ADVANCES IN CYBER-PHYSICAL SYSTEMS

Vol. 9, No. 1, 2024

ENCRYPTION METHOD BASED ON CODES

Alina Davletova[1], *Vasyl Yatskiv*[1], *Stepan Ivasiev*[1], *Mykola Karpinskyi*[2]

[1] *West Ukrainian National University, 11, Lvivska Str, Ternopil, 46009, Ukraine.*
[2] *University of the National Education Commission, 2, Podchorążych Str, Krakow, 30-084, Poland.*
Authors' e-mail*: a7davletova@gmail.com, jazkiv@ukr.net, stepanivasiev@gmail.com, mpkarpinski@gmail.com*

https://doi.org/10.23939/acps2024.01.024

Submitted on 30.03.2024

***Abstract*: This paper proposes an improvement of the McEliece asymmetric cryptosystem based on code-based cryptography by replacing the permutation matrix with a modulo operation and using a finite field GF(q).**

This approach increases the complexity of the decryption process for potential attackers, providing a high level of cryptographic security without changing the length of the key. The article provides a diagram of the improved operation of the cryptosystem and describes examples of application. An analysis of the number of possible combinations of matrices has been carried out for different implementation options of code (7,4) based on different numerical systems. It has been shown that achieving cryptographic security comparable to the original McEliece cryptosystem requires the use of q ≥ 5.

***Index Terms*: asymmetric cryptosystem, code-based cryptography, computational complexity, data encryption, finite field.**

I. INTRODUCTION

Asymmetric cryptosystems play a crucial role in ensuring the security of electronic communications, financial transactions, and the storage of confidential information. However, with the development of quantum computing technologies, standard asymmetric cryptosystems such as RSA and ECC become vulnerable to potential attacks using quantum computers, which can efficiently solve the problems of factorization and discrete logarithms [1]. Therefore, the development and research of post-quantum cryptographic algorithms are currently a relevant task.

One of the promising directions in this field is the development and improvement of code-based cryptosystems, which are characterized by a high level of security due to the complexity of attacks using coding theory.

II. LITERATURE REVIEW AND PROBLEM STATEMENT

Code-based cryptography is considered one of the potential directions for post-quantum cryptography because of its resilience to quantum computing. It utilizes the properties of algebraic codes to create cryptographic algorithms that remain secure even in the face of attacks using quantum computers. The McEliece cryptosystem, known for its resilience, is one of the most extensively researched code-based cryptographic systems.

The work [2] explores the possibilities of using code-based cryptography for multiparty computations and digital signatures. The proposed algorithms based on the McEliece cryptosystem and its modifications demonstrate high resilience to both classical and quantum attacks and efficiency in terms of key size and computation time.

In [3], concrete implementations for representation-based Information Set Decoding (ISD) algorithms, such as May-Meurer-Thomae (MMT) or Becker-Joux-May-Meurer (BJMM), optimized for the McEliece and quasi-cyclic schemes like BIKE and HQC are introduced. Despite higher memory consumption compared to naive ISD algorithms, such as P range, MMT and BJMM offer significant speedups for practical cryptanalysis on medium-sized instances. The paper provides record computations for McEliece-1223 and McEliece-1284, as well as for quasi-cyclic settings up to a code length of 2918. Based on these computations, the paper extrapolates the bit-security levels of proposed BIKE, HQC, and McEliece parameters in NIST's standardization process.

In [4], proposes to use non-cyclic noise-resistant codes on elliptic curves in a modified McEliece cryptosystem. The main criteria for constructing a modified crypto code based on the McEliece scheme on elongated elliptic codes are investigated.

In [5], an enhancement of the McEliece cryptosystem is presented by using sparse matrices, quantum resistant codes, and quantum key distribution (QKD) algorithms. An improved algorithm is proposed that combines various cryptographic methods such as homomorphic encryption and salted hashing, contributing to increased resilience of the cryptosystem against quantum and classical attacks.

In [6], a realization of the McEliece cryptosystem is proposed using the construction (C1, C1+C2) to generate a new code from two arbitrary linear codes. The proposed approach achieves a high level of security and reduces the key size by 25% compared to the classical McEliece implementation.

The paper [7] the algorithm for generating message authentication codes using a McEliece's cryptosystem based on universal hashing functions have been investigated.

