UDC 681.3.06 (0.43)
DOI: 10.15587/1729-4061.2020.210683

DEVELOPMENT OF A MODIFIED UMAC ALGORITHM BASED ON CRYPTO-CODE CONSTRUCTIONS

A. Gavrilova
Senior Lecturer*

I. Volkov
PhD
Scientific-Research Center of Missile Troops and Artillery
Herasima Kondratieva str., 165, Sumy, Ukraine, 40021

Y. Kozhedub
PhD
Scientific and Organizational Department of the Scientific and Research Center
Institute of Special Communication and Information Security of
National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute"
Verkhnokliuchova str., 4, Kyiv, Ukraine, 03056

R. Korolev
PhD*

O. Lezik
PhD, Associate Professor
Department of Defense Police Tactics SB**

V. Medvediev
PhD, Professor
Department of Radio-Technical and Special Troops
National Defence University of Ukraine named after Ivan Cherniakhovskyi
Povitroflotskyi ave., 28, Kyiv, Ukraine, 03049

O. Milov
PhD, Professor*
E-mail: Oleksandr.Milov@hneu.net

B. Tomashevsky
PhD, Associate Professor
Department of Cyber Security
Ternopil Ivan Puluj National Technical University
Ruska str., 56, Ternopil, Ukraine, 46001

A. Trystan
Doctor of Technical Sciences, Senior Research,
Head of Scientific Research Department
Head of Scientific Research Department Scientific Center**

O. Chekunova
PhD**

*Department of Cybersecurity and Information Technologies
Simon Kuznets Kharkiv National University of Economics
Nauky ave., 9-A, Kharkiv, Ukraine, 61166
**Ivan Kozhedub Kharkiv National Air Force University
Sumska str., 77/79, Kharkiv, Ukraine, 61023

The development of computer technology has determined the vector for the expansion of services based on the Internet and "G" technologies. The main requirements for modern services in the banking sector are security and reliability. At the same time, security is considered not only as ensuring the confidentiality and integrity of transactions, but also their authenticity. However, in the post-quantum period, US NIST specialists question the durability of modern means of providing basic security services based on symmetric and asymmetric cryptography algorithms. The increase in computing resources allows attackers to use modern threats in combination. Thus, there is a need to search for new and/or modify known algorithms for generating MAC (message authentication codes). In addition, the growth of services increases the amount of information that needs to be authenticated. Among the well-known hash algorithms, the hash functions of universal hashing are distinguished, which allow initially determining the number of collisions and their uniform distribution over the entire set of hash codes. Possibilities of modifying the cascade hashing algorithm UMAC (message authentication code based on universal hashing, universal MAC) based on the use of McEliece crypto-code construction on algebrogeometric (elliptic codes (EC), modified elliptic codes (MEC) and damaged codes (DC). This approach allows preserving the uniqueness property, in contrast to the classical UMAC scheme based on a block symmetric cipher (AES). The presented algorithms for evaluating the properties of universality and strict universality of hash codes make it possible to evaluate the security of the proposed hashing constructs based on universal hash functions, taking into account the preservation of the universality property

Keywords: authenticity, hashing algorithm, crypto-code constructions, elliptic codes, modified elliptic codes, damaged codes, UMAC algorithm, MV2 algorithm (universal damage mechanism), post-quantum cryptography

Received date 25.07.2020
Accepted date 17.07.2020
Published date 31.08.2020

1. Introduction

The development of the banking sector in the last decade has made it possible to significantly expand the range of its services based on the use of computing resources of Internet technologies and X "G" –LTE (Long-Term Evolution) technologies.

These changes contribute to the development of the digital economy, and in particular, electronic banking [1, 2]. However, this is accompanied by an increase in the number and diversity of cyber threats. In [3–5], the results of the analysis of cyber threats to automated banking systems (ABS) of banking sector organizations (BSO) over the past three years are presented (Fig. 1).

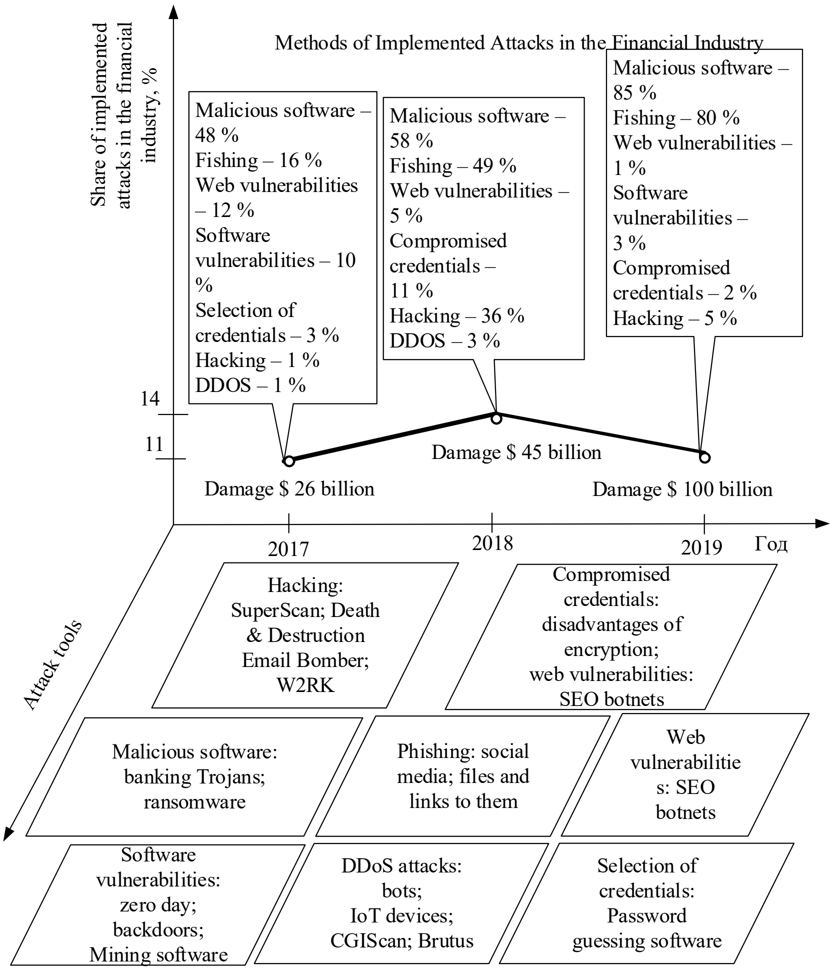

Share of implemented attacks in the financial industry, %
Methods of Implemented Attacks in the Financial Industry
Malicious software – 48 %
Fishing – 16 %
Web vulnerabilities – 12 %
Software vulnerabilities – 10 %
Selection of credentials – 3 %
Hacking – 1 %
DDOS – 1 %
Malicious software – 58 %
Fishing – 49 %
Web vulnerabilities – 5 %
Compromised credentials – 11 %
Hacking – 36 %
DDOS – 3 %
Malicious software – 85 %
Fishing – 80 %
Web vulnerabilities – 1 %
Software vulnerabilities – 3 %
Compromised credentials – 2 %
Hacking – 5 %
14
11
Damage $ 45 billion
Damage $ 26 billion
Damage $ 100 billion
2017
2018
2019
Год
Attack tools
Hacking: SuperScan; Death & Destruction Email Bomber; W2RK
Compromised credentials: disadvantages of encryption; web vulnerabilities: SEO botnets
Malicious software: banking Trojans; ransomware
Phishing: social media; files and links to them
Web vulnerabilitie s: SEO botnets
Software vulnerabilities: zero day; backdoors; Mining software
DDoS attacks: bots; IoT devices; CGIScan; Brutus
Selection of credentials: Password guessing software

Fig. 1. Trends in the focus of cyber-attacks on the financial sector: DDOS — Distributed Denial of Service; SuperScan — powerful TCP port scanner; Death&Destruction — deadly & devastating mail bomber, a program for organizing spam on a mail server; W2RK — Windows 2000 Resource Kit — Windows Administrator Toolkit; SEO — Search Engine Optimization — a set of measures to increase the visibility of a site in search engines for targeted search queries; IoT — Internet of Things

The above graph shows that in 2017–2019, the extent of damage from implemented attacks in the financial sector was significantly influenced by such methods of implemented attacks as phishing (2017 – 16 %, 2018 – 49 %, 2019 – 80 %) and malicious software (2017 – 48 %, 2018 – 58 %, 2019 – 85 %). From the point of view of the main vectors of modern attacks on ABS, the analysis carried out indicates their integration with the methods of social engineering. This leads to the appearance of the properties of hybridity and synergism in the already known threats [3–5].

The presented statistics show that the number of threats associated with the service of authenticity is growing steadily. Therefore, it is important to recognize scientific research focused on the formation of new approaches that provide an authenticity service in the face of the rapid growth of computing resources and modern threats based on a full-scale quantum computer, based on the development of a modified UMAC algorithm with McEliece crypto-code constructions (as a pad).

2. Literature review and problem statement

With the growth of computing resources and modern technologies for increasing data volumes, an integrated task arises to ensure not only security, efficiency, but also authenticity. For implementation, it is proposed to use the UMAC cascade hashing algorithm (message authentication code based on universal hashing), which at the same time allows providing the required level of security and efficiency based on the use of universal hashing functions. However, the classical scheme uses the Advanced Encryption Standard (AES) block-symmetric cipher algorithm to ensure the strength of the hash code, which ultimately does not allow for universality.

The works [3–5] present the results of studies of the vectors of cyber threats. The analysis shows that the use of threats aimed at cracking authentication mechanisms allows remote access to confidential information and/or to obtain “privileges” that provide the possibility of “hacking” an integrated information security system (IISS). In [6–8], the possibilities of using McEliece and Niederreiter crypto-code constructions, providing security services – confidentiality, integrity, are considered. In addition, the efficiency of crypto-transformations and reliability of data transmission are provided on the basis of the use of algebraic geometric (cyclic) codes. A significant disadvantage of their practical use is the possibility of finding elements of masking matrices (the private key of each user of the system) based on the Sidelnikov’s attack [9] and significant energy costs for practical implementation over a finite (computational) field of dimension GF (2^{10}–2^{13}) (Galois field). As known, a finite field is a finite set that defines arbitrary operations called addition, multiplication, subtraction and division.

In [10–15], the possibilities of full-scale quantum computers are considered, which provide the problem of breaking symmetric and asymmetric cryptography algorithms in polynomial time. Thus, the implementation of an attack on a quantum computer practically casts doubt on the stability of hashing algorithms based on block-symmetric ciphers in the CBC (Cipher Block Chaining) and CFB (Cipher Feedback Mode) modes.

The analysis of the possibilities of cryptanalysis in [16] confirms that based on the Shor and Grover quantum algorithms and a full-scale quantum computer, symmetric and asymmetric cryptography algorithms are susceptible to breaking in polynomial time.

The works [17, 18] consider the possibility of constructing a cascade UMAC algorithm based on the use of MASH-1 and MASH-2 keyless algorithms (MASH – Modular Arithmetic Secure Hash) as the formation of a pseudo-random pad on the third layer. However, the results presented by the authors indicate that the MASH-1 algorithm does not provide the required stability and versatility parameters and cannot be used in a modified (improved) algorithm. The MASH-2 algorithm provides the required level of cryptographic strength and retains the property of universality of the generated hash code in the improved UMAC, but does not provide its use in online mode, due to significant computing resources. In [19], the possibility of forming hash functions based on the use of cyclic algebraic geometric noise-resistant codes is considered. However, their use in the UMAC algorithm has not been considered, and research on their practical implementation has not been carried out. This approach provides versatility and allows the use of crypto-code constructs as a pseudo-random pad in a cascade hashing algorithm. In [7, 10, 20], hybrid crypto-code constructions (HCC) are considered based on the synthesis of the classical McEliece and Niederreiter schemes. However, the use of two schemes increases not only the capacitive costs for its practical implementation, but also reduces the efficiency of crypto-transformations, which is essential for use in the formation of the MAC code. In [21], the authors consider the use of Reed-Solomon cyclic codes, but they do not study the resistance of such a crypto-code structure to the Sidelnikov's attack, which does not allow using CCC as a "guarantor" of the hash code strength. In [22], the possibility of using the Niederreiter crypto-code construction in post-quantum cryptography is considered; however, this scheme uses two algorithms (equilibrium coding and McEliece's schemes), which complicates the formation of the pseudo-pad and practical implementation. In [23, 24], the security of universal hash functions is considered. The authors confirm that the use of universal hashing does not allow the formation of hack-resistant hash codes unambiguously, and suggest using additional encryption to ensure the strength of the hash code. This approach increases the computational costs of their online implementation. The work [25] proposes the use of a new multicast authentication scheme based on a symmetric algorithm. However, in the post-quantum period, this algorithm can be cracked, and the use of a modified cascade hashing algorithm will provide the required level of security and efficiency. The possibilities of forming a hash code based on the UMAC algorithm, proposed in [7, 10, 17–25], provide the property of universality, however, this requires significant energy costs for their practical implementation and does not ensure the use of the algorithm in the online mode of digest generation. The formation of a modified UMAC algorithm based on crypto-code constructions will provide a solution to the authentication problem in post-quantum cryptography, the required level of efficiency (online hash code generation) with the further growth of information data arrays, and a decrease in energy consumption during their practical implementation.

3. The aim and objectives of the study

The aim of the study is to develop a modified UMAC algorithm based on crypto-code constructions and algorithms for assessing the strength of the hash code.

