computing@computingonline.net
www.computingonline.net

Print ISSN 1727-6209
On-line ISSN 2312-5381
International Journal of Computing

POLYNOMIAL-TIME PLAINTEXT-RECOVERY ATTACK ON THE MATRIX-BASED KNAPSACK CIPHER

Aleksei Vambol

National Aerospace University "KhAI", 17 Chkalov st., Kharkiv, Ukraine, 61070,
o.vambol@csn.khai.edu

Paper history:
Received 18 May 2020
Received in revised form 29 August 2020
Accepted 04 September 2020
Available online 27 September 2020

Keywords:
matrix-based knapsack cipher;
plaintext-recovery attack;
cryptanalysis;
time complexity;
computational complexity;
asymmetric encryption;
homomorphic encryption;
knapsack cryptosystems;
group-based knapsack ciphers.

Abstract: The aim of the present paper is to propose a polynomial-time plaintext-recovery attack on the matrix-based knapsack cipher. The aforesaid algorithm uses only public information and has time complexity $O(t^{1.34})$, where t is the decryption time of the attacked cryptosystem. The matrix-based knapsack cipher is a novel additively homomorphic asymmetric encryption scheme, which is a representative of group-based knapsack ciphers. This cryptosystem is based on the isomorphic transformation's properties of the inner direct product of diagonal subgroups of a general linear group over a Galois field. Unlike the classical knapsack cryptoschemes, the cryptographic strength of the aforesaid cipher depends on the computational complexity of the multidimensional discrete logarithm problem. Due to the attack proposed in the given paper, the matrix-based knapsack cipher can be considered broken and should not be used as a privacy tool. However, this cryptosystem is still suitable for educational purposes as an example of the application of linear and abstract algebras in asymmetric cryptography.

1. INTRODUCTION

Asymmetric encryption schemes are widely used to ensure the confidentiality of communication via insecure channels. These cryptosystems allow the interacting parties to create a shared secret key for a symmetric cipher in such a way that an eavesdropper gets no information useful for cryptanalysis [1, 2]. Network protocols that use asymmetric encryption include TLS [3], S/MIME [4], OpenPGP [5], Tor [6] and many others [7].

Some of asymmetric ciphers are homomorphic meaning that they allow calculations on encrypted data to be performed without preliminary decryption. This property makes it possible to use the given cryptosystems in several areas of applications besides symmetric key establishment. In particular, homomorphic asymmetric ciphers are used in secret e-voting protocols [8] and cloud computing [9].

The matrix-based knapsack cipher is a novel additively homomorphic asymmetric encryption scheme, which is a representative of group-based knapsack ciphers [10]. This cryptosystem is based on the isomorphic transformation properties of the inner direct product of diagonal subgroups of a general linear group over a Galois field [11]. Unlike the classical knapsack cryptoschemes, the cryptographic strength of the aforesaid cipher depends on the computational complexity of the multidimensional discrete logarithm problem [10].

The given cipher was originally proposed in [11]. The approach to building this cryptosystem over a Galois field with a multiplicative group of a large smooth order was proposed in [12]. Another approach, in which the aforesaid cipher is built over a small Galois field, was used in [10], where the property of additive homomorphism was proven for this cryptoscheme. Also, in [10] a secret e-voting protocol based on the given cipher was briefly described.

The aim of the present paper is to propose a plaintext-recovery attack on the matrix-based

This paper has been submitted fir the Open Special Issue on Green Mobile Computing and IoT Systems. Assessment, Modeling, Assurance.

knapsack cipher. This algorithm uses only public information and has computational complexity polynomial in the time required for decryption by the attacked cryptoscheme.

2. MATRIX-BASED KNAPSACK CIPHER

The given cryptosystem has two parameters [10]:

1. The order of the finite field, over which the cipher is built. The given parameter is designated as q. It is necessary that q - 1 be small (or just smooth in the case of the approach using a large Galois field) and larger than 1.