III. SCOPE OF WORK AND OBJECTIVES

The aim of this work is to improve public key generation efficiency and cryptographic security by expanding the range of code word combinations through modular arithmetic, thereby increasing difficulty for attacks and requiring more computing resources.

IV. INVESTIGATION OF THE CLASSICAL MCELIECE CRYPTOSYSTEM

The main principles of the McEliece cryptosystem involve the utilization of cyclic codes that complicate decoding. The classical cryptosystem is implemented based on Goppa codes [8], which are efficient in terms of algorithmic operations. However, in certain cases, Hamming codes [9-10] may be more efficient or simpler to implement compared to more complex Goppa codes. The structure of Hamming codes facilitates their implementation, especially in conditions of limited computational resources, and can hinder attacks, which is important in the context of cryptographic applications. Hamming codes are efficient in error detection and correction, making them suitable for applications where data transmission encounters interference.

The message exchange scheme between the sender and the receiver in such a system (Fig.1) occurs as follows [11,12].

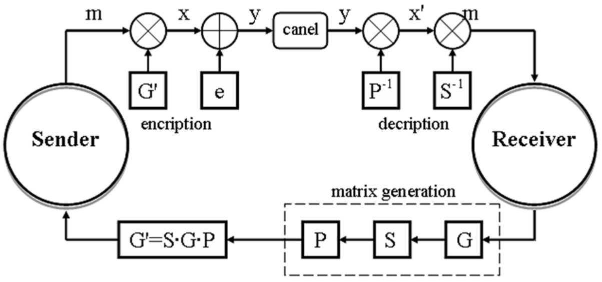

Fig. 1. McEliece coding scheme

1) Initialization:

- sender and receiver generate public and private keys;
- public keys are formed from generator matrices *G s* and *G_r* respectively, which are generators of codes for the sender and receiver, and from scrambler matrices *S_s* and *S_r* for altering the characteristics of the transmitted message;
- private keys are represented by the permutation matrices *P_s* and *P_r*.

2) Encryption of message:

- the sender encrypts the message *m* for the receiver using their public key G'_r and obtains the encrypted message *x*;
- the sender selects a random error vector *e* and computes $y = x + e$;
- the encrypted message *y* with error vector *e* is sent to the receiver.

3) Message decryption:

- the receiver gets the encrypted message *y* and the error vector *e*;
- the receiver computes $x' = yP\_r^{-1}$, where P_r^{-1} is a part of their private key used to restore the bit order in the received message;
- by performing $x'S\_r^{-1}$, the receiver can recover the original message *m*.

The complexity of the cryptosystem's operation can be characterized by the following criteria:

1) Temporal complexity determines how the execution time of the algorithm changes depending on the size of the input data. In the McEliece system, it is determined by the size of matrices and vectors, as well as the need to perform computationally complex operations over Galois fields.

2) Hardware complexity determines the amount of memory, computational resources required for executing the algorithm. In the case of McEliece, the memory required to store matrices, vectors, and other intermediate data can be significant, especially for large key sizes.

3) Cryptographic complexity determines the algorithm's resistance to cryptographic attacks. This parameter in the McEliece system is determined by the complexity of recovering the private key based on the public key, which is based on solving the code decoding problem, which is NP-hard.

4) Computational complexity is determined by the number and complexity of operations required to encrypt and decrypt data. Since McEliece is based on algebraic operations over matrices and Galois fields, its complexity includes algebraic operations such as matrix multiplication, inversion of elements in the Galois field, etc.

The listed criteria are important to assess the efficiency and suitability of the algorithm for practical use. Each criterion affects the overall complexity of the encryption algorithm. For example, an algorithm with high cryptographic complexity may be more costly in terms of time and hardware complexity. Therefore, when developing a cryptographic algorithm, it is necessary to balance between different aspects to ensure an effective and secure operation.

Fig. 2 illustrates the operation algorithm of the McEliece cryptosystem in its classical implementation using Hamming codes.

The key generation process exhibits the highest temporal and computational complexity among cryptographic algorithms. This happens because this process typically involves generating a large number of random bits or numbers and performing operations with large data volumes. To determine the time required to generate the code generator matrix *G*, the permutation matrix *P*, and the scrambler matrix *S*, as well as computing the inverse matrices P^{-1} and S^{-1}, algorithmic complexity analysis can be used.

Estimating the complexity of generating an $n \times n$ matrix takes into account the fact that each element in every row and column needs to be modified. Since the number of rows and columns is equal to *n*, the total number of elements to be modified is n^2. Thus, the complexity of generating such a matrix is assessed as $O(n^2)$.