To achieve the aim, the following objectives are set:

– to consider the requirements for algorithms of universal and strictly universal classes of hash functions;

– to analyze the construction of the cascade UMAC algorithm, taking into account the provision of universality;

– to develop algorithms for the modified UMAC algorithm based on the McEliece crypto-code construction on algebraic geometric and damaged codes;

– to develop algorithms for assessing the strength of hash codes of the modified UMAC algorithm based on evaluating the criteria for universality and strict universality of the classes of hash functions.

4. Basic requirements for algorithms of universal and strictly universal classes of hash functions

Universal classes of hash functions were first proposed in [26]. The basic properties of universality and strict universality of classes of hash functions were studied in [27–29].

The idea of universal hashing is to define such a set of elements of a finite set H of hash functions $h: A\rightarrow B$, $|A|=a$, $|B|=b$ (A – the set of outgoing messages; B – the set of MAC codes; |A| – number of outgoing messages; $|B|$ – the number of possible states of MAC codes; a – outgoing message; b – the state of MAC codes), so that random selection of the function $h\in H$ would provide a low probability of collision, i. e. for any different inputs x_1 and x_2, the probability that $h(x_1)=$ $=h(x_2)$ (probability of collision) cannot exceed some predetermined value (fixed precision) ε:

$$P_{col}=P\big(h(x_1)=h(x_2)\big)\le\varepsilon,$$

wherein the collision probability can be calculated as

$$P_{col}=\frac{\delta_H(x_1,x_2)}{|H|},$$

where $\delta_H(x_1,x_2)$ – the number of hash functions in H, at which the values $x_1, x_2\in A$, $x_1\neq x_2$ cause a collision, i. e. $h(x_1)=h(x_2)$.

The works [18, 30] give definitions of universal hashing:

1. Assuming that $0<\varepsilon<1$. H is an ε-universal hash class (abbreviated $\varepsilon-U(H,\ A,\ B)$), if for two different elements $x_1, x_2\in A$ there is no more than $(|H|\times\varepsilon)$ functions $f\in H$ such that $h(x_1)=h(x_2)$, if $\delta_H(x_1,x_2)\le\varepsilon|H|$ for all $x_1, x_2\in A$, $x_1\neq x_2$.

2. Assuming that $0<\varepsilon<1$. H is an ε-strictly universal hash class (abbreviated $\varepsilon-SU(H, A, B)$) if the following conditions are met:

– for each $x_1\in B$, and for each $y_1\in B$,

$$\left|\left\{h\in H: h(x_1)=y_1\right\}\right|=|H|/|B|;$$

– for each $x_1, x_2\in A$, and $x_1\neq x_2$ for each $y_1, y_2\in B$,

$$\left|\left\{h\in H: h(x_1)=y_1, h(x_2)=y_2\right\}\right|\le\varepsilon|H|.$$

The definition of a universal class of hash functions is equivalent to the definition of the MAC algorithm, in which the number of different rules for generating a hash code (the number of keys), in which there is a collision for two arbitrary input sequences, is limited. The number of such keys cannot exceed the value $\left(P_{кол}\times|H|\right)$, where P_{col} – collision probability, $|H|$ – the number of all rules (keys).

The universal class of hash functions satisfies such security conditions as resistance to a preimage, to another preimage, and to collision [29].

Authentication schemes that are equivalent to the universal classes of hash functions have not been put into practice due to their low resistance to intrusion threats. Consequently, in the definition of a universal class of hash functions, there is no condition that determines its resistance to differential analysis.

The ideas of universal authentication were developed in the theory of unconditional authentication using strictly universal hashing [18, 31]. So, in [19], a definition of a strictly universal class of hash functions is given, which is equivalent to the definition of such an algorithm for generating data integrity and authenticity control codes, under which the following rules will be fulfilled.

1. The number of MAC generation rules (number of keys), under which the value of the data integrity and authenticity control code does not change for an arbitrary input sequence, is limited. The number of such keys cannot exceed the value $|H|/|B|$.

2. The number of rules for generating the data integrity and authenticity control code (the number of keys), under which the corresponding MAC values do not change for two arbitrary input sequences, is limited. The number of such keys cannot exceed the value $P_{кол} \times |H|$.

The probability of collision of data integrity and authenticity control codes in a scheme with strictly universal hashing is defined as $P_{col} \leq \varepsilon$.

The conditions of strictly universal classes of hash functions are more "stringent" in comparison with the requirements of universal classes. Authentication schemes equivalent to strictly universal classes have a number of advantages, which are determined by the exact value of the collision probability, aversion to frequency and differential analysis, etc. However, the construction of such schemes is very problematic. Practical schemes are known that are equivalent to strictly universal classes, which have a significant drawback – a large amount of key data exceeding the amount of information.

Thus, to assess the strength of hash codes obtained on the basis of universal classes of hash functions, it is practically enough to evaluate the fulfillment of the criteria of universality and strict universality. In addition, knowing the number of collisions and even distribution of hash codes over the entire set allows using collision data as identifiers for large amounts of data and reducing the time to find the information needed.

5. Analysis of the construction of the cascade UMAC algorithm, taking into account the universality

For remote electronic interaction, it is important to ensure the process of object recognition by the presented parameters (indicators) and the associated authentication process. This problem is most acute in access control systems and authentication and verification systems for message transmission [32–40].

Information hashing is an effective mechanism for controlling the integrity and authenticity of information in modern telecommunication networks [33–36]. Today manipulation detection codes (MDC) for data integrity control, and message authentication codes (MAC) are used [34, 40–45].

The UMAC message authentication code was proposed in [35]. The algorithm is based on families of universal hash functions and provides provable security of the generated MAC [33–42]. At the same time, the security of the algorithm is justified by the strength of the AES block symmetric cipher (BSC) used in the UMAC scheme in the CBC mode (plain text block chaining) when forming the pad for the third layer of the UHASH-16 or UHASH-32 function (UHASH is a universal hash function, with a fixed hash code of 16 and 32 bits, respectively).

In [36], the construction of multilayer hashing functions by the example of the UMAC algorithm is presented as a symbiosis of multi-stage key universal hashing and the use of a symmetric block cipher to form the so-called pseudo-random pad. The use of universal hashing in the multilayer construction of UMAC allows ensuring the equiprobability of the formation of hash images for the entire set of key data used, on which the proof of the security of the algorithm is based [17, 18, 25, 37, 41–44]. The use of the AES encryption algorithm provides high cryptographic strength of the UMAC scheme [33–37, 45].

The UMAC message authentication code generation scheme uses a multi-level universal hashing construction *Hash*(*K,M,Taglen*) and pseudo-random pad generation procedure *Pad*. The use of universal hashing makes it possible to ensure the equiprobability of the formation of hash images for the entire set of key data used, on which the proof of the algorithm's security is based [45]. The UHash function compresses a message made up of three different layers:

– compression: the first layer uses the fast hash family NH to compress the message to a specified ratio;

– fixed length hash: the second layer uses the RP hash family, not as fast as NH, but generates a fixed length output using a fixed length key;

– strengthen and fold: the third layer uses the IP hash family, which reduces the length of its input to a more appropriate size.

Fig. 2 shows the general principle of the algorithm for generating the MAC code using the class of universal hash functions UMAC-16 and UMAC-32.

On the first layer L1 of the Hash function, the NH hash function is used, the result of which is the division of the message into blocks of 1,024 bytes. In this case, messages are received 128 times less than the input (Fig. 3).

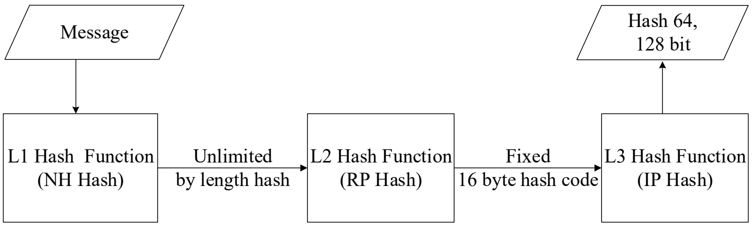


Fig. 2. Scheme of the universal class UHASH

After block A is formed, the received data are transferred to the second layer L2 of the Hash function (Fig. 4): A – input message, L2-Key – key, ADD – concatenation mod 32, MOD p64 – operation of calculating the remainder of division by a 64-bit long prime number, MULT – multiplexer, FF – data, Reset – reset, SEL – select operation, ZERO PAD – zero vector, B – output block, Compare $2^{64}–2^{32}$ – comparison operation, K^2 mod p, K mod p – operations with keys.

This layer uses RP. The universal RP family of hash functions is based on polynomial computation. A string of n words with a length of n bits can be viewed as a polynomial of degree *n* in the range, where each word in the string serves as a coefficient. To calculate the hash, it is necessary to calculate the polynomial for an arbitrarily chosen point (key).

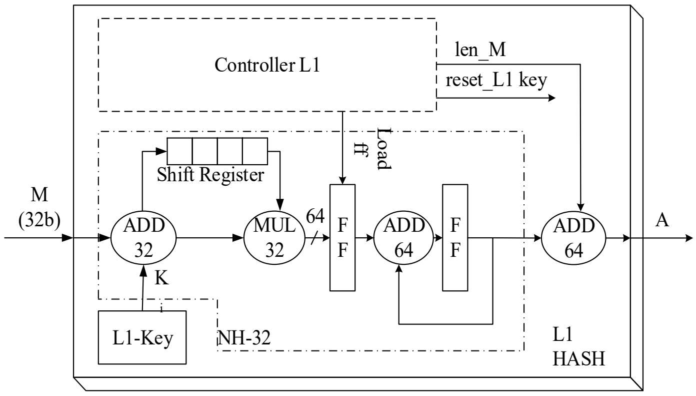


Fig. 3. Scheme of the first layer L1 of the Hash function: M — input message, L1-Key — key, ADD 32(64) — concatenation mod 32, Shift Register — shift register, MUL 32 — multiplication by module 32, FF — data , Load ff — data download, reset_L1 key — key reset, len_M — message length, A — output block

The purpose of this layer is to transform the result obtained on the first layer using the POLY polynomial function. If the length of the input vector is longer than 1,024 bits, then the polynomial function uses additional parameters to form the remainder.

The third layer L3 of the Hash function uses the IP function (Fig. 5).

The layer with the generic IP hash family reduces the length of its input, since the RP hash layer generates outputs that are much larger than the expected probability of error. It is based on the processing of internal data in the range of values (multiplying the input words with keywords and composing the results). The probability of an error from the previous layer has passed to this one and remains.

The purpose of the third layer is to transform the input vector B with a length of 16 bytes into a string equal to 4 bytes.

The formation of the pseudo-random pad with a cryptographically strong algorithm of the AES symmetric block cipher ensures the cryptographic strength of the UMAC algorithm at the level of the applied cryptoalgorithm [14]. This UMAC generation scheme has potentially high efficiency rates, which is ensured by overlaying pseudo-random pads *Pad*=*Hash*(*K, Nonce, Taglen*) on the generated hash codes *Y*=*Hash*(*K, M, Taglen*). The overlay procedure is equivalent to the bitwise addition operation. This approach is the classic implementation method for the UMAC algorithm.

The considered multi-layer construction in the formation of UMAC has potentially high efficiency rates. But after overlaying pseudo-random pads on the last layer, the UMAC formation algorithm loses the "universality" property of hashing and its collision properties are significantly deteriorated. This is due to the use of block symmetric encryption, which does not guarantee the preservation of the universality property of the resulting MAC code [18, 19, 32].

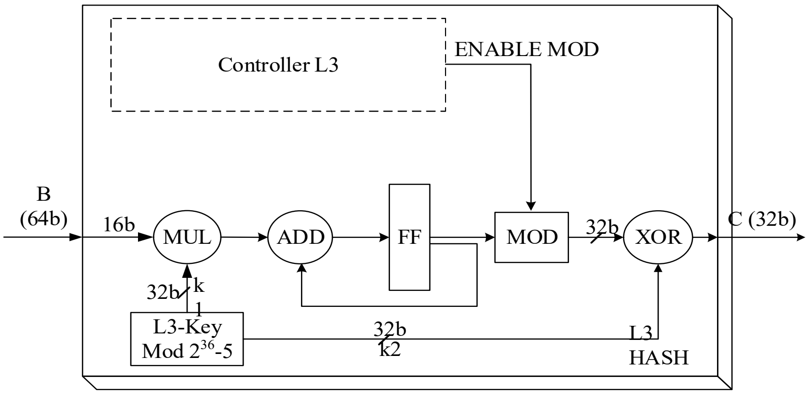


Fig. 5. Scheme of the third layer L3 of the Hash function: B — input message, L3-Key — key, ADD — concatenation, MOD — operation for calculating the remainder of the division, MUL — multiplication, FF — data, Reset — reset, SEL — selection operation, Enable Mod — a command that allows performing MOD, C — output block

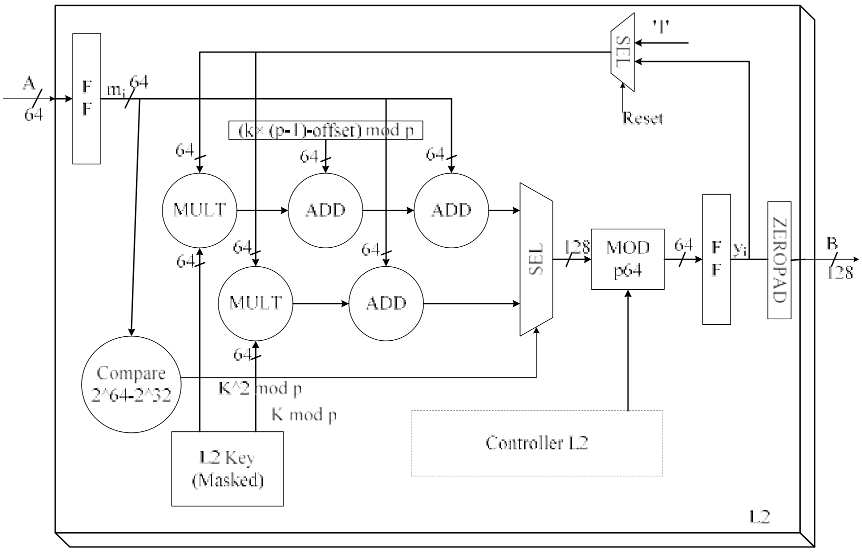


Fig. 4. Scheme of the second layer L2 of the Hash function

To ensure an increase not only in the cryptographic strength of message encryption algorithms for transmission over communication channels, but also to preserve universality, in [19, 32] it is proposed to use the universal hashing based on modular transformations. The provision of high cryptographic stability of transmitted messages occurs due to the impossibility of decrypting these messages during computing time. But this method is stable only with existing computing power, and with the advent of high-performance quantum computers, the risk of breaking it will increase.