2. The order of the square matrices being used. It is denoted as n. The minimum value of *n* is 2.

The key generation procedure begins with choosing the generating set of the abelian group G, which is the diagonal subgroup of the general linear group GL(n, GF(q)). This set is represented by the tuple $(g_1, g_2, ..., g_n)$ obtained from $(z_1, z_2, ..., z_n)$, where z_i is a randomly chosen primitive element of GF(q). The element g_i is obtained from the n-dimensional identity matrix over GF(q) by means of replacing the (i, i) entry with z_i [10]. Since the order of each g_i is equal to q - 1, each $d \in G$ has a single representation in the following form [10]:

$$d = g_1^{p_1} \cdot g_2^{p_2} \cdot ... \cdot g_n^{p_n}, \quad (1)$$

where p_i is a nonnegative integer less than q - 1. Therefore, it is not hard to see the correctness of the formula

$$ent_i(d) = z_i^{p_i}, \quad (2)$$

in which $ent_i(d)$ is the (i, i) entry of d.

The private key is a randomly selected matrix $s \in GL(n, GF(q))$. This matrix is used to define the group H, which is a subgroup of GL(n, GF(q)), the isomorphism $f: G \to H$ and its inverse $f^{-1}: H \to G$. This pair of isomorphisms can be described as follows [10]:

$$f: \delta \to s^{-1} \cdot \delta \cdot s,$$
$$f^{-1}: \mu \to s \cdot \mu \cdot s^{-1}.$$

The public key is a tuple $(e_1, e_2, ..., e_n)$. Its elements are calculated by the formula

$$e_i = f(g_{\sigma_i}), \quad (3)$$

where $(\sigma_1, \sigma_2, ..., \sigma_n)$ is a random permutation of (1, 2, ..., n). Although the original version of the considered cipher does not use the aforementioned secret permutation [10, 11], this feature should be introduced to complicate a cryptanalytic attack on the given cryptosystem.

The encryption procedure converts a plaintext into an integer tuple $(x_1, x_2, ..., x_n)$, for which $0 \le x_i \le q - 2$, and computes the ciphertext c in the following way [10]:

$$c = e_1^{x_1} \cdot e_2^{x_2} \cdot ... \cdot e_n^{x_n}. \quad (4)$$

Since $(g_1, g_2, ..., g_n)$ is a generating set of G, the encryption procedure and (3) imply that each element of H belongs to the set of ciphertexts. Thus, there is a bijection between plaintexts and elements of H.

Decryption is performed as follows:

1. The tuple $(y_1, y_2, ..., y_n)$, where y_i is the (i, i) entry of $f^{-1}(c)$, is computed. By virtue of (1)-(4), y_{σ_i} equals z_{σ_i} to the power of x_i.

2. The tuples $(z_1, z_2, ..., z_n)$ and $(\sigma_1, \sigma_2, ..., \sigma_n)$ are found using the following condition. If the (k, k) entry of $f^{-1}(e_i)$ is not equal to 1, then σ_i is k and z_k equals this entry. This approach follows from (3) and the definition of g_i. The given step can be avoided by storing the aforesaid tuples along with the private key.

3. The plaintext tuple $(x_1, x_2, ..., x_n)$ is restored by the formula

$$x_i = \mathrm{dlog}_{z_{\sigma_i}}(y_{\sigma_i}), \quad (5)$$

where $\mathrm{dlog}_\beta(\alpha)$ is the discrete logarithm of α base β. Since q - 1 is small (or at least smooth), this step can be performed efficiently.

The given cipher is additively homomorphic due to the following properties [10]:

1. The plaintexts set is an additive abelian group under the operation ⊕, which is defined as follows:

$$(u_1, ..., u_n) \oplus (v_1, ..., v_n) =$$
$$= ((u_1 + v_1) \bmod (q-1), ..., (u_n + v_n) \bmod (q-1)),$$

where $(u_1, u_2, ..., u_n)$ and $(v_1, v_2, ..., v_n)$ are plaintext tuples. Thus, the plaintext group is s an additive group of the n-dimensional module over the residue ring modulo *n*.

2. The ciphertext set equipped with the matrix product operation is the multiplicative abelian group H mentioned above.

3. If c_i denotes the ciphertext obtained from the plaintext tuple m_i by encryption performed using some fixed public key, then decryption of the ciphertext $c_1 \cdot c_2 \cdot ... \cdot c_k$ with the corresponding private key produces $m_1 \oplus m_2 \oplus ... \oplus m_k$.