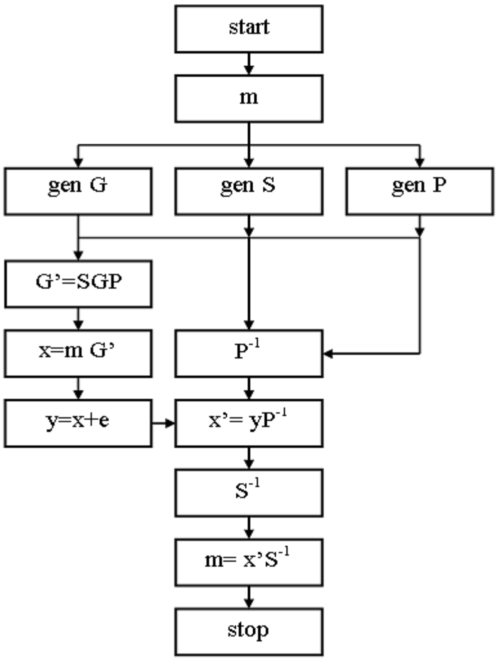

Fig. 2. Scheme of the operation of the cryptosystem in the classical implementation

To ensure the possibility of decoding a codeword, certain conditions must be met:

– the generated matrix must be square;

– its determinant must not be zero.

This condition ensures the existence of an inverse matrix, which is crucial to recovering the original message during decryption. Various algorithms are used to compute the inverse matrix, such as the Gauss-Jordan method or the LU decomposition method.

The overall complexity of computing the inverse matrix amounts to $O(n^3)$, where n is the size of the matrix or vectors in the respective operations. Therefore, the total time to generate the matrix and computing its inverse appears as $O(n^2)+O(n^3)=O(n^3)$,since n^3 dominates over n^2 for sufficiently large values of *n*.

In the case where $\det(matrix)=0$, the matrix is singular, and lacks an inverse matrix. This situation may occur if one of the matrix rows is a linear combination of other rows.

Considering the possibility of generating singular matrices, the time for generating and computing matrices for key formation will depend on the specifics of generating and processing singular cases. Therefore, it is necessary to consider cases where: a valid matrix is generated, which can be inverted; a singular matrix is generated, which requires additional processing.

Taking this into account, we can estimate the time it takes to generate and compute the inverse matrix, considering the possibility of generating singular matrices. The need for additional checks for singularity may affect the generation time. Thus, the formula for the matrix generation time can be expressed as $O(f(n))$, where $f(n)$ is a function that accounts for the additional singularity check.

So, the general formulas for estimating time for each case can be as follows:

– for matrix generation: $O(f(n))$;

– for computing the inverse matrix: $O(n^3)$ or $O(g(n))$, depending on the singularity of the input matrix, where g(m) is the complexity of additional processing of the singular matrix.

Generation of the *G* code generator matrix is based on algorithms that utilize the characteristics of Hamming codes to ensure error correction during transmission over communication channels. The formation occurs without significant time costs and does not affect the overall algorithmic time.

The scrambler matrix *S* is formed using random or pseudo-random numbers to create variation parameters that affect the properties of the transmitted message, with the aim of improving the reliability or security of transmission. This matrix must satisfy the invertibility condition to ensure the possibility of recovering the original message during decryption.

In the case of generating matrices in a modular arithmetic system (mod *p*), with an increase in the base p, the number of possible element values also increases. Thus, expanding the range of possible values from 0 to $p-1$ promotes diversity in valid matrices, potentially reducing the time required for their generation.

The results of experimental research on the number of iterations for forming the matrix *S* ($k \times k$), where *k*=4, with values of *p* = 2, 5, 7, 13 are presented in Table 1.

Table 1

The number of iterations to form the matrix *S* at various values of *p*

Experiment	*mod(2)*	*mod(3)*	*mod(5)*	*mod(7)*	*mod(13)*
1	9	10	1	1	1
2	17	9	3	2	2
3	2	4	3	1	1
4	91	12	1	1	2
5	12	1	4	3	1
6	7	3	1	1	1
7	22	12	4	1	1
8	14	10	1	1	1
9	15	1	3	2	1
10	34	2	3	1	1
11	38	8	6	5	1
12	12	3	3	1	2
13	76	1	1	2	1
14	6	10	4	3	1
15	88	5	2	2	1

To analyze Table 1, let us check the average number of iterations for each value of *p*:

For *p* = 2: average number of iterations: 443/15 ≈ ≈ 29,533.