In [19], the application of universal hashing based on modular transformations using the RSA algorithm (Rivest, Shamir, Adleman), which is based on elliptic curves and the computational complexity of the large-number factorization problem, is considered. At the final stage of hashing, this approach provides processing of strictly universal hashing by a cryptographically strong function based on modular transformations using loop functions

$$f(x_i,H_{i-1})=(x_i\oplus H_{i-1})^e \bmod(N)$$

and/or

$$f(x_i,H_{i-1})=\left(\alpha^{x_i\oplus H_{i-1}}\right)\bmod(p).$$

In this case, the bulk of information data is processed by the first layers of universal hashing. And the pseudo-random pa is processed by a cryptographically strong strictly universal hashing function based on modular transformations using the RSA algorithm. However, this approach does not ensure the efficiency of crypto-transformations, and its strength in post-quantum cryptography does not meet the requirements.

In [32], an improved method for generating data integrity and authenticity control codes is proposed based on the use of the UMAC algorithm. The first two layers are high-speed, but cryptographically weak universal hashing schemes, the last layer is proposed to be implemented using the developed secure (cryptographically strong) strictly universal hashing scheme based on modular transformations.

Generation of message authentication codes using key hashing, built on the basis of the keyless MASH-2 algorithm with variable initialization vectors, in certain cases allows building universal and strictly universal classes of hash functions. However, this condition is not satisfied for all values of the initial parameters (primes *p* and *q*).

Formally, the proposed scheme for the cascade generation of data integrity and authenticity control codes using modular transformations is shown in Fig. 6. This approach makes it possible to ensure the universality and stability of hash codes, but is practically not applicable in the online mode of hash code generation due to the low speed of forming the pseudo-random pad based on the keyless MASH-2 algorithm.

Thus, to eliminate the revealed drawback, it is proposed to use crypto-code constructions on algebraic geometric and damaged codes as a mechanism for forming the pseudo pad for the third layer of the UMAC cascade hashing algorithm.

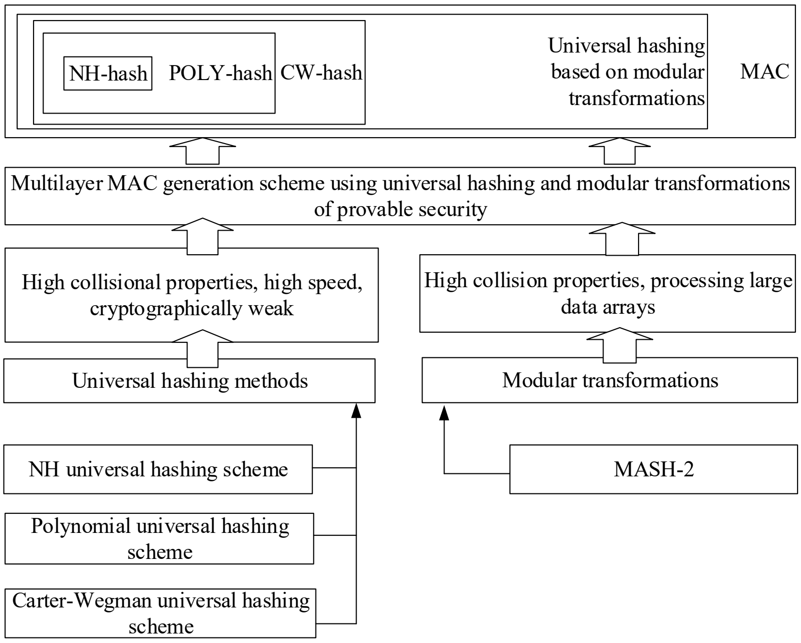

NH-hash
POLY-hash
CW-hash
Universal hashing based on modular transformations
MAC
Multilayer MAC generation scheme using universal hashing and modular transformations of provable security
High collisional properties, high speed, cryptographically weak
High collision properties, processing large data arrays
Universal hashing methods
Modular transformations
NH universal hashing scheme
MASH-2
Polynomial universal hashing scheme
Carter-Wegman universal hashing scheme

Fig. 6. Scheme of cascading generation of data integrity and authenticity control codes using the MASH-2 keyless hashing algorithm

The main approaches to ensuring the reduction of energy costs for the practical implementation of CCC on EC, MEC, DC are considered in [14, 46, 47]. This approach allows forming a modified UMAC algorithm depending on the requirements and computational capabilities.

6. Development of algorithms for modified UMAC based on McEliece CCC on EC, MEC and DC

In [14, 47–50], algorithms for constructing McEliece CCC on EC (MEC), DC, which can be used to form the pseudo-random pad, are considered.

So to build the pad on McEliece CCC on EC, the following algorithm is used:

– *Step 1.* Entering the information to be encoded (use the plain text of the message). Entering the public key G_X^{EC};

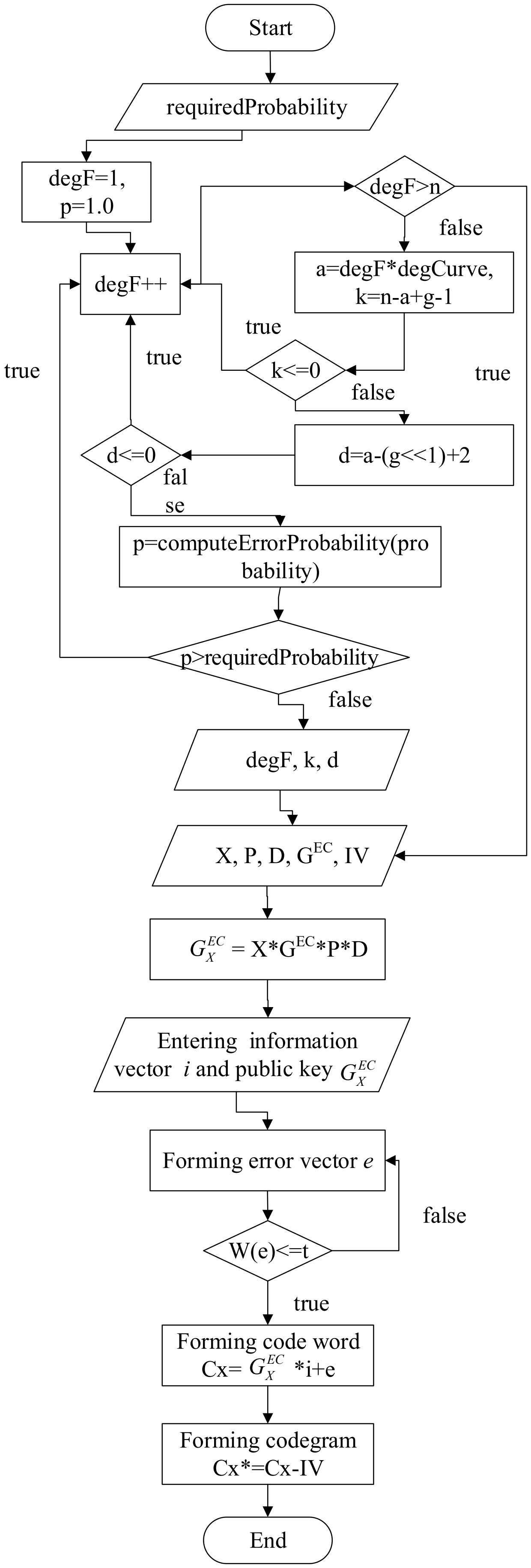
– *Step 2.* Encoding information with an elliptic code. Formation of a code word c_X of the elliptic code given by the matrix G_X^{EC} (forming the first part of the pad);

– *Step 3.* Formation of the error vector *e*, whose weight does not exceed ≤ t – correcting ability of elliptic code (formation of the second part of the pad);

– *Step 4.* Formation of the codogram (pad) $c_X^*=c_X+e$.

Thus, the code generated on EC is the pad that is used on the third layer of the modified UMAC algorithm on McEliece CCC. This approach ensures the preservation of the universality property of the obtained hash code and the required level of efficiency. However, it requires significant energy costs to ensure the required level of durability (it is necessary to build a CCC over GF(2^{10}–2^{13}). To reduce energy costs, it is proposed to use McEliece CCC on MEC, which allows reducing energy costs (building CCC over GF(2^6–2^8) and provide the required level of durability.

In [51] (Fig. 7) and in [14] (Fig. 8), algorithms for forming the pad for generating a hash code in a modified UMAC algorithm on McEliece CCC with shortened or lengthened MECs are presented.



Stage 1. Set code parameters

requiredProbability – defined probability of block distortion;
n - total number of symbols in code (code length);
k – number of information symbols;
d – minimal distance of code combinations by Hemming;
g – curve genus;
degF – the degree of generating function;
degCurve – curve degree.

Stage 2. Forming private and public keys of asymmetric cryptosystem, entering information package

X – non-degenerate matrix *kxk* over *GF(q)*;
P – permutational matrix *nxn* over *GF(q)*;
D – diagonal matrix *nxn* over *GF(q);*
G^{EC} – check matrix *rxn* of elliptic code over *GF(q)*;
a_i – coefficients set of curve polynomial $a_1 \dots a_6$;
IV – initialization vector, $IV=|h|=1/2k$-reducing elements.

Stage 3. Forming session key and codegram

vector *e* forms randomly,equiprobably and independently from another secret texts;
communication channel receives code without zero elements of initialization vector (shortening operation)

Fig. 7. Algorithm of forming a codogram in a modified McEliece CCC with a shortened modified code (MEC)

