	23,79%	4,72
Gentry	512	7	0,3290	25,1%	5,25
Improved Gentry	512	9	0,2958	15,1%	4,01

5 Conclusions

Thus, in this article we give a full description of the improved of the Gentry for the encryption and decryption of information using additional operation. Given that the schemes of the algoritm of the classical and improved Gentry are similar, the modification chosen did not require major changes. The main difference was use an additional encryption scheme in which the session key is further encrypted using an asymmetric RSA algorithm and transmitted to the communication channel in an encrypted form. As a result of the modification, the the ratio of time that has been spent to perform all program default functions to time is set a priori to perform all functions has decreased to 15,1% and the average time to execute software functions of all experiment has decreased by 23.6%, from 1,17 to 1,31 times, depending on the

method used. Thus, in this way, the modification of the Gentry algorithm for the encryption and decryption of information provide cryptographic stability, software performance assessment and reliability of an algorithm as for cryptoanalysis.

References

1. ISO/IEC. ISO/IEC 14756:2010, Information technology — Measurement and rating of performance of computer-based software systems (2010).
2. Gentry C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st ACM Symposium on Theory of Computing – STOC 2009, pp. 169-178. ACM, New York (2009).
3. Gentry C.: Toward basing fully homomorphic encryption on worst-case hardness. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 116-137. Springer, Heidelberg (2010).
4. Gentry C., Halevi S.: Implementing gentry's fully-homomorphic encryption scheme. Cryptology ePrint Archive, Report 2010/520 (2010).
5. Gentry C., Halevi S.: Implementing Gentry's Fully-Homomorphic Encryption Scheme. In: Paterson K.G. (eds) Advances in Cryptology – EUROCRYPT 2011. LNCS, vol 6632, pp. 129–148. Springer, Berlin, Heidelberg (2011).
6. Stehlé D., Steinfeld R.: Faster fully homomorphic encryption. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 377-394. Springer, Heidelberg (2010).
7. Kazmirchuk S., Anna I., Sergii I.: Digital signature authentication scheme with message recovery based on the use of elliptic curves. In: Hu, Z., Petoukhov, S., Dychka, I., He, M. (eds.) ICCSEEA 2019. AISC, vol. 938, pp. 279-288. Springer, Cham (2020), https://doi.org/10.1007/978-3-030-16621-2_26.
8. Vysotska O., Davydenko A.: Keystroke Pattern Authentication of Computer Systems Users as One of the Steps of Multifactor Authentication. In: Hu Z., Petoukhov S., Dychka I., He M. (eds) ICCSEEA 2019. AISC, vol. 938, pp. 356-368. Springer, Cham (2020), https://doi.org/10.1007/978-3-030-16621-2_33.
9. Korchenko O., Davydenko A., Vysotskaya O.: Method of authentication of information systems users by their handwriting with multi-step correction of primary data, Information security, vol. 21, №1, pp. 4051 (2019), https://doi.org/10.18372/ 2410-7840.21.13546.
10. Yudin O., Ziatdinov Yu., Voronin A., Ilyenko A., Basic Concepts and Mathematical Aspects in Channel Coding: Multialternative Rules, Cybernetics and Systems Analysis, vol. 52, pp. 878-883 (2016). https://doi.org/10.1007/s10559-016-9889-z.
11. Rivest R, Adleman L, Dertouzos M.: On data banks and privacy homomorphisms. In: Foundations of secure computation, Academic Press, pp 169-177 (1978)
12. Menezes A., Vanstone S., Van Oorschot P.: Handbook of Applied Cryptography. CRC Press, London (1996).
13. Schneier B.: Applied Cryptography, 2nd edn. John Wiley & Sons, Inc., New Jersey, USA (2015).
14. Yi X., Paulet R., Bertino E. Homomorphic Encryption. In: Homomorphic Encryption and Applications. SpringerBriefs in Computer Science. Springer, Cham (2014)
15. P. Paillier, D. Pointcheval, Efficient public-key cryptosystems provably secure against active adversaries, in Proceedings of Advances in Cryptology, ASIACRYPT'99, 1999, pp. 165-179.
16. T. El Gamal, A public-key cryptosystem and a signature scheme based on discrete

logarithms. IEEE Trans. Inf. Theory 31(4), 469–472 (1985).
17. Oksiiuk O., Chaikovska V., Fesenko A. Security technique for authentication process in the cloud environment, Proceedings of 2019 IEEE International Scientific-Practical Conference: Problems of Infocommunications Science and Technology, PIC S and T 2019, pp. 379-382, 2019.
18. S. Gnatyuk, V. Kinzeryavyy, M. Iavich, D. Prysiazhnyi, Kh. Yubuzova, High-Performance Reliable Block Encryption Algorithms Secured against Linear and Differential Cryptanalytic Attacks, CEUR Workshop Proceedings, vol. 2104, pp. 657-668, 2018.
19. Fedushko, S., Ustyianovych, T., Gregus, M. Real-time high-load infrastructure transaction status output prediction using operational intelligence and big data technologies. (2020) 9 (4), art. no. 668. DOI: 10.3390/electronics9040668
20. A. Kuznetsov, I. Svatovskij, N. Kiyan and A. Pushkar'ov, Code-based public-key cryptosystems for the post-quantum period, 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, 2017, pp. 125-130. DOI: 10.1109/INFOCOMMST.2017.8246365.
21. Iavich M., Gagnidze A., Iashvili G., Gnatyuk S., Vialkova V. Lattice based Merkle, CEUR Workshop Proceedings, vol. 2470, pp. 13-16, 2019.
22. M. Iavich, S. Gnatyuk, E. Jintcharadze, Yu. Polishchuk, R. Odarchenko, Hybrid Encryption Model of AES and ElGamal Cryptosystems for Flight Control Systems, Proceedings of the 2018 IEEE 5th International Conference on Methods and Systems of Navigation and Motion Control, October 16-18, 2018, Kyiv, Ukraine, pp. 229-233.