These properties, together with the bijection between elements of H and plaintexts, make the ciphertext group isomorphic to the plaintext one.

The following toy example of this cryptosystem, where q = 13 and n = 4, aims at demonstrating its property of additive homomorphism. Due to the aforesaid values of the parameters, the group G is a diagonal subgroup of GL(4, GF(13)). The tuple (z_1, z_2, z_3, z_4) is selected as (2, 6, 7, 11), therefore (g_1, g_2, g_3, g_4) is described as follows:

$$g_1 = \begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, g_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

$$g_3 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 7 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, g_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 11 \end{pmatrix}.$$

Since each g_i is of order 12, a plaintext tuple must contain only integers lying in the interval [0 .. 11].

The private key matrix s and its inverse s^{-1} are chosen in the following way:

$$s = \begin{pmatrix} 4 & 0 & 2 & 10 \\ 3 & 10 & 0 & 11 \\ 10 & 12 & 9 & 8 \\ 11 & 4 & 6 & 10 \end{pmatrix}, s^{-1} = \begin{pmatrix} 6 & 4 & 12 & 6 \\ 5 & 11 & 1 & 12 \\ 7 & 8 & 4 & 7 \\ 8 & 2 & 10 & 4 \end{pmatrix}.$$

The tuple $(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$ used in this example is (2, 4, 3, 1). Therefore, in accordance with (3), the public key (e_1, e_2, e_3, e_4) is specified as follows:

$$e_1 = \begin{pmatrix} 9 & 5 & 0 & 12 \\ 9 & 5 & 0 & 7 \\ 3 & 10 & 1 & 11 \\ 4 & 9 & 0 & 7 \end{pmatrix}, e_2 = \begin{pmatrix} 11 & 6 & 9 & 2 \\ 7 & 0 & 5 & 4 \\ 3 & 7 & 5 & 11 \\ 11 & 4 & 6 & 11 \end{pmatrix},$$

$$e_3 = \begin{pmatrix} 6 & 6 & 11 & 4 \\ 8 & 8 & 2 & 9 \\ 6 & 2 & 9 & 10 \\ 2 & 5 & 7 & 0 \end{pmatrix}, e_4 = \begin{pmatrix} 12 & 0 & 12 & 8 \\ 7 & 1 & 10 & 11 \\ 2 & 0 & 2 & 5 \\ 6 & 0 & 3 & 3 \end{pmatrix}. \quad (6)$$

The plaintexts tuples m_1 = (3, 8, 1, 5) and m_2 = (9, 7, 4, 11) are chosen for encryption. The corresponding pair of ciphertexts c_1 and c_2 is obtained as shown below:

$$c_1 = e_1^3 \cdot e_2^8 \cdot e_3^1 \cdot e_4^5 = \begin{pmatrix} 10 & 10 & 8 & 0 \\ 4 & 5 & 4 & 12 \\ 7 & 6 & 12 & 2 \\ 10 & 0 & 6 & 3 \end{pmatrix},$$

$$c_2 = e_1^9 \cdot e_2^7 \cdot e_3^4 \cdot e_4^{11} = \begin{pmatrix} 10 & 10 & 10 & 12 \\ 9 & 0 & 9 & 6 \\ 11 & 4 & 12 & 6 \\ 7 & 3 & 8 & 1 \end{pmatrix}.$$

The ciphertext c_p selected for decryption is defined in the following way:

$$c_p = c_1 \cdot c_2 = \begin{pmatrix} 5 & 2 & 0 & 7 \\ 5 & 1 & 8 & 10 \\ 10 & 7 & 11 & 12 \\ 5 & 3 & 1 & 3 \end{pmatrix}. \quad (7)$$

The decryption procedure starts by computing the value of $f^{-1}(c_p)$. Inasmuch as

$$f^{-1}(c_p) = \begin{pmatrix} 3 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 11 & 0 \\ 0 & 0 & 0 & 5 \end{pmatrix},$$

the first step of the decryption establishes the identity (y_1, y_2, y_3, y_4) = (3, 1, 11, 5).