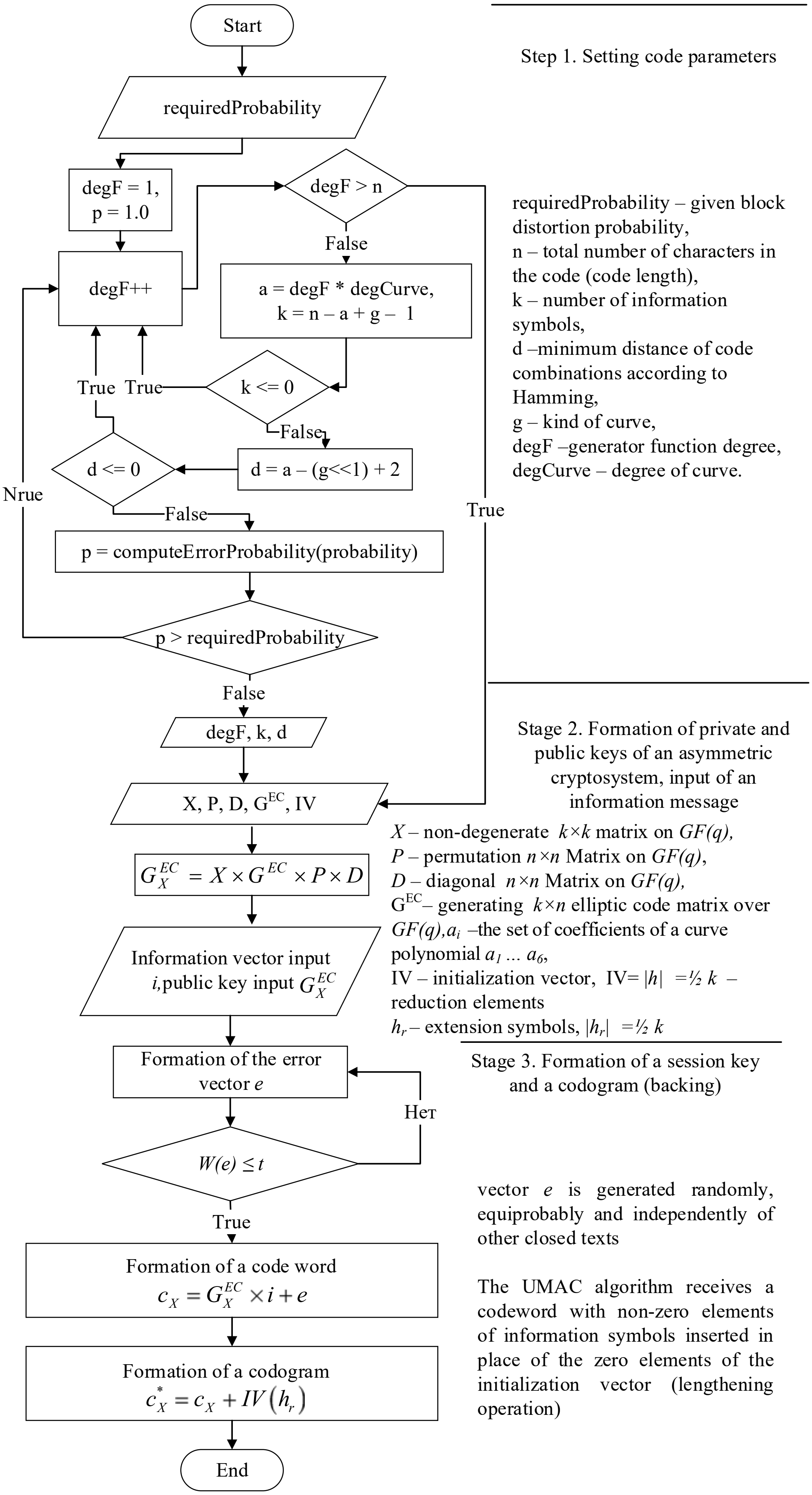

Start
requiredProbability
degF = 1,
p = 1.0
degF > n
False
degF++
a = degF * degCurve,
k = n – a + g – 1
True True
k <= 0
False
d <= 0
d = a – (g<<1) + 2
Nrue
True
False
p = computeErrorProbability(probability)
p > requiredProbability
False
degF, k, d
X, P, D, G^{EC}, IV
$G_X^{EC} = X \times G^{EC} \times P \times D$
Information vector input
i,public key input G_X^{EC}
Formation of the error vector e
Нет
$W(e) \le t$
True
Formation of a code word
$c_X = G_X^{EC} \times i + e$
Formation of a codogram
$c_X^* = c_X + IV(h_r)$
End
Step 1. Setting code parameters
requiredProbability – given block distortion probability,
n – total number of characters in the code (code length),
k – number of information symbols,
d –minimum distance of code combinations according to Hamming,
g – kind of curve,
degF –generator function degree,
degCurve – degree of curve.
Stage 2. Formation of private and public keys of an asymmetric cryptosystem, input of an information message
X – non-degenerate $k \times k$ matrix on $GF(q)$,
P – permutation $n \times n$ Matrix on $GF(q)$,
D – diagonal $n \times n$ Matrix on $GF(q)$,
G^{EC}– generating $k \times n$ elliptic code matrix over $GF(q)$,a_i –the set of coefficients of a curve polynomial $a_1 \ldots a_6$,
IV – initialization vector, IV= $|h| = ½ k$ – reduction elements
h_r– extension symbols, $|h_r| = ½ k$
Stage 3. Formation of a session key and a codogram (backing)
vector e is generated randomly, equiprobably and independently of other closed texts
The UMAC algorithm receives a codeword with non-zero elements of information symbols inserted in place of the zero elements of the initialization vector (lengthening operation)

Fig. 8. Algorithm of forming a codogram in a modified McEliece CCC with an extended modified code (MEC)

The results of the study showed that the application of this approach in practice contributes both to an increase in the cryptographic strength of the generation of an authentication code and raises the issue of the speed of operations related to this process.

To construct crypto-code constructions based on modified (shortened or extended) cyclic codes on elliptic curves, the parameters presented in Table 1 are used. Table 2 shows the parameters of asymmetric cryptosystems on the corresponding CCC.

Table 1

Main (*n, k, d*) parameters of MEC

Parameters	shortened MEC	elongated MEC
(n, k, d) parameters of the code built through the mapping of type $\varphi: X \to P^{k-1}$	$n = 2\sqrt{q} + q + 1 - x$, $k \geq \alpha - x$, $d \geq n - \alpha$, $\alpha = 3 \times \deg F, k + d \geq n$	$n = 2\sqrt{q} + q + 1 - x + x_1$, $k \geq \alpha - x + x_1$, $d \geq n - \alpha$, $\alpha = 3 \times \deg F$
(n, k, d) parameters of the code built through the mapping of type $\varphi: X \to P^{r-1}$	$n = 2\sqrt{q} + q + 1 - x$, $k \geq n - \alpha$, $d \geq \alpha$, $\alpha = 3 \times \deg F$, $k + d \geq n$	$n = 2\sqrt{q} + q + 1 - x + x_1$, $k \geq n - \alpha$, $d \geq \alpha, \alpha = 3 \times \deg F$

In [14, 49–51], practical algorithms for constructing a modified UMAC algorithm on CCC with MEC are considered.

The proposed approach will make it possible to form the pseudo-random pad with a further reduction in computing resources by reducing the computational field of elliptic curves based on damage (reduction or lengthening of the code sequence).

The process of confirming the integrity of information during transmission from the sending to the receiving side on the basis of checking the verification of codograms and hash codes using the McEliece CCC on MEC is schematically shown in Fig. 9.

The use of the UMAC algorithm on the proposed crypto-code constructions will make it possible to detect modifications of the plain text when transmitted through an open channel. When developing a mathematical model for the formation of a hash code in the UMAC algorithm, a pseudo-random sequence is used, which ensures the cryptographic strength of this hash code. The algorithm of forming the pad is the McEliece crypto-code construction on MEC.

Application of modification changes to elliptic codes allows reducing the load on computing resources and leads to an increase in the efficiency of generating MAC codes in real time.

Table 2

Main parameters of modified cryptosystems based on McEliece CCC on MEC

Parameters	Shortened MEC	Elongated MEC
Secret key dimension	$l_{K+} = x \times \left\lceil \log_2\left(2\sqrt{q} + q + 1\right) \right\rceil$	$l_{K+} = \left(x - x_1\right) \times \log_2\left(2\sqrt{q} + q + 1\right)$
Information vector dimension	$l_I = (\alpha - x) \times \mathrm{m}$	$l_I = (\alpha - x + x_1) \times \mathrm{m}$
Cryptogram dimension	$l_S = \left(2\sqrt{q} + q + 1 - x\right) \times m$	$l_S = \left(2\sqrt{q} + q + 1 - x + x_1\right) \times m$
Relative encoding rate	$R = (\alpha - x) / \left(2\sqrt{q} + q + 1 - x\right)$	$R = \left(-x + x_1\right) / \left(2\sqrt{q} + q + 1 - x + x_1\right)$

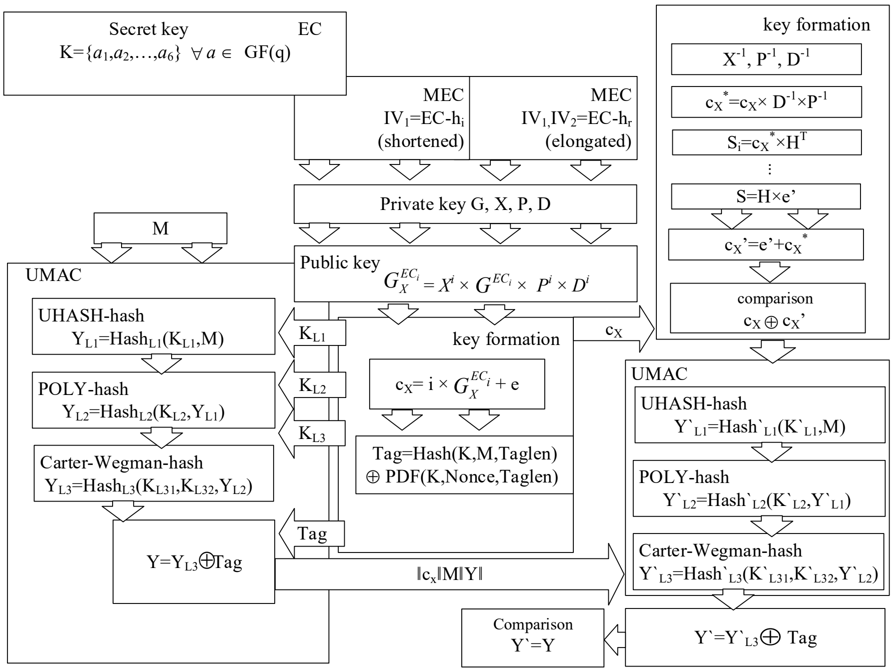
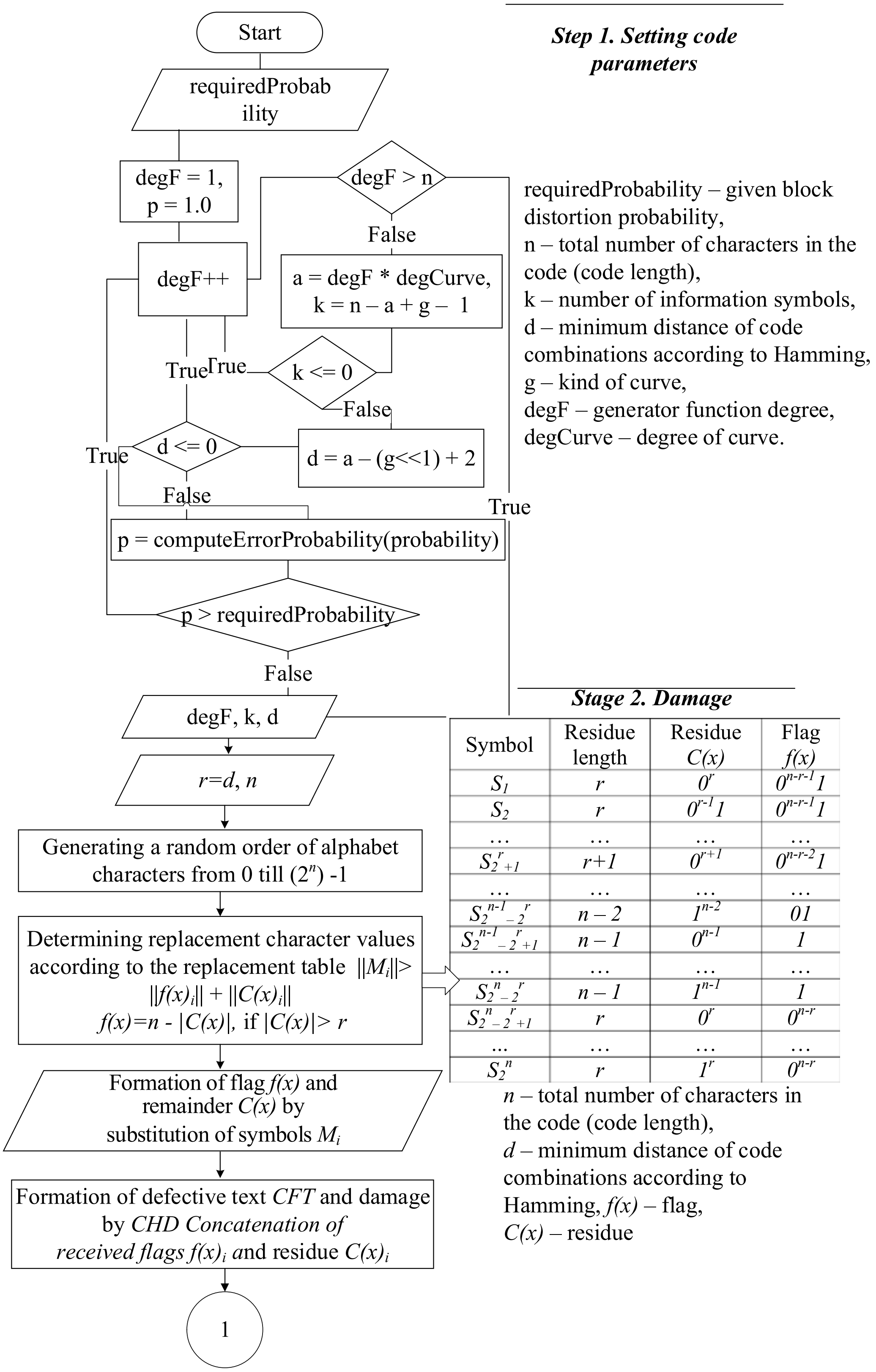

Secret key
EC
K={a1,a2,…,a6} ∀ a ∈ GF(q)
MEC IV1=EC-hi (shortened)
MEC IV1,IV2=EC-hr (elongated)
Private key G, X, P, D
M
UMAC
Public key $G_X^{EC_i} = X^i \times G^{EC_i} \times P^i \times D^i$
UHASH-hash YL1=HashL1(KL1,M)
POLY-hash YL2=HashL2(KL2,YL1)
Carter-Wegman-hash YL3=HashL3(KL31,KL32,YL2)
KL1
KL2
KL3
key formation
cX
$c_X = i \times G_X^{EC_i} + e$
Tag=Hash(K,M,Taglen) ⊕ PDF(K,Nonce,Taglen)
Tag
Y=YL3⊕Tag
‖cX‖M‖Y‖
key formation
X-1, P-1, D-1
cX*=cX× D-1×P-1
Si=cX*×HT
⋮
S=H×e'
cX'=e'+cX*
comparison cX ⊕ cX'
UMAC
UHASH-hash Y`L1=Hash`L1(K`L1,M)
POLY-hash Y`L2=Hash`L2(K`L2,Y`L1)
Carter-Wegman-hash Y`L3=Hash`L3(K`L31,K`L32,Y`L2)
Y`=Y`L3⊕ Tag
Comparison Y`=Y

Fig. 9. Scheme of message transmission from the sender to the recipient and checking the integrity of the message received by comparing codograms and hash codes using McEliece CCC on MEC

In [50], it was proposed to use the McEliece crypto-code construction using damaged codes as a mechanism for forming the pseudo-random pad of the third UMAC layer. The main ideas of damaged cryptography are proposed in [52, 53]. This approach allows the use of a hybrid CCC (HCCC) construction with multichannel cryptography and provides the maximum substrate formation rate with the required level of security. In addition, it allows reducing energy costs (construction of the HCCC over GF (2^4–2^6) while maintaining the required level of hash code strength. Fig. 10, 11 show an algorithm for using McEliece HCCC to form the pad.