For *p* = 3: average number of iterations: 91/15 ≈ ≈ 6,07.

For *p* = 5: average number of iterations: 40/15 ≈ ≈ 2,67.

For p = 7: average number of iterations: 27/15 ≈ ≈ 1,87.

For p = 13:average number of iterations: 18/15 ≈ 1,2.

Based on the data in Table 1, a graph (Fig.3) depicts the average number of iterations to form the matrix S at different values of p has been constructed.

From the graph in Figure 1, the following conclusions can be drawn: the number of iterations is the highest when p = 2, and the lowest number of iterations is observed when p = 13, indicating the efficiency of the matrix S generation process with increasing p.

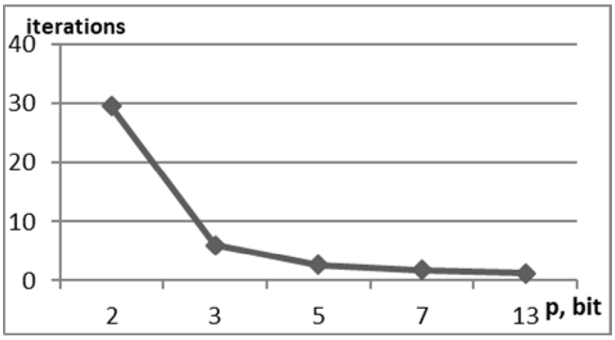

Fig. 3. Average value of the number of iterations of the formation of the matrix S at different values p.

The analysis demonstrates that the generation of matrix S depends on the size of the field represented by the parameter p. According to the experimental results, as the value of p increases (from 2 to 13), the generation time of the scrambler matrix decreases. This can be explained by the fact that with an increase in the size of the field, the number of available elements to form the matrix S increases.

The increased possibilities for selecting element values in the field allow for more effective creation of the scrambler matrix, leading to reduced computational and time complexity, and consequently requiring fewer computational resources. Generation of the permutation matrix P, designed to change the order of bits of the codeword before processing or transmission, is typically carried out using random or pseudorandom methods. This ensures an effective shuffling of elements in the input vector or message. Regardless of the base of the numbering system, the elements of this matrix usually take values of 0 or 1. Matrix P must also satisfy the invertibility condition to allow the restoration of the original vector or message after permutation.

The size of matrix P, $n \times n$ where n is the length of the codeword, is the largest among the necessary matrices, since this size allows for a complete permutation of all elements of the codeword.

Table 2 presents the results of the experimental research on the duration t (seconds) and the number of iterations i required to find the necessary matrix P at different values of n=7, 8, 9, 10, 15.

Table 2

Parameters for generating matrix *P* at different values of *n*

№	*n=7*		*n=8*		*n=9*		*n=10*		*n=11*		*n=12*		*n=14*		*n=15*	
	t	*i*	*t*	*i*	*t*	*i*	*t*	*i*	*t*	*i*	*t*	*i*	*t*	*i*	*t*	*i*
1	0,018	278	0,005	94	0,017	356	0,470	5010	0,304	2899	2,36	21874	35,253	251407	10,657	68506
2	0,008	183	0,030	717	0,019	373	0,003	12	0,142	1333	0,164	1495	2,242	15888	91,810	584384
3	0,007	156	0,028	630	0,095	2140	0,159	1669	0,400	3864	3,442	32225	2,6	18665	26,059	166279
4	0,005	137	0,009	200	0,007	143	0,657	7037	1,944	18893	5,445	50780	6,548	44968	5,081	32724
5	0,013	308	0,020	460	0,043	948	0,312	3343	1,535	14889	1,1	10292	9,600	65684	84,027	538807
6	0,005	108	0,023	525	0,022	457	0,899	9581	1,207	11609	3,445	31888	10,978	79292	19,478	125218
7	0,004	91	0,006	125	0,067	1422	0,142	1510	0,584	5665	1,52	14052	3,553	25507	16,422	105311
8	0,002	58	0,004	73	0,037	778	0,283	3009	0,587	5694	0,447	4175	34,764	249912	211,017	1354150
9	0,003	56	0,011	249	0,047	1049	0,168	1764	1,473	14193	1,72	15941	12,672	90782	31,401	202215
10	0,007	151	0,014	312	0,037	796	0,843	8891	2,948	28539	0,341	3159	19,526	140877	21,198	132289
11	0,013	143	0,073	1789	0,057	1218	0,086	917	0,116	1074	1,437	13363	15,71	111937	42,331	301174
12	0,002	13	0,012	284	0,090	1952	0,426	4626	2,681	27199	0,041	364	42,331	301174	42,756	305723
13	0,002	52	0,030	713	0,060	1243	0,245	2630	0,440	4469	0,801	7517	42,756	305723	12,24	88117
14	0,014	356	0,004	67	0,014	288	0,464	5052	0,309	3130	4,005	37687	12,24	88117	37,864	270913
15	0,007	151	0,014	312	0,037	796	0,843	8891	1,113	11278	0,892	8336	37,864	270913	42,331	301174