The optional next step begins with calculating the value of $f^{-1}(e_1)$. The second element of the main diagonal of $f^{-1}(e_1)$ is 6 and other ones are equal to 1, so $\sigma_1 = 2$ and $z_2 = 6$. The same approach applied to e_2, e_3 and e_4 is used to determine that $(\sigma_1, \sigma_2, \sigma_3, \sigma_4)$ is (2, 4, 3, 1) and (z_1, z_2, z_3, z_4) equals (2, 6, 7, 11).

The final step of the decryption lies in computing the elements of the plaintext tuple (x_1, x_2, x_3, x_4) in the following way:

$$x_1 = \operatorname{dlog}_{z_{\sigma_1}}(y_{\sigma_1}) = \operatorname{dlog}_6(1) = 0,$$
$$x_2 = \operatorname{dlog}_{z_{\sigma_2}}(y_{\sigma_2}) = \operatorname{dlog}_{11}(5) = 3,$$
$$x_3 = \operatorname{dlog}_{z_{\sigma_3}}(y_{\sigma_3}) = \operatorname{dlog}_7(11) = 5,$$
$$x_4 = \operatorname{dlog}_{z_{\sigma_4}}(y_{\sigma_4}) = \operatorname{dlog}_2(3) = 4.$$

Thus, the decryption of c_p defined as $c_1 \cdot c_2$ yields the plaintext (0, 3, 5, 4), which equals $m_1 \oplus m_2$ due to the used cipher being additively homomorphic.

The only known plaintext-recovery attack on this cryptosystem lies in solving the multidimensional discrete logarithm problem, which can be described

by (4). General purpose algorithms, which are used for solving problems of this kind in arbitrary groups, are considered to be computationally difficult for non-quantum computers [7, 13]. Nevertheless, the special purpose algorithm proposed in the section below solves the aforesaid problem in polynomial time and does not require a quantum computer.

3. PLAINTEXT-RECOVERY ATTACK

The attack proposed in this section recovers the plaintext from the ciphertext of the matrix-based knapsack cipher by using only public information. This algorithm relies on the properties of the polynomials $B(\lambda)$ and $W_i(\lambda)$, which are defined over GF(q) in the following way:

$$B(\lambda) = \det(\lambda \cdot I - c),$$
$$W_i(\lambda) = \det(\lambda \cdot I - c \cdot e_i),$$

where c is the ciphertext chosen for decryption, $(e_1, e_2, ..., e_n)$ denotes the corresponding public key and I stands for the n-order unit matrix over GF(q). It is clear that $B(\lambda)$ and $W_i(\lambda)$ are characteristic polynomials [14] of the matrices c and $c \cdot e_i$ respectively. The theorem proposed below describes the relationship between these polynomials and the variables of the decryption procedure.

Theorem 1. If the permutation $(\sigma_1, \sigma_2, ..., \sigma_n)$ is used to generate the public key and the tuple $(y_1, y_2, ..., y_n)$ is obtained on the initial step of decryption, then y_{σ_i} can be found by the formula

$$y_{\sigma_i} = \lambda - \frac{B(\lambda)}{\gcd\left(B(\lambda), W_i(\lambda)\right)}, \quad (8)$$

where $\gcd(B(\lambda), W_i(\lambda))$ denotes the monic greatest common divisor of $B(\lambda)$ and $W_i(\lambda)$.

Proof. The isomorphism $f^{-1}(\mu)$ preserves the characteristic polynomial [14] of μ, since μ and $f^{-1}(\mu)$ are similar [14] matrices. The identity

$$f^{-1}(c \cdot e_i) = f^{-1}(c) \cdot g_{\sigma_i}$$

holds true due to (3). Therefore, $B(\lambda)$ and $W_i(\lambda)$ are characteristic polynomials of the matrices $f^{-1}(c)$ and $f^{-1}(c) \cdot g_{\sigma_i}$ respectively. Inasmuch as $f^{-1}(c) \in G$, it can be shown using (1) and (2) that $f^{-1}(c)$ differs from $f^{-1}(c) \cdot g_{\sigma_i}$ only in the (σ_i, σ_i) entry. Since both $f^{-1}(c)$ and $f^{-1}(c) \cdot g_{\sigma_i}$ are diagonal matrices, the foregoing implies that the difference between the root multisets of $B(\lambda)$ and $W_i(\lambda)$ contains only the (σ_i, σ_i) element of $f^{-1}(c)$. By virtue of the first step of the decryption algorithm, this element equals y_{σ_i}, so the aforesaid difference of multisets is $\{ y_{\sigma_i} \}$. Hence,