Step 1. Setting code parameters
Start
requiredProbability
degF = 1,
p = 1.0
degF > n
False
degF++
a = degF * degCurve,
k = n – a + g – 1
True
True
k <= 0
False
True
d <= 0
d = a – (g<<1) + 2
False
True
p = computeErrorProbability(probability)
p > requiredProbability
False
degF, k, d
r=d, n
Generating a random order of alphabet characters from 0 till (2^n) -1
Determining replacement character values according to the replacement table ||M_i||> ||$f(x)_i$|| + ||$C(x)_i$||
$f(x)=n$ - |$C(x)$|, if |$C(x)$|> r
Formation of flag $f(x)$ and remainder $C(x)$ by substitution of symbols M_i
Formation of defective text CFT and damage by CHD Concatenation of received flags $f(x)_i$ and residue $C(x)_i$
1
Stage 2. Damage

requiredProbability – given block distortion probability,
n – total number of characters in the code (code length),
k – number of information symbols,
d – minimum distance of code combinations according to Hamming,
g – kind of curve,
degF – generator function degree,
degCurve – degree of curve.

Symbol	Residue length	Residue $C(x)$	Flag $f(x)$
S_1	r	0^r	$0^{n-r-1}1$
S_2	r	$0^{r-1}1$	$0^{n-r-1}1$
…	…	…	…
S_{2^r+1}	$r+1$	0^{r+1}	$0^{n-r-2}1$
…	…	…	…
$S_{2^{n-1}-2^r}$	$n-2$	1^{n-2}	01
$S_{2^{n-1}-2^r+1}$	$n-1$	0^{n-1}	1
…	…	…	…
$S_{2^n-2^r}$	$n-1$	1^{n-1}	1
$S_{2^n-2^r+1}$	r	0^r	0^{n-r}
...	…	…	…
S_{2^n}	r	1^r	0^{n-r}

n – total number of characters in the code (code length),
d – minimum distance of code combinations according to Hamming, $f(x)$ – flag,
$C(x)$ – residue

Fig. 10. Formation of the pad based on McEliece HCCC with damaged codes

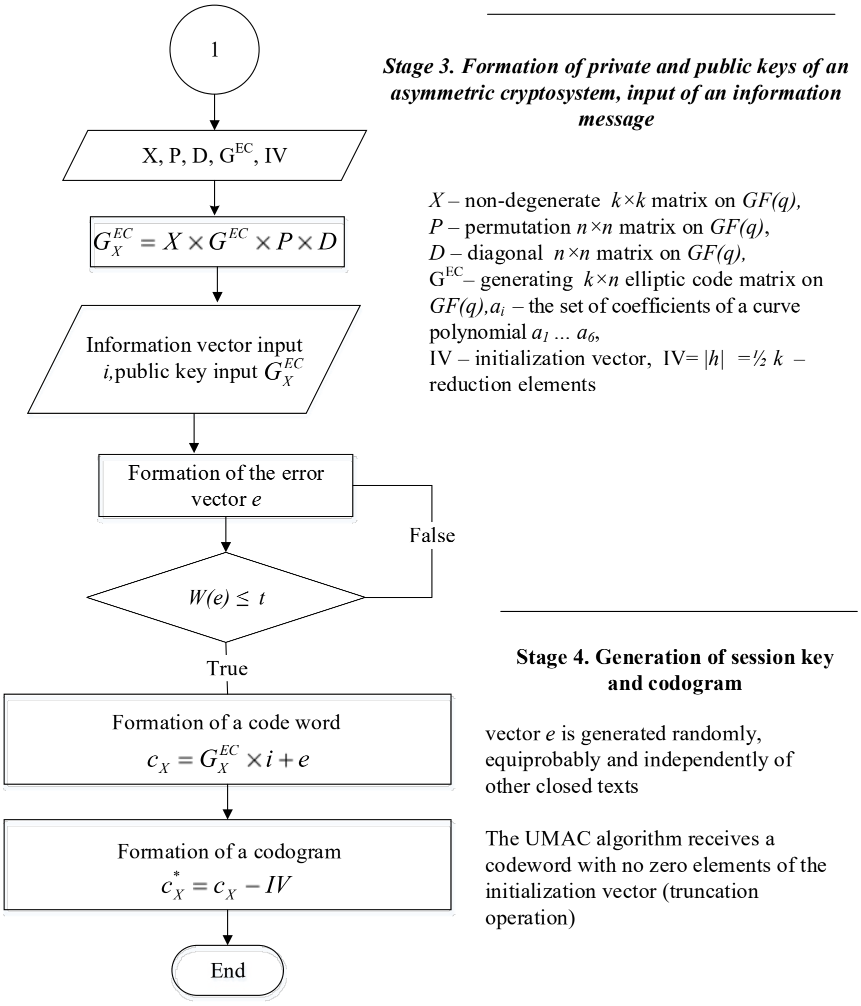

1
X, P, D, G^{EC}, IV
$G_X^{EC} = X \times G^{EC} \times P \times D$
Information vector input i, public key input G_X^{EC}
Formation of the error vector e
False
$W(e) \le t$
True
Formation of a code word $c_X = G_X^{EC} \times i + e$
Formation of a codogram $c_X^* = c_X - IV$
End
Stage 3. Formation of private and public keys of an asymmetric cryptosystem, input of an information message
X – non-degenerate $k \times k$ matrix on $GF(q)$,
P – permutation $n \times n$ matrix on $GF(q)$,
D – diagonal $n \times n$ matrix on $GF(q)$,
G^{EC} – generating $k \times n$ elliptic code matrix on $GF(q)$, a_i – the set of coefficients of a curve polynomial $a_1 \dots a_6$,
IV – initialization vector, IV= $|h|$ =½ k – reduction elements
Stage 4. Generation of session key and codogram
vector e is generated randomly, equiprobably and independently of other closed texts
The UMAC algorithm receives a codeword with no zero elements of the initialization vector (truncation operation)

Fig. 11. Formation of the pad based on McEliece HCCC with damaged codes

The proposed approach will make it possible to form the pseudo-random pad with a further reduction in computing resources by reducing the computational field of elliptic curves based on damage (reduction or lengthening of the code sequence).

7. Development of algorithms for assessing the strength of hash codes of the modified UMAC algorithm based on evaluating criteria of universality and strict universality of hash function classes

The proposed algorithms for assessing the strength of hash codes of the modified UMAC algorithm (statistical study of collisional properties of the generated elements $h(x)$) are based on empirical assessment of the maximum number of keys (hashing rules) for which:

1) for arbitrary $x_1, x_2 \in A$, $x_1 \neq x_2$, equality holds:

$$h(x_1)=h(x_2); \qquad (1)$$

2) for arbitrary $x_1 \in A$ and $y_1 \in B$ equality holds:

$$h(x_1)=y_1; \qquad (2)$$

3) for arbitrary $x_1, x_2 \in A$, $x_1 \neq x_2$ and $y_1, y_2 \in B$ equalities hold:

$$h(x_1)=y_1,\ h(x_2)=y_2. \qquad (3)$$

The assessment according to the first criterion corresponds to checking the fulfillment of the condition for the universal class of hash functions, the assessment according to the second and third criterion corresponds to the conditions for the strictly universal class of hash functions.

To estimate the above equalities, the following notation is introduced [54]:

$$n_1(x_1,x_2)=\left|\left\{h \in H : h(x_1)=h(x_2)\right\}\right|,$$

$$x_1,x_2 \in A,\ x_1 \neq x_2,$$

$$n_2(x_1,y_1)=\left|\left\{h \in H : h(x_1)=y_1\right\}\right|,\ x_1 \in A,\ y_1 \in B;$$

$$n_3(x_1,x_2,y_1,y_2)=\left|\left\{h \in H : h(x_1)=y_1, h(x_2)=y_2\right\}\right|,$$

$$x_1,x_2 \in A,\ x_1 \neq x_2,$$

$$y_1,y_2 \in B.$$

The first indicator $n_1(x_1,x_2)$ characterizes the number of hashing rules (MAC formation rules), under which for the given $x_1, x_2 \in A$, $x_1 \neq x_2$, equality (1) holds, i. e. the number of keys for which there is a collision (MAC match) for two input sequences x_1 and x_2.

The second indicator $n_2(x_1,y_1)$ characterizes the number of hashing rules (MAC formation rules), under which for the given $x_1 \in A$, $y_1 \in B$ equality (2) holds, i. e., the number of keys, for which the hash code value (MAC) y_1 for the input sequence x_1 does not change.

The third indicator $n_3(x_1,x_2,y_1,y_2)$ characterizes the number of hashing rules (MAC formation rules). For given $x_1,x_2 \in A,\ x_1 \neq x_2,\ y_1,y_2 \in B$, equality (3) holds, i. e., the number of keys for which for two input sequences x_1 and x_2 their corresponding hash values (MAC) y_1 and y_2 do not change.

Since the number of keys must not exceed their corresponding values $P_{\text{кол}} \cdot |H|$, $|H|/|B|$ and $P_{\text{кол}} \cdot |H|$, the maximum number of such keys for each of the considered set of elements is of interest.

To conduct research, it is necessary to determine the maximums of these values, and then compare the results with the number $\left(P_{\text{кол}} \cdot |H|\right)$ (for the first criterion), with the number $\left(|H|/|B|\right)$ (for the second criterion) and the number $\left(P_{\text{кол}} \cdot |H|\right)$ (for the third criterion).

Thus, as statistical indicators for assessing collision properties, use [54]:

– mathematical expectations $m(n_1)$, $m(n_2)$ and $m(n_3)$ of maxima of the number of hashing rules (MAC formation rules) at which equalities (1), (2) and (3) are satisfied, respectively;

– dispersion $D(n_1)$, $D(n_2)$ and $D(n_3)$, characterizing the dispersion of the values of the number of hashing rules (MAC formation rules), at which equalities (1), (2) and (3) are satisfied, with respect to their mathematical expectations $m(n_1)$, $m(n_2)$ and $m(n_3)$, respectively.

The assessment of collisional properties according to the given criteria of universality and strict universality is carried out in the average statistical sense. So, when setting up an experiment, a limited set of elements $x_1, x_2 \in A$, $x_1 \neq x_2$ and their corresponding hash images (MAC) $y_1, y_2 \in B$ is used, considering the relevant results as a sample from the general population.

The natural estimate $\tilde{m}$ for the mathematical expectation m of a random variable X is the arithmetic mean of its observed values X_i (or statistical mean) [19]:

$$\tilde{m} = \frac{1}{N}\sum_{i=1}^{N} X_i,$$

where N – number of implementations of a random variable X.

The estimate of the variance $\tilde{D}$ of a random variable X is determined by the expression:

$$\tilde{D} = \frac{1}{N-1}\sum_{i=1}^{N}\left(X_i - \tilde{m}\right)^2.$$

By virtue of the central limit theorem of probability theory for large values of the number of implementations N, the arithmetic mean will have a distribution close to normal, with a mathematical expectation [19]:

$$m[\tilde{m}] \approx \tilde{m}$$

and standard deviation

$$\sigma[\tilde{m}] \approx \frac{\sigma}{\sqrt{N}},$$

where σ – standard deviation of the estimated parameter.

Moreover, the probability that the estimate $\tilde{m}$ deviates from its mathematical expectation by less than ε (confidence level) is equal to [19]:

$$P\left(|\tilde{m} - m| < \varepsilon\right) \approx 2\Phi\left(\frac{\varepsilon}{\sigma[\tilde{m}]}\right), \tag{4}$$

where Φ(x) – Laplace function, defined by the expression:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_0^x e^{-\frac{t^2}{2}}\,\mathrm{d}t. \tag{5}$$

When conducting experimental studies of collision properties, it was proposed to use the methods of statistical testing of hypotheses and mathematical statistics [18, 19]:

1. From the general population of a random variable X, a sample is formed as follows:

– for the average estimate of the mathematical expectation $m(n_1)$ and variance $D(n_1)$, the maximum $n_1(x_1, x_2)$ is used as a random variable for which the equality $h(x_1)=h(x_2)$ holds, hence the sample of size N: X_1, X_2,..., X_N will be formed by selecting N sets, each of which contains M pairs of elements $x_1, x_2 \in A$, $x_1 \neq x_2$ and estimated as $n_1(x_1, x_2)$, i. e. the total volume of formed pairs of elements $x_1,x_2 \in A$, $x_1 \neq x_2$ will be NM;

– for the average estimate of $m(n_2)$ and $D(n_2)$, the maximum $n_2(x_1,y_1)$ is used as a random variable for which the equality $y_1=h(x_1)$ holds, hence the sample of size N: X_1, X_2, ..., X_N will be formed by selecting N sets, each of which contains M pairs of elements $x_1 \in A$, $y_1 \in B$ and estimated as $n_2(x_1,y_1)$. The total volume of formed pairs of elements $x_1 \in A$, $y_1 \in B$, will be NM;

– for the average estimate of $m(n_3)$ and $D(n_3)$, the maximum $n_3(x_1, x_2, y_1, y_2)$ is used as a random variable for which the equalities $y_1=h(x_1)$ and $y_2=h(x_2)$ hold, hence the sample of size N: X_1, X_2, ..., X_N will be formed by selecting N sets, each of which contains M quadruples of elements $x_1, x_2 \in A$, $x_1 \neq x_2$, $y_1,y_2 \in B$ and estimated as $n_3(x_1,x_2,y_1,y_2)$, the total volume of formed quadruples will be NM.