The data in Table 2 demonstrate the dependence of time and computational complexity on the size of matrix n. As it is evident from the data, for smaller values of n (7, 8, 9), the duration t is insignificant, but it increases with increasing n. For example, for n = 7, the duration varies from 0,002 to 0,018 seconds, whereas for n = 15, it ranges from 5,081 to 211,017 seconds. Judging from this, it can be concluded that increasing the size of the n matrix P leads to an increase in the time complexity of the algorithm.

Table 2 also illustrates that the number of iterations i increases with the increase in n. For example, for n = 7, the number of iterations required to generate matrix P ranges from 13 to 356, while for n = 15, it ranges from 32,724 to 1,354,150. This is explained by the fact that as the value of n increases, the complexity of searching for a valid permutation matrix P also increases, requiring more iterations to find the optimal solution. Consequently, this leads to an increase in the computational complexity of the encryption algorithm.

To analyze Table 2, let us calculate the average value of the number of iterations for each value of *n*.

For n = 7:
average number of iterations: 2222/15 ≈ 148,133.
For n = 8:
average number of iterations: 6281/15 ≈ 418,733.
For n = 9:
average number of iterations: 14730/15 ≈ 982,0.
For n = 10:
average number of iterations: 65871/15 ≈ 4391,4.
For n = 11:
average number of iterations: 154728/15 ≈ 10325,2.
For n = 12:
average number of iterations: 253148/15 ≈ 16876,53.
For n = 14:
average number of iterations: 2060846/15 ≈ ≈ 137389,7.
For n = 15:
average number of iterations: 4486117/15 ≈ ≈ 299074,467.

Based on the data in Table 2, a graph (Fig.4) shows the average number of iterations to form the matrix *P* at different values of n has been constructed.

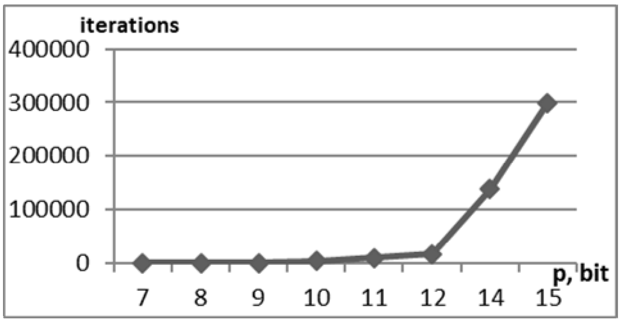

Fig. 4. Average number of iterations form matrix P at different values of n is shown below.

From the graph in Fig. 2, it can be observed that increasing the value of *n* leads to a significant increase in the number of iterations required for generating the permutation matrix *P*. This can pose significant challenges in computational tasks, especially at large values of *n*, and may require substantial computational resources for efficient matrix generation.

The increase in time and computational complexity with the increase in the size of the codeword leads to higher hardware complexity, necessitating greater computational power and resources for the generation process.

The analysis carried reveals that the operation of forming matrix *P* is characterized by the highest temporal and computational complexity between all stages of the algorithm. This is due to additional requirements for the matrix, including having only one unit in each row, and all other elements being zeros.

The formation of such a matrix requires random or pseudo-random selection of combinations and permutations, which can lead to significant computational overhead. Therefore, optimizing this stage of the algorithm can improve the overall speed of the encryption or transmission system.

V. IMPROVEMENT OF THE ENCRYPTION METHOD

Replacing the operation of forming the permutation matrix, which requires the most resources, with the operation of modulo and computing the inverse matrix of the scrambler

$$S^{-1}(\bmod p), \tag{1}$$

allows for ensuring high cryptographic security of the encryption algorithm. This approach complicates the decryption process for potential attackers.