$$B(\lambda) = \varphi \cdot \left(\lambda - y_{\sigma_i}\right) \cdot \gcd\left(B(\lambda), W_i(\lambda)\right), \quad (9)$$

where $\varphi \in GF(q)$. Since $B(\lambda)$ is a monic polynomial, φ equals 1. Thus, (9) can be transformed into (8). ■

The value of z_{σ_i}, which is used on the third step of decryption, can be found without possessing the private key s. The identity

$$z_{\sigma_i} = \det(g_{\sigma_i})$$

follows from the definition of g_{σ_i}. Due to (3), the matrices e_i and g_{σ_i} are similar, so their determinants are equal. Therefore,

$$z_{\sigma_i} = \det(e_i), \quad (10)$$

where $(e_1, e_2, ..., e_n)$ is the public key tuple.

The plaintext-recovery attack receives the ciphertext c and the corresponding public key $(e_1, e_2, ..., e_n)$ as input. The cipher parameters q and n are considered to be specified along with the public key. The output of this algorithm is the recovered plaintext tuple $(x_1, x_2, ..., x_n)$. The attack procedure consists of the following steps:

1. The coefficients of $B(\lambda)$ are calculated, and the variable i is set to 0.
2. The coefficients of $W_i(\lambda)$ are computed, and the variable i is increased by 1.
3. The value of y_{σ_i} is obtained by (8).
4. The value of z_{σ_i} is computed using (10).
5. The plaintext fragment x_i is recovered in accordance with (5).
6. If $i < n$, the algorithm proceeds to the second step. Otherwise, the plaintext tuple $(x_1, x_2, ..., x_n)$ is output.

The following toy example of the aforementioned attack is constructed using the decryption instance described in Section 2. The input is represented by the ciphertext c, which equals c_p in (7), and the public key (e_1, e_2, e_3, e_4) defined by (6). The parameters q and n are 13 and 4 respectively.

The first two steps of the attack establish the identities

$$B(\lambda) = \lambda^4 + 6\lambda^3 + 5\lambda^2 + 5\lambda + 9,$$
$$W_1(\lambda) = \lambda^4 + \lambda^3 + 9\lambda^2 + 10\lambda + 2,$$
$$W_2(\lambda) = \lambda^4 + 8\lambda^3 + \lambda^2 + 8\lambda + 8,$$

$$W_3(\lambda) = \lambda^4 + 5\lambda^3 + \lambda^2 + 8\lambda + 11,$$
$$W_4(\lambda) = \lambda^4 + 3\lambda^3 + 4\lambda^2 + 5.$$

The intermediate calculations performed on the third step yield the following results:

$$\gcd\left(B(\lambda), W_1(\lambda)\right) = \lambda^3 + 7\lambda^2 + 12\lambda + 4,$$
$$\gcd\left(B(\lambda), W_2(\lambda)\right) = \lambda^3 + 11\lambda^2 + 8\lambda + 6,$$
$$\gcd\left(B(\lambda), W_3(\lambda)\right) = \lambda^3 + 4\lambda^2 + 10\lambda + 11,$$
$$\gcd\left(B(\lambda), W_4(\lambda)\right) = \lambda^3 + 9\lambda^2 + 6\lambda + 10.$$

The outcomes of the third and the fourth steps can be described as follows:

$$y_{\sigma_1} = 1,\ y_{\sigma_2} = 5,\ y_{\sigma_3} = 11,\ y_{\sigma_4} = 3,$$
$$z_{\sigma_1} = 6,\ z_{\sigma_2} = 11,\ z_{\sigma_3} = 7,\ z_{\sigma_4} = 2.$$

The fifth step determines that $x_1 = 0$, $x_2 = 3$, $x_3 = 5$ and $x_4 = 4$.