2. In experimental studies of the collisional properties of hashing, the arithmetic mean $\tilde{m}(n_i)$ of observed maximum values n_i and variance $\tilde{D}(n_i),\ i=1,2,3.$ are estimated.

3. The reliability of the average estimates obtained is substantiated as follows. The accuracy ε is recorded and the values of the Laplace function are calculated, which, in accordance with the confidence probability, will give the corresponding confidence probabilities:

$$P\left(|\tilde{m}(n_i) - m(n_i)| < \varepsilon\right) \approx 2\Phi\left(\frac{\varepsilon}{\sigma\left[\tilde{m}(n_i)\right]}\right),$$

$$\sigma\left[\tilde{m}(n_i)\right] \approx \frac{\sqrt{\tilde{D}(n_i)}}{\sqrt{N}},\ i=1,2,3.$$

When the problem is inversely stated, i. e., for a fixed confidence level P_σ for a sample size of N, the confidence interval is determined as follows:

$$\tilde{m}(n_i) - t_\rho \cdot \sigma\left[\tilde{m}(n_i)\right] < m(n_i) < \tilde{m}(n_i) + t_\rho \cdot \sigma\left[\tilde{m}(n_i)\right],$$

$$i=1,2,3,$$

where t_ρ – the root of the equation $2\Phi\left(t_\rho\right) = P_{\text{д}}$.

The algorithm for checking hash codes for compliance with the rules of the universal class of hash functions is shown in Fig. 12.

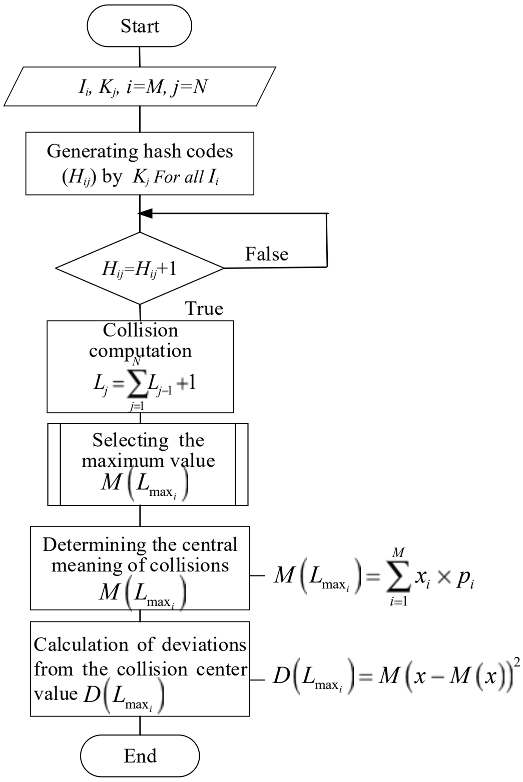

Start
I_i, K_j, $i=M$, $j=N$
Generating hash codes (H_{ij}) by K_j For all I_i
$H_{ij}=H_{ij}+1$
False
True
Collision computation
$L_j=\sum_{j=1}^{N} L_{j-1}+1$
Selecting the maximum value $M\left(L_{\max_i}\right)$
Determining the central meaning of collisions $M\left(L_{\max_i}\right)$
$M\left(L_{\max_i}\right)=\sum_{i=1}^{M} x_i \times p_i$
Calculation of deviations from the collision center value $D\left(L_{\max_i}\right)$
$D\left(L_{\max_i}\right)=M\left(x-M(x)\right)^2$
End

Fig. 12. Algorithm for checking hash codes for compliance with the requirements of a universal class of hash functions

The implementation of the algorithm can be described by the following steps.

Step 1. Forming input messages.

Step 2. Forming keys.

Step 3. Forming hash codes H_{ij} for each input message I_i using keys K_j.

Step 4. Performing a sequential comparison of the obtained hash codes H_{ij} by the same key K_j on all incoming messages among themselves on the basis of the following condition: if the hash values match ($H_{ij}=H_{ij+1}$), which indicates the occurrence of a collision (L_j), then 1 is added to the collision counter:

$$L_j=\sum_{j=1}^{N} L_{j-1}+1,$$

where $L_{j\text{-}1}$ – the previous value of the collision counter on the *j*-th key.

Step 5. Within one message on all keys we choose the maximum value of collisions $L_{\max_i}$.

Step 6. Calculating the arithmetic mean or central value of the maximum number of collisions by finding their mathematical expectation $M\left(L_{\max_i}\right)$:

$$M\left(L_{\max_i}\right)=\sum_{i=1}^{M} x_i \times p_i.$$

Step 7. Calculating the value of the scatter of the maximum values of collisions around their central value by finding the variance $\left(D\left(L_{\max_i}\right)\right)$ on the maximum number of collisions:

$$D\left(L_{\max_i}\right)=M\left(x-M(x)\right)^2.$$

The algorithm for checking hash codes for compliance with the requirements of a strictly universal class of hash functions according to the first criterion is shown in Fig. 13.

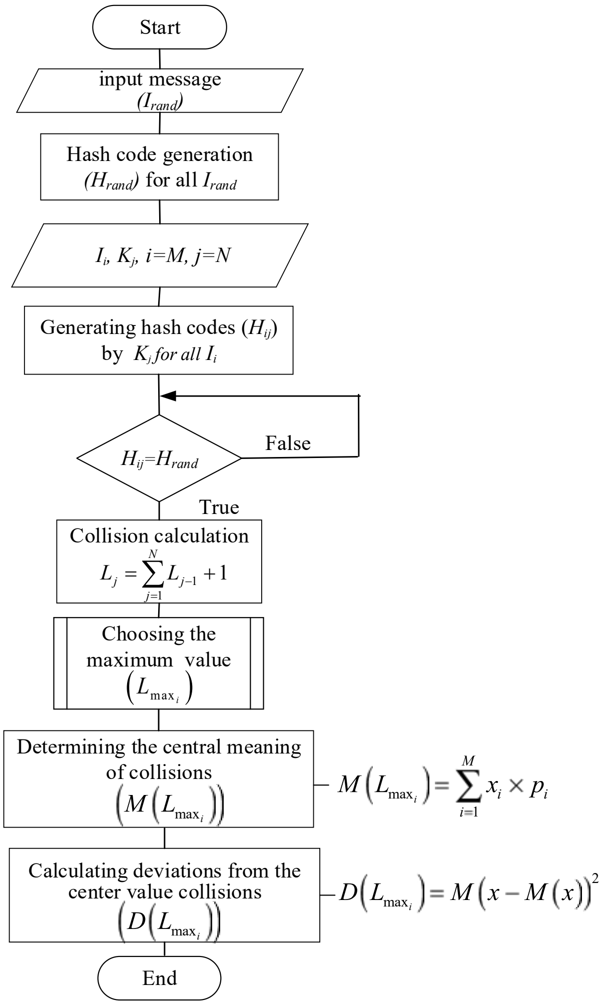

Start
input message (I_{rand})
Hash code generation (H_{rand}) for all I_{rand}
I_i, K_j, $i=M$, $j=N$
Generating hash codes (H_{ij}) by K_j for all I_i
$H_{ij}=H_{rand}$
False
True
Collision calculation
$L_j=\sum_{j=1}^{N} L_{j-1}+1$
Choosing the maximum value $\left(L_{\max_i}\right)$
Determining the central meaning of collisions $\left(M\left(L_{\max_i}\right)\right)$
$M\left(L_{\max_i}\right)=\sum_{i=1}^{M} x_i \times p_i$
Calculating deviations from the center value collisions $\left(D\left(L_{\max_i}\right)\right)$
$D\left(L_{\max_i}\right)=M\left(x-M(x)\right)^2$
End

Fig. 13. Algorithm for checking hash codes for compliance with the requirements of a strictly universal class of hash functions according to the first criterion

The implementation of the algorithm can be described by the following steps:

Step 1. Forming one random incoming message $I_{rand.}$

Step 2. Forming a hash code H_{rand} of the random message I_{rand}.

Step 3. Forming incoming messages.

Step 4. Forming keys.

Step 5. Forming hash codes H_{ij} for each incoming message I_i using keys K_j.

Step 6. Performing a sequential comparison of the obtained hash codes H_{ij} by the same key K_j on all incoming messages with a hash code H_{rand} of the random message I_{rand} on the basis of the following condition:

– if the hash values match ($H_{ij}=H_{rand}$), which indicates the occurrence of a collision (L_j), then 1 is added to the collision counter:

$$L_j = \sum_{j=1}^{N} L_{j-1} + 1.$$

Step 7. Within one message for all keys, select the maximum value of the collision $L_{\max_i}$.

Step 8. Calculating the arithmetic mean or central value of the maximum number of collisions by finding their mathematical expectation $M\left(L_{\max_i}\right)$:

$$M\left(L_{\max_i}\right) = \sum_{i=1}^{M} x_i \times p_i.$$

Step 9. Calculating the value of the scatter of the maximum values of collisions around their central value by finding the variance $\left(D\left(L_{\max_i}\right)\right)$ on the maximum number of collisions:

$$D\left(L_{\max_i}\right) = M\left(x - M(x)\right)^2.$$

The algorithm for checking hash codes for compliance with the requirements of a strictly universal class of hash functions according to the second criterion is shown in Fig. 14.

The implementation of the algorithm can be described by the following steps:

Step 1. Forming two different random incoming messages I_{rand1} and I_{rand2}.

Step 2. Forming hash codes H_{rand1} and H_{rand2} for each of the messages I_{rand1} and I_{rand2}.

Step 3. Forming incoming messages.

Step 4. Forming keys.

Step 5. Forming hash codes H_{ij} for each incoming message I_i using keys K_j.

Step 6. Performing a sequential comparison of the obtained hash codes H_{rand1} and H_{rand2} by the same key K_j for all incoming messages with hash codes of two random messages:

– if the hash values match ($H_{ij}=H_{rand1}$ or $H_{ij}=H_{rand2}$), which indicates the occurrence of a collision (L_j), then 1 is added to the collision counter:

$$L_j = \sum_{j=1}^{N} L_{j-1} + 1.$$

Step 7. Within one message for all keys, selecting the maximum value of the collision $L_{\max_i}$.

Step 8. Calculating the arithmetic mean or central value of the maximum number of collisions by finding their mathematical expectation $M\left(L_{\max_i}\right)$.

$$M\left(L_{\max_i}\right) = \sum_{i=1}^{M} x_i \times p_i.$$

Step 9. Calculating the value of the scatter of the maximum values of collisions around their central value by finding the variance $\left(D\left(L_{\max_i}\right)\right)$ by the maximum number of collisions:

$$D\left(L_{\max_i}\right) = M\left(x - M(x)\right)^2.$$

Thus, the proposed algorithm allows evaluating not only the fulfillment of the criteria of universality and strict universality of the obtained MAC code, but also the level of stability.

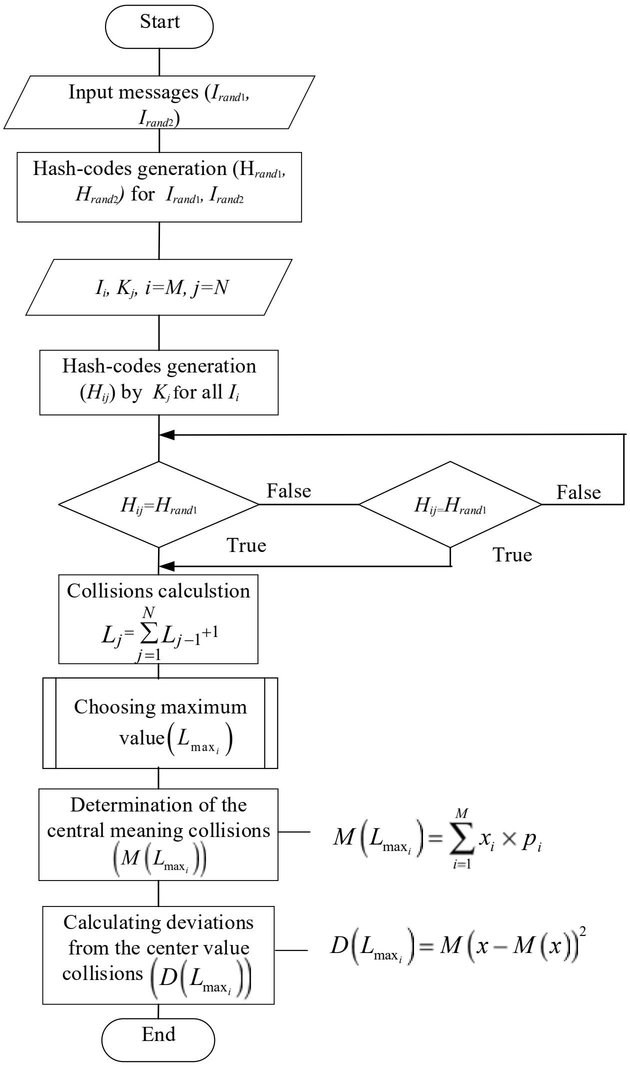

Start
Input messages (I_{rand1}, I_{rand2})
Hash-codes generation (H$_{rand1}$, H_{rand2}) for I_{rand1}, I_{rand2}
I_i, K_j, $i=M$, $j=N$
Hash-codes generation (H_{ij}) by K_j for all I_i
$H_{ij}=H_{rand1}$
False
$H_{ij}=H_{rand1}$
False
True
True
Collisions calculstion
$L_j=\sum_{j=1}^{N} L_{j-1}+1$
Choosing maximum value$\left(L_{\max_i}\right)$
Determination of the central meaning collisions $\left(M\left(L_{\max_i}\right)\right)$
$M\left(L_{\max_i}\right)=\sum_{i=1}^{M} x_i \times p_i$
Calculating deviations from the center value collisions $\left(D\left(L_{\max_i}\right)\right)$
$D\left(L_{\max_i}\right)=M\left(x-M(x)\right)^2$
End

Fig. 14. Algorithm for checking hash codes for compliance with the requirements of a strictly universal class of hash functions according to the second criterion

8. Discussion of the results of the practical implementation of the modified UMAC on CCC

To ensure the validity of the proposed approach, consider a practical example of the implementation of the modified UMAC cascade algorithm on CCC with EC, MEC and DC.

