2024. Volume 8, Number 2 *Pages 101-104*

UOT: 004.056
DOI: https://doi.org/10.30546/09090.2025.01.2015

INVESTIGATION OF THE SECURITY OF A KEY EXCHANGE PROTOCOL BASED ON MATRIX ALGEBRA

VAGIF GASIMOV[1], JABIR MAMMADOV[2], NARGIZ MAMMADZADA[3], NIHAT MAMMADLI[4]

[1,2,3]Baku Engineering University, Khirdalan,

[4]ADA University, Baku

[1]vaqasimov@beu.edu.az, [2]camammadov@beu.edu.az, [3]mammadzada.nargiz@ gmail.com [4]nihat.memmedli.200@gmail.com

ARTICLE INFO

Article history:
Received: 2025-03-12
Received in revised form:2025-03-18
Accepted: 2025-04-04
Available online

Keywords:
Matrix,
key,
security analysis

ABSTRACT

This paper considers the issue of assessing the security of a key exchange protocol based on matrix algebra. To protect the confidentiality of information, it is important that the keys are exchanged securely. This can be done by using methods such as encrypted key transmission, creation of dedicated secure channels, formation of a common key between parties without exchanging secret information using public key algorithms, etc. The paper analyzes the security of the protocol implemented using the approach based on non-invertible matrices against linear algebra and brute force attacks, shows that the use of non-invertible matrices increases the security of the system, and also evaluates the consumption of computing resources. The results confirm that this approach can be used as a secure and practical method of key exchange.

1. INTRODUCTION

The main problem of the symmetric encryption method, which is one of the most common methods of protecting the confidentiality of information, is the reliable delivery of the secret key to the parties. To solve this problem, various methods are used, such as sending the key in encrypted form, creating separate protected channels, delivering the key by courier, forming a common key between the parties without exchanging secret information using public-key algorithms, etc. The protocol created based on the Diffie-Hellman algorithm is a clear example of the public-key encryption method [1-2]. This algorithm allows two parties (sender and receiver) to create a common secret key using an open channel. The basis of the Diffie-Hellman key exchange protocol is one-way functions, that is, functions that are simple to calculate in one direction, but require relatively large resources to calculate in the opposite direction. The secret key created based on the key exchange protocol can later be used to encrypt data.

Another interesting method for generating a shared secret key is based on matrix algebra. In general, the essence of this method is based on building a public-key exchange process based on matrices whose inverse cannot be calculated. It is known that the product of two matrices can be calculated only if the number of columns in the first matrix is equal to the number of rows in the second matrix. In the special case, if both matrices are square matrices of the same size, then their product can always be found.

When matrices A and X satisfying these conditions are multiplied together, the matrix C of the corresponding size is obtained:

$$A \cdot X = C \tag{1}$$

If C and A are known, the matrix X can be calculated based on the following expression:

$$X = C \cdot A^{-1} \tag{2}$$

Here, the matrix A^{-1} is the inverse matrix of the matrix A. If we denote the determinant of the matrix A by *detA*, and the minor by A_{nm} (i,j=1,2,...,n), the following expression is used to find the inverse matrix:

$$A^{-1} = \frac{1}{\det A}\begin{pmatrix} A_{11} & A_{21} & \cdots & A_{n1} \\ A_{12} & A_{22} & \cdots & A_{n2} \\ \cdots & \cdots & \cdots & \cdots \\ A_{1n} & A_{2n} & \cdots & A_{nn} \end{pmatrix} \tag{3}$$

As can be seen, if *det*A=0, it is necessary to perform the division by zero operation, which leads to uncertainty, that is, it is impossible to calculate the inverse of the matrix. In other words, although it is always possible to find the product of two matrices when one of the factors and the product are given, in some cases it is impossible to calculate the other factor. This is because if the known factor is a matrix that does not have an inverse, then for the other factor there are infinitely many solutions that satisfy equation (1). It is this property of matrices that allows us to create a one-way function with their help [3-4]. In the studies conducted by the authors in the literature [2], a key exchange protocol based on non-invertible matrices was proposed. In this paper, the issue of assessing the security and efficiency of this protocol is considered.