The following is the scheme of the enhanced operation of the cryptosystem as illustrated in Fig.5.

Replacement of the operation of forming the permutation matrix P with the operation of modulo *p* brings several advantages. Computing (1) is a less resource-intensive process compared to generating the permutation matrix *P*, leading to improved encryption efficiency.

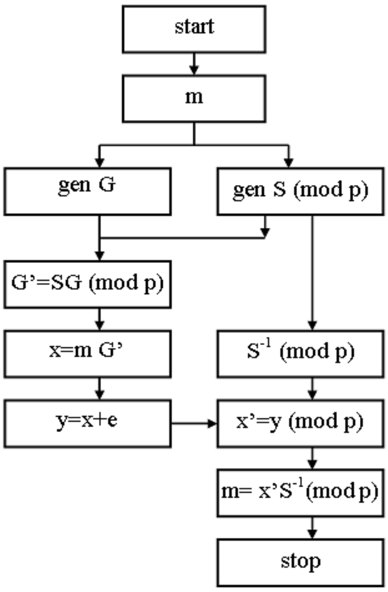

Fig. 5. The scheme of the proposed encryption method

The utilization of modular operations and computing inverse matrices modulo *p* is based on the properties of algebraic structures, ensuring a high level of mathematical security in the encryption system.

Modular operations can be applied across a wide range of cryptographic algorithms and encryption systems, providing versatility and universality.

Example: Let the original message have a length of *k* = 4 information bits

$m = [10, 5, 7, 2]$.

To transmit it, we will use a Hamming code (7, 4). For this, it is necessary to add $r = 3$ parity bits to each four-bit block, as this satisfies the condition

$$2^r \geq k + r + 1, \tag{2}$$

to ensure detection and correction of errors in the message $2^4 \geq 4 + 3 + 1$.

Therefore, the original message *m* will be transformed into a codeword *x* with a length of 7 bits.

At the initialization stage, we generate matrices G and S using a number system with a base of 13. As a result, we obtain the following:

$$G=\begin{pmatrix}1&1&1&0&0&0&0\\1&0&0&1&1&0&0\\0&1&0&1&0&1&0\\1&1&0&1&0&0&1\end{pmatrix},$$

$$S=\begin{pmatrix}2&0&4&0\\9&5&10&3\\3&8&9&11\\0&1&2&1\end{pmatrix}.$$

To form the public key, we need to compute

$$G'=SG(\mathrm{mod}\ p) \qquad (3).$$

As a result (3), we obtain the following:

$$G'=\begin{pmatrix}2&6&2&4&0&4&0\\4&9&9&5&5&10&3\\9&10&3&2&8&9&11\\2&3&0&4&1&2&1\end{pmatrix}.$$

To create the private key, we will compute (1), and we will obtain:

$$S^{-1}=\begin{pmatrix}10&3&2&8\\10&3&6&3\\5&5&12&9\\6&0&9&6\end{pmatrix}.$$

During the encryption stage, after the encryption

$$\mathrm{x}=\mathrm{mG}', \qquad (4)$$

we obtain the codeword

x = [107, 181, 86, 87, 83, 157, 94].

After adding the error vector e = [0, 0, 0, 0, 0, 0, 8] by $\mathrm{y}=\mathrm{x}+\mathrm{e}$, we get:

y = [107, 181, 86, 87, 83, 157, 101].

During the decryption stage, we perform the operation $y(\mathrm{mod}p)$ and obtain: x' = [3, 12, 8, 9, 5, 1, 10].

The determination of the position a_i and the error size v is performed by computing the checksums, resulting in $a_i = 7$ and $v = 5$.

After correction of error, we obtain the following

x' = [3, 12, 8, 9, 5, 1, 3].

By performing

$$m=x'S^{-1}(\mathrm{mod}\ p), \qquad (5)$$

we obtain the original message.

VI. THE CRYPTOGRAPHIC RESILIENCE OF THE PROPOSED METHOD

The number of possible codewords during data transmission in the McEliece cryptosystem depends on the number of information bits and can be calculated using the formula p^k, where k is the number of information bits, and p is the base of the numeral system. The scrambler matrix S and the permutation matrix P are used to enhance the resilience of the code. They affect the number of possible codewords by changing the structure and properties of the codeword.