To model the implementation of the modified UMAC algorithm on algebrogeometric (EC, MEC) and damaged codes based on McElliese CCC (HCCC), the following input data are used:

– data block length (bytes) – 32;

– secret key length – 32 bytes;

– transmitted plain text I array length – 3bytes;

– pseudo-random key sequence length (number of subkeys) –1,027;

– transmitted plaintext (k-bit information vector over $GF(q)$) – 11;

– secret error vector e=00000200 (session key);

– masking matrix (user private key – KR_i);

– non-degenerate k×k matrix – X;

– n×n permutation matrix $P;$

– diagonal matrix D;

– generating matrix G;

The generating matrix (user public key – KU_i) – G_X^{EC}, G_X^{MEC}, depending on the McEliece CCC (HCCC).

The conversion of the codogram values according to the *MV2* algorithm is given in Table 3.

Table 3

Conversion of the codogram values according to the *MV2* algorithm

Binary representation of the symbol	Remainder length	Remainder $C(\mathbf{x})$	Flag $F(x)$
000	2	00	1
001	2	01	1
010	2	10	1
011	2	11	1
100	2	00	0
101	2	01	0
110	2	10	0
111	2	11	0

Next is the formation of a hash code on the basis of the modified UMAC algorithm with various mechanisms of forming the pseudo pad on the basis of CCC with EC, MEC, DC.

I. Creating a hash code based on the UMAC algorithm.

Step 1. Forming the first layer.

The value of the first-level hash function UHASH-hash Y_{L1I} is calculated by the formula:

$$Y_{L1I} = Hash_{L1}\left(K_{L1I}, I\right).$$

For the formation of K_{L1I} it is presented as a sequence of keys from four-byte blocks:

$$K_{L1I} = K_{1I} \| K_{2I} \| \ldots \| K_{nI},$$

where $\|$ – concatenation (connection) of the lines corresponding to subkeys.

The amount of data of the subkey:

$$n = \frac{1{,}072}{32} = 33.5 \approx 33 => i = 1,2,\ldots,33.$$

Since $T_i = \text{Index}\|i$, then for the first level *Index*=1, =>T_i=K_{iI}:

K_{1I}=00000001 000000001, K_{2I}=00000001 000000010,
K_{3I}=00000001 000000011, K_{4I}=00000001 000000100,
K_{5I}=00000001 000000101, K_{6I}=00000001 000000110,
K_{7I}=00000001 000000111, K_{8I}=00000001 000001000,
K_{9I}=00000001 000001001, K_{10I}=00000001 00001010,
K_{11I}=00000001 00001011, $K_{12I=}$00000001 00001100,
K_{13I}=00000001 00001101, K_{14I}=00000001 00001110,
$K_{15I=}$00000001 00001111, K_{16I}=00000001 00010000,
K_{17I}=00000001 00010001, $K_{18I=}$00000001 00010010,
K_{19I}=00000001 00010011, K_{20I}=00000001 00010100,
K_{21I}=00000001 00010101, K_{22I}=00000001 00010110,
K_{23I}=00000001 00010111, K_{24I}=00000001 00011000,
K_{25I}=00000001 00011001, K_{26I}=00000001 00011010,
K_{27I}=00000001 00011011, K_{28I}=00000001 00011100,
K_{29I}=00000001 00011101, K_{30I}=00000001 00011110,
K_{31I}=00000001 00011111, K_{32I}=00000001 00100000,
K_{33I}=00000001 00100001.

Based on the length M of the input message, the number of blocks is equal to T=1, so, the number of subkeys at this level is the same, therefore,

$$K_{L1I} = T_1 = 0000000100000001.$$

The hash function values of this layer are calculated by the following formula:

$$Y_{L1I} = (I + K_{L1I}) \bmod 32 =$$
$$= (0100110 + 10000001) \bmod 32 = 111.$$

Step 2. Forming the second layer.

Because the length M is less than 1024 bytes, this level of hashing will not be performed, and we will perform calculations using the third-level hash code.

Step 3. Forming the third layer.

The number of subkeys for K_{L31}:

$$n = \frac{64 \times 4}{32} = 8 => i = 1,2,3,4,5,6,7,8.$$

Therefore, for the formation of K_{L31I} it can be presented as a sequence of keys of eight four-byte blocks:

$$K_{L31I} = K_{1I} \| K_{2I} \| K_{3I} \| K_{4I} \| K_{5I} \| K_{6I} \| K_{7I} \| K_{8I}.$$

For the third level *Index* =3, =>T_i=K_{iI}:

K_{1I}=00000011 00000001, K_{2I}=00000011 00000010,
K_{2I}=00000011 00000010, K_{2I}=00000011 00000010,
K_{3I}=00000011 00000011, K_{4I}=00000011 00000100,
K_{5I}=00000011 00000101, K_{6I}=00000011 00000110,
K_{7I}=00000011 00000111, K_{8I}=00000011 00001000.

The number of subkeys for K_{L32I}:

$$n = \frac{4 \times 4}{32} = 0.5 \approx 1 => i = 1.$$

For the formation of K_{L32I} it can be presented as a key sequence of 1 four-byte subblock:

$$K_{L32I} = K_{1I}.$$

For the third level *Index*=4, =>T_i=K_{1I}:

K_{1I}=00000100 00000001.

The value of the hash function of the third layer is calculated by the following formula:

$$Y_{L3I} = \left(\left(Y_{L1I} \bmod \left(2^{36} - 5\right)\right) \bmod 2^{32}\right) \operatorname{xor} Y_{L32I} =$$
$$= \left(\left(\left(\left(I + K_{1I}\right) \bmod 32\right) \bmod \left(2^{36} - 5\right)\right) \bmod 2^{32}\right) \operatorname{xor} Y_{L32I},$$

$$Y_{L3I} =$$
$$= \left(\left(11 \bmod \left(2^{36} - 5\right)\right) \bmod 2^{32}\right) \operatorname{xor} 00000100\,00000001 =$$
$$= 10000000010.$$

II. Forming key data of modified McEliece crypto-code constructions with EC, MEC, DC [7, 8, 19, 37].

Step 1. The public key for McEliece CCC on EC is formed by the expression:

$$G_X^{EC} = X \times G \;\; \times P \times D.$$

The public key for McEliece CCC on MEC is formed by the expression:

$$G_X^{MEC} = X \times G^{EC} \times P \times D.$$

The public key for McEliece HCCC on DC is formed by the expression:

$$G_X^{HMEC} = X \times G^{MEC} \times P \times D.$$

The result of the formation of the public key is the generating matrix of the algebrogeometric code:

$$G_X^{EC} = G_X^{MEC} = G_X^{HMEC} = \begin{bmatrix} 2 & 1 & 3 & 0 & 1 & 1 & 1 & 0 \\ 0 & 2 & 2 & 2 & 2 & 0 & 3 & 2 \end{bmatrix}.$$

Step 2. The cryptogram (codegram) formed from the information message I is a vector of length n, which is calculated by the following formula:

– for McEliece CCC on EC:

$$C_X^* = I \times G_X^{EC} \oplus e.$$

– for McEliece CCC on MEC:

$$C_X^* = I \times G_X^{MEC} \oplus e.$$

– for McEliece HCCC on DC:

$$C_X^* = I \times G_X^{HMEC} \oplus e.$$

The result of the generated vector is the following value:

$$C_X^* = 23023322.$$

Step 3. Forming the initialization vector IV=00100000, which determines the location of the code sequence reduction/extension symbols:

– for shortened MEC – $C_x^* = 2323322$;

– for elongated MEC – $C_x^* = 23123322$.

Step 4. Forming damage and damaged code, through damage based on the *MV2* codeword algorithm:

– source text (word):

– for shortening –

$$C_x^* = 2323322_{10} = 010011\ 000010011011\ 010\ 010_2.$$

Step 5. Transmitting damage (flag) by one of the channels to the recipient and sending of the damaged code (remainder) by another channel to the recipient:

– for shortened MEC:

– sending damage to the channel –

$$C_x^* = 010011\ 000010011011\ 010\ 010_2 = 45818_{10};$$

– sending the damaged code to the channel – $F(x)$= $=11111111_2=255_{10}$;

– for elongated MEC:

– sending damage to the channel –

$$C_x^* = 010011\ 000010011011\ 010\ 010_2 = 2495337_{10};$$

– sending the damaged code to the channel – $F(x)$= $=111111111_2=511_{10}$.

III. Forming the pseudo-random pad is carried out on the basis of the proposed modified McEliece crypto-code constructions with EC, MEC, DC:

$$Pad = PDF(K, Nonce, Taglen) = \\ = PDF(0106, 8, 4) = 1101010.$$

IV. Hash code generation:

$$Tag = UMAC(K, I, Nonce, Taglen) = \\ = Hash(K, I, Taglen) \oplus \\ \oplus PDF(K, Nonce, Taglen) = Y_{L3M} \oplus Pad = \\ = 10000000010 \oplus 1101010 = 10001101100.$$

$$Y = Y_{L3M} \oplus Tag.$$

$$Y = 10000000010 \oplus \\ \oplus 10001101100 = 1101110 = 110_{10}.$$

VI. DSA-based verification (Digital Signature Algorithm).

When using the modified UMAC algorithm with McEliece CCC (HCCC) in the DSA standard, the sender uses a private key (KR_i), the recipient uses the public key KU_i to verify the digital signature.

The proposed approach provides the required level of efficiency of hash code generation (online) taking into account the growth of computing resources and the amount of data transmitted. Table 4 shows the results of studies of energy costs for the practical implementation of the proposed algorithms for forming the pad based on the use of McEliece CCC (HCCC) on algebrogeometric (EC, MEC) and damaged codes.

Table 4

Dependence of the software implementation of McEliece CCC (HCCC) on the field strength (number of group operations)

Cryptosystems	2^5	2^6	2^7	2^8	2^9	2^{10}
CCC on EC	10018042	18048068	32847145	47489784	63215578	**82467897**
CCC on shortened MEC	10007947	**17787431**	28595014	44079433	61974253	79554764
CCC on elongated MEC	11156138	**18561228**	33210708	48297112	65171690	84051337
HCCC on shortened MEC	**7900315**	14892945	25565274	42279183	58963778	76564173
HCCC on elongated MEC	**7905257**	14682411	25595014	42116327	58468143	75474764

Analysis of Table 4 confirms the theoretical calculations of the reduction of costs for the practical implementation of McEliece CCC (HCCC) on EC (MEC), DC while maintaining the required indicators of the hash code stability in the modified UMAC algorithm. Table 5 presents the results of studies of the statistical properties of the proposed methods based on the *NIST STS 822* package, which confirm the stability of the proposed McEliece CCC (HCCC) on EC (MEC), DC [55].

Table 5

Results of statistical security study

Cryptosystems	Number of tests in which more than 99 % of the sequences were tested	Number of tests in which more than 96 % of the sequences were tested	Number of tests in which less than 96 % of the sequences were tested
McEliece CCC on EC	149 (78.83 %)	189 (100 %)	0 (0 %)
McEliece MCCC on shortened MEC	151 (79.89 %)	189 (100 %)	0 (0 %)
McEliece MCCC on elongated MEC	152 (80.42 %)	189 (100 %)	0 (0 %)
HCCC on elongated MEC	153 (80.95 %)	189 (100 %)	0 (0 %)
HCCC on shortened MEC	155 (82 %)	189 (100 %)	0 (0 %)

Thus, the presented results confirm the possibility of implementing this approach to the formation of a modified cascade hashing algorithm using crypto-code constructs as a mechanism to ensure the required level of stability in the post-quantum period. In this case, the use of a particular design depends on the computing power and/or platform on the basis of which the software is developed.

A further area of research is the practical confirmation of statistical estimates of the properties of universality and strict universality of the modified UMAC algorithm with the formation of the pad on the McEliece CCC (HCCC).

9. Conclusions

1. In the context of increasing computing resources, expanding the scope of the digital economy and e-banking services, one of the conditions for providing security services is the search for new and/or modification of known methods. The growth and integration of modern threats, their hybridity and synergy require the introduction of strict criteria for special mechanisms to ensure authenticity. Among the known algorithms for generating MAC codes, universal hash functions occupy a special place. However, their use without additional hash code encryption does not provide the required level of robustness.

2. Analysis of the formation of hash codes based on the cascading application of universal hash functions showed that the use of a block symmetric AES algorithm as a mechanism of pseudo-random pad in the UMAC algorithm does not allow “preserving” the universality of the hash code. And the improvement of the mechanism based on modular arithmetic (MASH-2 algorithm) does not meet the requirements for the efficiency of transformations. Thus, it is proposed to use one of the promising areas – crypto-code constructions based on algebrogeometric and damaged codes.