The essence of the key exchange protocol based on non-invertible matrices given in [2] is as follows:

1) the parties (Parties 1 and 2) who want to exchange information choose a common square matrix C with determinant equal to 0;

2) The first party chooses an nxn matrix A and a coefficient q, and the second party chooses a matrix B and a coefficient p. The first party calculates the product $S_{A1}= A \cdot C^q$ and sends it to the second party, and the second party calculates the product $S_{B1}= C^p \cdot B$ in a similar manner to the first party.

3) The first party chooses an nxn matrix M and multiplies the product $M \cdot C^q$ by the matrix S_{B1} obtained from the second party from the left, and sends the resulting matrix $S_{A2}= M \cdot C^q \cdot C^p \cdot B$ to the second party. The second party selects an nxn matrix N and sends the matrix $S_{B2}= A \cdot C^q \cdot C^p \cdot N$ obtained by multiplying the product $C^p \cdot N$ by the matrix S_{A1} from the right to the first party.

4) The first party multiplies the product $M \cdot A^{-1}$ from the second party to the matrix S_{B2} from the left, and the second party multiplies the product $B^{-1} \cdot N$ from the first party to the matrix S_{A2} from the right. Both parties get the same result: $S= M \cdot C^q \cdot C^p \cdot N$. The resulting matrix is used as a secret key that the parties jointly form.

2. LINEAR ALGEBRA-BASED SECURITY ANALYSIS

Since the intermediate keys, S_{A1} and S_{A2}, are obtained from the product of the matrix C by another matrix, they must be non-invertible matrices (the product of non-invertible and inverse matrices is equal to the non-invertible matrix). Therefore, it is not possible to calculate the matrix

$M \cdot C^q$ from the equation $S_{A2} = M \cdot C^q \cdot S_{B1}$. Because the number of matrices $M \cdot C^q$ satisfying this equation is greater than one. If the matrices M and N are not used in the protocol, the equation $S_{A2} = M \cdot C^q \cdot C^p \cdot B$ takes the form $S_{A2} = C^q \cdot S_{B1}$. Then the number q can be calculated by checking all possible values until the equation $S_{A2} = C^q \cdot S_{B1}$ is satisfied. The number p can be calculated in the same way. However, when the matrix M is included in the equation, it is necessary to find not only the number q, but also the matrix M. This increases both the number of cases to be checked and the number of solutions to the equation $S_{A2} = M \cdot C^q \cdot C^p \cdot B$. Thus, it becomes extremely difficult to verify the correctness of the solution found. By the same logic, the importance of including the matrix N can be clearly shown.

3. KEY FİELD ANALYSİS

The brute force resistance of the proposed method directly depends on the number of non-inverse matrices that can be used. To calculate this number, let us assume that the elements of the matrix Z with any inverse take values from 0 to 999. Then the first column of the matrix Z can take $(1000^n - 1)$ different values (not every element in the first column can be equal to 0 at the same time). The second column must be outside the one-dimensional space formed by the first column. That is, if we denote the first column by a_1, then the second column must be different from a_1, $2 * a_1$, $3 * a_1$, ... , $1000 * a_1$. Therefore, a_2 can take $(1000^n - 1000)$ values. The third column must be outside the space formed by the vectors a_1 and a_2. That is, it can take $(1000^n - 1000^2)$ values. Continuing in a similar way, we see that the value of the vector a^i is $(1000^n - 1000^i)$. Taking these into account, we can see that the number of possible values of the matrix Z is $(1000^n - 1) * (1000^n - 1000) * (1000^n - 1000^2) * ... * (1000^n - 1000^{n-1})$. If the elements of the matrix Z can take the value "m", then the number of such matrices is $(m^n - 1) * (m^n - m) * (m^n - m^2) * ... * (m^n - m^{n-1})$. The number of non-invertible matrices is equal to the difference between the number of all matrices and the number of invertible matrices. Given that the number of all matrices is m^{n*n}, the number of non-invertible matrices is equal to $m^{n*n} - (m^n - 1) * (m^n - m) * (m^n - m^2) * ... * (m^n - m^{n-1})$. Table 1 shows the number of non-invertible matrices depending on the size of the matrix and the largest value of its elements. The results prove that the algorithm is quite strong against brute force cracking. For example, for matrices of size 4x4 and 5x5 with 4-digit elements, the numbers 10^{60} and 10^{96} were obtained, which correspond to 200 and 319-bit numbers when represented in binary code. As can be seen, the key area of the algorithm is quite large, and if necessary, this area can be further increased by changing the size of the matrix and the range of values that its elements take.