The total number of scrambler matrices S with size $k\times k$, where k is the number of information bits in the original message, is $p^{k\times k}$, since each element of the matrix can take any possible value from 0 to $p-1$.

The size of the permutation matrix P of size $n\times n$, where n is the number of bits in the codeword, has a specific structure where there is only one unity in each row of the permutation matrix and the rest of the elements are zeros. This condition, allows for an efficient permutation of the code word bits. This is particularly useful in cryptographic applications where ensuring the security of information transmission is crucial.

Such a structure enables the permutation of code word bits to be performed easily without loss of information. Each unity in a row of the matrix indicates the position at which the corresponding position in the output vector will be moved. Other positions containing zeros remain unchanged.

This condition is essential for ensuring the proper functioning of encryption and decryption algorithms, such as the McEliece cryptosystem, where the permutation matrix is used to shuffle bits before transmission and to restore them on the receiver's side. To find the maximum possible number of such permutation matrices, we need to consider the number of ways we can arrange the 1s in each row so such that only one 1 appears, and the rest are 0.

For the first row, there are n options to place 1. Once the position of the 1 in the first row is determined, there are *(n-1)* options left for placing the 1 in the second row (since it cannot occupy the same column as the 1 in the first row). Continuing this pattern, the number of options placing of the 1 in each subsequent row decreases by 1.

Therefore, the total number of such permutation matrices is the product of the number of options for each row, which can be expressed as:

$$n\cdot(n-1)\cdot(n-2)\cdot\ldots\cdot 1. \qquad (6)$$

Expression (6) is equivalent to n!

The total number of possible combinations of matrices S and P, without considering the condition of their invertibility, is

$$p^{k\times k}\cdot n!. \qquad (7)$$

Comparison of implementation options for the (7,4) code:

- Base 2 numbering system:

The number of possible combinations for the scrambler matrix $S=2^{4\times 4}=2^{16}=65536$, as each element of the matrix can be 0 or 1, and the scrambler matrix has a size of 4×4.

The number of possible combinations for the permutation matrix $\mathrm{P} = 7! = 5040$, since for the 7×7 permutation matrix, each row can contain only one 1, and this 1 can be in any of the 7 positions.

- Base q numbering system:

The number of possible combinations for the scrambler matrix $S = q^{4\times4} = q^{16}$, as each element of the matrix can take values from 0 to $q-1$, and the scrambler matrix has a size of 4×4.

To achieve a greater number of possible combinations than (7), it is necessary to choose a value for q that satisfies the condition

$$q^{k\times k} > p^{k\times k} n! . \tag{8}$$

For example, for the (7,4) code, where $p = 2$ (for the binary numbering system), the size of the scrambler matrix $k = 4$, and the size of the permutation matrix $n = 7$. Therefore, after substituting the known values, we get the following inequality: $q^{4\times4} > 2^{4\times4} \cdot 7!$

Now we need to find the value of q that satisfies this inequality:

$q^{16} > 65536 \cdot 5040$; $q^{16} > 4{,}056088$.

Therefore, to achieve a greater number of possible combinations in the proposed implementation of the cryptosystem compared to the classical McEliece cryptosystem, we need to choose a value of $q = 5$.

VII. CONCLUSION

The paper proposes an enhancement to the McEliece asymmetric cryptosystem by replacing the permutation matrix with modulo operation and using a finite field GF(q). It is demonstrated that achieving cryptographic security analogous to the McEliece cryptosystem requires using $q \geq 5$.

This enhancement aims to improve the efficiency of public key generation operations and increase cryptographic security by expanding the number of possible combinations of codeword elements. The use of modular arithmetic extends the space of possible combinations, making attacks more difficult and requiring greater computational effort.

The proposed approach combines technical efficiency with a high level of security, making it potentially promising for practical applications in various fields of information security, particularly for data protection in cyber-physical and embedded systems.