3. Modifications of construction of the cascade hashing algorithm on the basis of using CCC on EC, MEC, DC are proposed. This approach allows “preserving” the versatility and all the advantages of this class of hash functions, as well as provides the required parameters for efficiency and durability in the conditions of post-quantum cryptography and the emergence of a full-scale quantum computer.

4. The proposed algorithms for assessing the resilience of hash codes of the modified UMAC algorithm on McEliece crypto-code constructions with EC, MEC, DC allow not only assessing the fulfillment of universality and strict universality criteria, but also provide assessments of hash code resilience to modern threats. The proposed algorithms for the statistical study of collision properties generated by MAC codes are based on an empirical estimate of the maximum number of keys (hashing rules).

References

1. Evseev, S., Kotz, H., Korol, O. (2015). Analysis of the legal framework for the information security management system of the NSMEP. Eastern-European Journal of Enterprise Technologies, 5 (3 (77)), 48–59. doi: https://doi.org/10.15587/1729-4061.2015.51468
2. Evseev, S., Abdullayev, V. (2015). (2015). Monitoring algorithm of two-factor authentication method based on passwindow system. Eastern-European Journal of Enterprise Technologies, 2 (2 (74)), 9–16. doi: https://doi.org/10.15587/1729-4061.2015.38779
3. Aktual’nye kiberugrozy – 2017: trendy i prognozy (2018). Positive technologies. Available at: https://www.ptsecurity.com/ru-ru/research/analytics/cybersecurity-threatscape-2017/
4. Aktual’nye kiberugrozy – 2018. Trendy i prognozy (2019). Positive technologies. Available at: https://www.ptsecurity.com/ru-ru/research/analytics/cybersecurity-threatscape-2018/
5. Aktual’nye kiberugrozy: itogi 2019 goda (2020). Positive technologies. Available at: https://www.ptsecurity.com/ru-ru/research/analytics/cybersecurity-threatscape-2019/
6. Yevseiev, S., Hryhorii, K., Liekariev, Y. (2016). Developing of multi-factor authentication method based on niederreiter-mceliece modified crypto-code system. Eastern-European Journal of Enterprise Technologies, 6 (4 (84)), 11–23. doi: https://doi.org/10.15587/1729-4061.2016.86175
7. Yevseiev, S., Korol, O., Kots, H. (2017). Construction of hybrid security systems based on the crypto-code structures and flawed codes. Eastern-European Journal of Enterprise Technologies, 4 (9 (88)), 4–21. doi: https://doi.org/10.15587/1729-4061.2017.108461

8. Yevseiev, S., Tsyhanenko, O., Ivanchenko, S., Aleksiyev, V., Verheles, D., Volkov, S. et. al. (2018). Practical implementation of the Niederreiter modified cryptocode system on truncated elliptic codes. Eastern-European Journal of Enterprise Technologies, 6 (4 (96)), 24–31. doi: https://doi.org/10.15587/1729-4061.2018.150903
9. Sidel'nikov, V. M. (2002). Kriptografiya i teoriya kodirovaniya. Materialy konferentsii "Moskovskiy universitet i razvitie kriptografii v Rossii".
10. Bartock, M., Cichonski, J., Souppaya, M., Smith, M., Witte, G., Scarfone, K. (2016). Guide for cybersecurity event recovery. NIST. doi: https://doi.org/10.6028/nist.sp.800-184
11. Security requirements for cryptographic modules. Available at: https://csrc.nist.gov/csrc/media/publications/fips/140/2/final/documents/fips1402.pdf
12. Cichonski, J., Franklin, J. M., Bartock, M. (2017). Guide to LTE security. NIST. doi: https://doi.org/10.6028/nist.sp.800-187
13. Lohachab, A., Lohachab, A., Jangra, A. (2020). A comprehensive survey of prominent cryptographic aspects for securing communication in post-quantum IoT networks. Internet of Things, 9, 100174. doi: https://doi.org/10.1016/j.iot.2020.100174
14. Petrenko, K., Mashatan, A., Shirazi, F. (2019). Assessing the quantum-resistant cryptographic agility of routing and switching IT network infrastructure in a large-size financial organization. Journal of Information Security and Applications, 46, 151–163. doi: https://doi.org/10.1016/j.jisa.2019.03.007
15. Hryshchuk, R., Yevseiev, S., Shmatko, A. (2018). Construction methodology of information security system of banking information in automated banking systems. Vienna: Premier Publishing s. r. o., 284. doi: https://doi.org/10.29013/r.hryshchuk_s.yevseiev_a.shmatko.cmissbiabs.284.2018
16. Gorbenko, Y., Ganzya, R. (2014). Analysis of the possibility of quantum computers and quantum computings for cryptanalysis of modern cryptosystems. Eastern-European Journal of Enterprise Technologies, 1 (9 (67)), 8–16. doi: https://doi.org/10.15587/1729-4061.2014.19897
17. Korol, O. G., Parhuts, L. T., Evseev, S. P. (2013). Method of forming cascade mac-code using modular transformation. Nauchnye vedomosti Belgorodskogo gosudarstvennogo universiteta. Seriya: Ekonomika. Informatika, 15 (158), 147–157.
18. Kuznetsov, A. A., Korol, O. G., Evseev, S. P. (2012). Studying collision characteristics of authentication codes of messages UMAC. Applied Radio Electronics, 11 (2), 171–183.
19. Evseev, S., Yokhov, O., Korol, O. (2013). Data Hashing in Information Systems. Kharkiv: Vyd. KhNEU, 312.
20. Kuznetsov, O. O., Horbenko, Yu. I., Kiyan, A. S., Uvarova, A. O., Kuznetsova, T. Yu. (2018). Porivnialni doslidzhennia ta analiz efektyvnosti hibrydnoi kodovoi kryptosystemy. Radyotekhnyka, 195, 61–69. Available at: http://nbuv.gov.ua/UJRN/rvmnts_2018_195_9
21. Marquez-Corbella, I., Tillich, J.-P. (2016). Using Reed-Solomon codes in the (U | U + V) construction and an application to cryptography. 2016 IEEE International Symposium on Information Theory (ISIT). doi: https://doi.org/10.1109/isit.2016.7541435
22. Kapshikar, U., Mahalanobis, A. (2018). A Quantum-Secure Niederreiter Cryptosystem using Quasi-Cyclic Codes. Proceedings of the 15th International Joint Conference on e-Business and Telecommunications. doi: https://doi.org/10.5220/0006843005060513
23. Abidin, A. (2012). On Security of Universal Hash Function Based Multiple Authentication. Lecture Notes in Computer Science, 303–310. doi: https://doi.org/10.1007/978-3-642-34129-8_27
24. Handschuh, H., Preneel, B. (2008). Key-Recovery Attacks on Universal Hash Function Based MAC Algorithms. Advances in Cryptology – CRYPTO 2008, 144–161. doi: https://doi.org/10.1007/978-3-540-85174-5_9
25. Abouhogail, R. A. (2011). New multicast authentication protocol for entrusted members using advanced encryption standard. The Egyptian Journal of Remote Sensing and Space Science, 14 (2), 121–128. doi: https://doi.org/10.1016/j.ejrs.2011.11.003
26. Carter, J. L., Wegman, M. N. (1979). Universal classes of hash functions. Journal of Computer and System Sciences, 18 (2), 143–154. doi: https://doi.org/10.1016/0022-0000(79)90044-8
27. Stinson, D. R. (1994). Combinatorial techniques for universal hashing. Journal of Computer and System Sciences, 48 (2), 337–346. doi: https://doi.org/10.1016/s0022-0000(05)80007-8
28. Sarvate, D. G. Seberry, J. (1986) Encryption methods based on combinatorial designs. Available at: https://ro.uow.edu.au/cgi/viewcontent.cgi?article=2034&context=infopapers
29. Khalimov, G. Z. (2013). Strongly universal hashing. Applied Applied Radio Electronics, 12 (2), 220–224.
30. Simmons, G. J. (1988). An Impersonation-Proof Identity Verification Scheme. Lecture Notes in Computer Science, 211–215. doi: https://doi.org/10.1007/3-540-48184-2_17
31. Simmons, G. J. (1985). Authentication Theory/Coding Theory. Lecture Notes in Computer Science, 411–431. doi: https://doi.org/10.1007/3-540-39568-7_32
32. Kuznetsov, A. A., Korol', O. G., Bos'ko, V. V. (2011). Model of forming of codes of authentification of messages with the use of universal hash functions. Systemy obrobky informatsiyi, 3 (93), 117–125.
33. Alekseev, M. O. (2014). Protection against algebraic manipulations based on a scalar product operation. Problemy informatsionnoy bezopasnosti. Komp'yuternye sistemy, 2, 47–53.
34. Alekseev, M. O., Mironchikov, E. T. (2011). Ob obnaruzhenii oshibok s pomoshch'yu nelineynyh kodov. Nauchnaya sessiya GUAP, 1, 40–43.
35. Black, J., Halevi, S., Krawczyk, H., Krovetz, T., Rogaway, P. (1999). UMAC: Fast and Secure Message Authentication. Lecture Notes in Computer Science, 216–233. doi: https://doi.org/10.1007/3-540-48405-1_14

36. Ferguson, N., Schneier, B. (2004). Practical Cryptography. Moscow: Izdatel'skiy dom "Vil'yams", 432.
37. Kuznetsov, A. A., Pushkarev, A. I., Svatovskiy, I. I., Shevtsov, A. V. (2016). Nesimmetrichnye kriptosistemy na algebraicheskih kodah dlya postkvantovogo perioda. Radiotehnika, 186, 70–90.
38. Krovetz, T., Rogaway, P. (2001). Fast Universal Hashing with Small Keys and No Preprocessing: The PolyR Construction. Information Security and Cryptology – ICISC 2000, 73–89. doi: https://doi.org/10.1007/3-540-45247-8_7
39. Krovetz, T. (2000). Software-Optimized Universal Hashing and Message Authentication. University of California Davis, 269.
40. Krovetz, T. (Ed.). (2006). UMAC: Message Authentication Code using Universal Hashing. doi: https://doi.org/10.17487/rfc4418
41. Korol, O. G. (2015). Evaluation of the computational complexity of some hash functions. Systemy obrobky informatsiyi, 4, 105–110.
42. Krovetz, T., Black, J., Halevi, S., Hevia, A., Krawczyk, H., Rogaway, P. (2000). UMAC. Primitive submitted to NESSIE, 157–160.
43. Bosselaers, A., Govaerts, R., Vandewalle, J. (1996). Fast Hashing on the Pentium. Lecture Notes in Computer Science, 298–312. doi: https://doi.org/10.1007/3-540-68697-5_23
44. Final report of European project number IST-1999-12324, named New European Schemes for Signatures, Integrity and Encryption. Version 0.15 (beta). Springer-Verlag.
45. Evseev, S., Korol, O., Ohurtsov, V. (2014). Advanced algorithm UMAC based modular transformations. Eastern-European Journal of Enterprise Technologies, 1 (9 (67)), 16–23. doi: https://doi.org/10.15587/1729-4061.2014.20130
46. Yevseiev, S., Kots, H., Minukhin, S., Korol, O., Kholodkova, A. (2017). The development of the method of multifactor authentication based on hybrid cryptocode constructions on defective codes. Eastern-European Journal of Enterprise Technologies, 5 (9 (89)), 19–35. doi: https://doi.org/10.15587/1729-4061.2017.109879
47. Yevseiev, S. (2017). The use of damaged codes in crypto code systems. Systemy obrobky informatsiyi, 5, 109–121. Available at: http://nbuv.gov.ua/UJRN/soi_2017_5_17
48. Havrylova, A., Korol, O., Milevskyi, S. (2019). Mathematical model of authentication of a transmitted message based on a mceliece scheme on shorted and extended modified elliptic codes using UMAC modified algorithm. Cybersecurity: Education, Science, Technique, 5, 40–51. doi: https://doi.org/10.28925/2663-4023.2019.5.4051
49. Yevseiev, S., Havrylova, A. (2020). Improved umac algorithm with crypto-code mceliece's scheme. Modern problems of computer science and IT-education. Vienna, 79–92. doi: https://doi.org/10.29013/melnikk.shmatkoo.mpcsie.2020.352
50. Korol, O., Havrylova, A., Yevseiev, S. (2019). Practical UMAC algorithms based on crypto code designs. Przetwarzanie, transmisja I bezpieczenstwo informacji. Vol. 2. Bielsko-Biala: Wydawnictwo naukowe Akademii Techniczno-Humanistycznej w Bielsku-Bialej, 221–232.
51. Yevseiev, S., Rzayev, K., Korol, O., Imanova, Z. (2016). Development of mceliece modified asymmetric crypto-code system on elliptic truncated codes. Eastern-European Journal of Enterprise Technologies, 4 (9 (82)), 18–26. doi: https://doi.org/10.15587/1729-4061.2016.75250
52. Mishchenko, V. A., Vilanskiy, Yu. V. (2007). Ushcherbnye teksty i mnogokanal'naya kriptografiya. Minsk: Entsiklopediks, 292.
53. Mishchenko, V. A., Vilanskiy, Yu. V., Lepin, V. V. (2007). Kriptograficheskiy algoritm MV 2. Minsk: Entsiklopediks, 176.
54. Korol', O. G. (2010). Issledovanie kollizionnyh svoystv kodov autentifikatsii soobshcheniy UMAC. Systemy obrobky informatsiyi. Problemy i perspektyvy rozvytku IT-industriyi, 7 (88), 221.
55. Rukhin, A., Sota, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S. et. al. (2000). A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST. doi: https://doi.org/10.6028/nist.sp.800-22