Table 1. The number of non-invertible matrices based on the matrix size and the maximum value of the element

Matrix size	Maximum value of the elements of the matrix			
	99	999	9999	99999
2	$1.009 * 10^6$	10^9	10^{12}	10^{15}
3	$1.009 * 10^{16}$	10^{24}	10^{32}	10^{40}
4	$1.009 * 10^{30}$	10^{45}	10^{60}	10^{75}
5	$1.009 * 10^{48}$	10^{72}	10^{96}	10^{120}

4. EVALUATION OF THE RESOURCES USED

One of the main parameters evaluated is the determination of the time spent on key generation. As can be seen from the algorithm, a very large part of the resource is spent on calculating the product of matrices. The size of the matrices and the maximum value that the

elements can take significantly affect the time spent on calculating the common key. Naturally, the characteristics of the computer on which the program is executed also play a major role here. The results obtained by implementing the program designed for the algorithm under consideration on a computer with the parameters *CPU: Intel® Core™ i5-8265U CPU @ 1.60GHz, RAM: 8 GB* are given in Table 2. As can be seen, the results are quite high for such a process and allow calculating a separate key for each information exchange session.

Table 2. Time spent on key formation (*in seconds*)

Matrix size	Maximum value of the elements of the matrix				
	100	1000	10000	100000	1000000
3	0.002627849	0.002312302	0.002815985	0.002208733	0.002175283
4	0.006254410	0.006436371	0.006414818	0.006233954	0.006390404
5	0.056668591	0.054777193	0.059532523	0.058203577	0.057142686
6	0.114728498	0.118162989	0.119587588	0.120708131	0.124338245
7	0.230506992	0.233087420	0.230650210	0.231170415	0.254934334

5. CONCLUSION

The article evaluates the security of the protocol for generating a shared secret key based on non-invertible matrices. The evaluation is based on linear algebra and key space analyses. In the presented article, the time spent on implementing the protocol based on the application of non-invertible matrices is also evaluated. The analyses have shown that the calculation of the shared secret key by the considered method has high resistance to linear algebra and brute force attacks. The proposed approach, in addition to providing a high level of security for key exchange, has practical efficiency. In future work, it is planned to investigate and optimize wider application areas of the protocol.

REFERENCES

1. Gasimov V.A. Fundamentals of information security. Textbook, Baku, 2009, 340 p.
2. Gasimov V., Mammadzada N. and Mammadov J., "New Key Exchange Protocol Based on Matrix Algebras", *2023 5th International Conference on Problems of Cybernetics and Informatics (PCI)*, Baku, Azerbaijan, 2023, pp. 1-3, doi: 10.1109/PCI60110.2023.10326004.
3. Rahman, Nael and Shpilrain, Vladimir. "MAKE: A matrix action key exchange" *Journal of Mathematical Cryptology*, vol. 16, no. 1, 2022, pp. 64-72. https://doi.org/10.1515/jmc-2020-0053
4. Daniel Brown, Neal Koblitz & Jason LeGrow. (2021) Cryptanalysis of `MAKE', Cryptology ePrint Archive, Paper 2021/465, https://eprint.iacr.org/2021/465