Hence, the algorithm outputs (0, 3, 5, 4), which equals the plaintext tuple obtained in Section 2 by decryption of the corresponding ciphertext.

4. TIME COMPLEXITY OF THE PLAINTEXT-RECOVERY ATTACK

The most computationally difficult arithmetic operations in GF(q) are multiplication and division. The last one for a finite field is multiplication by the inverse of the divisor. The time complexity for these operations is $O(\log^2(q))$ [15]. Thus, multiplication of η-degree polynomials in GF(q), as well as their division, takes $O(\eta^2 \cdot \log^2(q))$ time. Multiplication of η-order matrices over GF(q) has time complexity $O(\eta^3 \cdot \log^2(q))$.

Coefficients of the characteristic polynomial of an arbitrary η-dimensional matrix over GF(q) can be efficiently found using the Hessenberg algorithm by performing $O(\eta^3)$ arithmetic operations in the given field [16, 17]. Thus, the time complexity of the first two steps of the attack procedure is $O(n^3 \cdot \log^2(q))$.

The greatest common divisor of two polynomials over GF(q) can be calculated using the Euclidean algorithm by means of performing $O(\eta^2)$ field arithmetic operations, where η is the largest of the degrees of the aforementioned polynomials [15]. So the third step has time complexity $O(n^2 \cdot \log^2(q))$.

The determinant of a square matrix over GF(q) can be found using the Gaussian elimination in $O(\eta^3)$ field arithmetic operations, where η denotes the order of the given matrix [18, 19]. Thus, the time complexity of the fourth step is $O(n^3 \cdot \log^2(q))$.

A discrete logarithm in GF(q) can be efficiently computed using the Pohlig-Hellman algorithm by executing $O(\log^2(q))$ field arithmetic operations due to q - 1 being smooth or small [20, 21]. Therefore, the fifth step requires at most $O(\log^4(q))$ time.

Since each step except the first is performed by the attack algorithm n times, the foregoing implies that the time complexity of the plaintext-recovery attack is $O(n^4 \cdot \log^2(q) + n \cdot \log^4(q))$.

The time complexities of the considered attack and the decryption procedure can be compared in the following way. The first decryption step requires 2 multiplications of η-order matrices over GF(q). The optional next step is recommended to be omitted by means of storing (σ_1, σ_2, σ_3, σ_4) and (z_1, z_2, z_3, z_4) along with the private key. The last step consists in computing n discrete logarithms in GF(q). In light of the above, performing the decryption procedure requires $O(n^3 \cdot \log^2(q) + n \cdot \log^4(q))$ time. Therefore, if t denotes the decryption time, the time complexity of the plaintext-recovery attack is $O(t^{1.34})$.

5. CONCLUSION

The plaintext-recovery attack proposed in the present paper has time complexity $O(t^{1.34})$, where t stands for the decryption time of the attacked cryptosystem. In terms of the parameters of the matrix-based knapsack cipher, the time complexity of the given cryptanalytic method can be expressed as $O(n^4 \cdot \log^2(q) + n \cdot \log^4(q))$. Hence, the aforesaid encryption scheme can be considered broken and should not be used as a privacy tool. However, this cipher is still suitable for educational purposes as an example of the application of linear and abstract algebras in asymmetric cryptography. The obtained results help to eliminate the information security risks, which arise from the use of the aforesaid cipher in the absence of information about its vulnerability.

6. REFERENCES

[1] B. Schneier, *Applied Cryptography: Protocols, Algorithms and Source Code in C*, 20th ed., John Wiley & Sons, 2015, 784 p.

[2] H. van Tilborg, S. Jajodia, *Encyclopedia of Cryptography and Security*, 2nd ed., Springer, 2011, 1416 p.

[3] Y. Li, S. Schäge, Z. Yang, F. Kohlar, J. Schwenk, "On the security of the pre-shared key ciphersuites of TLS," *Proceedings of the 17th International Conference on Practice and Theory in Public-Key Cryptography (PKC 2014)*, Buenos Aires, Argentina, March 26-28, 2014, pp. 669-684.