References

[1] Dam, D. T., Tran, T. H., Hoang, V. P., Pham, C. K., & Hoang, T. T. (2023). A survey of post-quantum cryptography: Start of a new race. *Cryptography*, *7*(3), 40. DOI: 10.3390/cryptography 7030040

[2]Kichna, A., & Farchane, A. (2023, May). Secure and efficient code-based cryptography for multi-party computation and digital signatures. In *Computer Sciences & Mathematics Forum* (Vol. 6, No. 1, p. 1). MDPI. DOI:10.3390/cmsf2023006001

[3]Esser, A., May, A., & Zweydinger, F. (2022, May). McEliece needs a break–solving McEliece-1284 and quasi-cyclic-2918 with modern ISD. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques* (pp. 433-457). Cham: Springer International Publishing. DOI:10.1007/978-3-031-07082-2 16

[4]Yevseiev, S., Korol, O., Pohasii, S., & Khvostenko, V. (2021, September). Evaluation of cryptographic strength and energy intensity of design of modified crypto-code structure of McEliece with modified Elliptic codes. III International Scientific and Practical Conference "Information Security And Information Technologies", Odesa, Ukraine, September 13–19, 2021, Vol-3200, 2021, 144-157. ISSN 1613-0073. https://ceur-ws.org/Vol-3200/paper20.pdf.

[5]Parashar, A., & Jadiya, D. Enhanced McEliece Algorithm for post-quantum cryptosystems. DOI:10.13140/RG.2.2.22002.93125.

[6]Bindal, E., & Singh, A. K. (2024). Secure and compact: A new variant of McEliece Cryptosystem. *IEEE Access*. DOI: 10.1109/ACCESS.2024.3373314.

[7]Yevseiev, S., Korol, O., & GavrilovA, A. (2019). Development of authentication codes of messages on the basis of UMAC with crypto-code McEliece's scheme. *International Journal of 3D printing technologies and digital industry*, *3*(2), 153-170.

[8]McEliece, R. J. (1978). A public-key cryptosystem based on algebraic. *Coding Thv*, *4244*, 114-116.

[9]Isakov, D. A., & Sokolov, A. V. (2022). McEliece cryptosystem based on quaternary hamming codes. *Informatics & Mathematical Methods in Simulation*, *12*(4). 280-287. DOI: 10.15276/imms.v12.no4.280

[10] Freudenberger, J., & Thiers, J. P. (2021). A new class of q-ary codes for the McEliece cryptosystem. *Cryptography*, *5*(1), 11. DOI: 10.3390/cryptography5010011.

[11] Ukwuoma H., Gabriel A., Thompson A., Boniface A. (2022) Post-quantum cryptography-driven security framework for cloud computing. *Open Computer Science 12(1).* 142-153. DOI:10.1515/comp-2022-0235.

[12] Kabeya T. (2019) McEliece's Crypto System based on the Hamming Cyclic Codes. *International Journal of Innovative Science and Research Technology*, *4(7).* 293-296.

Alina Ya. Davletova was born in Ternopil, Ukraine, in 1981. In 2016, she completed the full course at Ternopil National Economic University and obtained a specialist degree in the field of System Software, being qualified as a Specialist in Software Systems.

She is currently pursuing a Ph.D. degree in cybersecurity at Western Ukrainian National University.Since 2021, she has been a lecturer at the Department of Cybersecurity.

She is a co-author of 3 collective monographs and has published over 80 articles and holds 17 patents. Her research interests include optimizing computational processes in specialized devices, information theory and encoding.

Vasyl V. Yatskiv was born in Ivano-Frankivsk region, Ukraine, in 1972. He received a specialist degree in Process Automation at Ivano-Frankivsk Technical University of Oil and Gas, Ukraine, 1996. Doctor of Philosophy in Computers, Systems and Networks, Lviv Polytechnic University, Ukraine, 2001. Associated professor, approved by the Ministry of Education and Science of Ukraine, 2004. Doctor of Engineering Science degree according to specialty Computer Systems and Components, Lviv Polytechnic National University in 2016. He is an author of more than 130 scientific articles and inventions. His research interests include secure data storage systems, code-based encryption methods, corrective codes in residue number system.

Stepan V. Ivasiev was born in Ternopil, Ukraine, in 1986. He obtained his B.S. and M.S. degrees in Software for Automated Systems at Ternopil National Economic University. In 2009, he defended his thesis for the degree of Candidate of Technical Sciences in the field of Computer Systems and Components and obtained a Ph.D. in Ternopil National Economic University. He is a co-author of 4 monographs and has published over 100 articles. His research interests include residue class systems, number theory, encoding, and factorization.

Mykola Karpinskyj, Prof., DSc. Professor at University of the National Education Commission, Krakow, Poland. His research interests include issues of improving the efficiency and safety of wireless sensory networks; construction of cryptographic means of information protection.