[4] F. Schillinger, C. Schindelhauer, "End-to-end encryption schemes for online social

networks," *Proceedings of the 12th International Conference on Security, Privacy, and Anonymity in Computation, Communication, and Storage (SpaCCS 2019)*, Atlanta, USA, July 14-17, 2019, pp. 133-146.

[5] F. Maury, J.-R. Reinhard, O. Levillain, H. Gilbert, "Format Oracles on OpenPGP," *Proceedings of the Cryptographer's Track at the RSA Conference 2015 (CT-RSA 2015)*, San Francisco, USA, April 20-24, 2015, pp. 220-236.

[6] S. Ghosh, A. Kate, "Post-quantum forward secure onion routing (future anonymity in today's budget)," *Proceedings of the 13th International Conference on Applied Cryptography and Network Security (ACNS 2015)*, New York, USA, June 2-5, 2015, pp. 263-286.

[7] M. Campagna et al., *ETSI White Paper No. 8. Quantum Safe Cryptography and Security: An introduction, benefits, enablers and challenges*, European Telecommunications Standards Institute, 2015, 64 p.

[8] I. Damgård, J. Groth, G. Salomonsen, "The theory and implementation of an electronic voting system," in: D. A. Gritzalis (Eds.), *Secure Electronic Voting*, Springer, 2003, pp. 77-99.

[9] J. Liu, L. Chen, S. Mesnage, "Partially homomorphic encryption schemes over finite fields," *Proceedings of the 6th International Conference on Security, Privacy and Applied Cryptography Engineering (SPACE 2016)*, Hyderabad, India, December 14-18, 2016, pp. 109-123.

[10] A. Vambol, "The matrix-based knapsack cipher in the context of additively homomorphic encryption," *Proceedings of the 3rd International Conference on Computational Linguistics and Intelligent Systems (COLINS)*, Kharkiv, Ukraine, April 18-19, 2019, pp. 344-354.

[11] A. Zhivotova, N. Ziuliarkina, Y. Kostygina, "Modification of the cryptosystem with public key on the basis of knapsack problem," *UrFR Newsletter. Information Security*, issue 1 (11), pp. 16-20, 2014. (in Russian)

[12] A. Vambol, "The prospects for group-based knapsack ciphers in the post-quantum era," *Proceedings of the 9th IEEE International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, Ukraine, May 24-27, 2018, pp. 271-275.

[13] N. Smart, *Cryptography Made Simple*, Springer, 2015, 481 p.

[14] S. Roman, *Advanced Linear Algebra*, 2nd ed., Springer, 2005, 482 p.

[15] A. Menezes, P. van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996, 816 p.

[16] H. Cohen, *A Course in Computational Algebraic Number Theory*, Springer, 1993, 536 p.

[17] M. Law, M. Monagan, "Computing characteristic polynomials of matrices of structured polynomials," *Proceedings of the 18th International Workshop on Computer Algebra in Scientific Computing (CASC 2016)*, Bucharest, Romania, September 19-23, 2016, pp. 336-348.

[18] O. Vasilenko, *Number-Theoretic Algorithms in Cryptography*, American Mathematical Society, 2007, 243 p.

[19] P. Bürgisser, "Permanent versus determinant, obstructions, and Kronecker coefficients," *Séminaire Lotharingien de Combinatoire*, vol. 75, 2015, 19 p.

[20] S. Pohlig, M. Hellman, "An improved algorithm for computing logarithms over GF(p) and its cryptographic significance," *IEEE Transactions on Information Theory*, vol. 24, issue 1, pp. 106-110, 1978.

[21] H. van Tilborg, *Fundamentals of Cryptology: A Professional Reference and Interactive Tutorial*, Springer, 2000, 506 p.

Aleksei Vambol *is a PhD student at the Department of Computer Systems, Networks and Cybersecurity of National Aerospace University «KhAI», Kharkiv, Ukraine. He received the master's degree in specialized computer systems at the same university. His areas of scientific interest include public key ciphers, digital signatures, key-agreement protocols, post-quantum cryptography, number theory and abstract algebra.*