International Journal of 3D Printing Technologies and Dijital Industry
Uluslararası 3B Yazıcı Teknolojileri ve Dijital Endüstri Dergisi

DEVELOPMENT OF AUTHENTICATION CODES OF MESSAGES ON THE BASIS OF UMAC WITH CRYPTO-CODE MCELIECE'S SCHEME

Serhii Yevseiev[a], Olha Korol[a], Alla Havrylova[a*]

[a]Simon Kuznets Kharkiv National University of Economics, Department of Cyber Security and Information Technology, UKRAINE

[*]*Corresponding Author: Alla.Havrylova@hneu.net*

(Received: 22.04.2019**; Revised:** 02.07.2019**; Accepted:** 15.07.2019)

ABSTRACT

The development of decentralized systems and blockchain technology have expanded the range of cryptocurrency-based banking services. The main difference from the hierarchical structures of the organizations of the banking sector (national and commercial banks) is the formation of valid nodes ensuring the confirmation of transactions based on the checking and verification of digital signatures and MAC codes. The Bitcoin protocols use the SHA-256 algorithm to form MAC codes, however, the rapid growth of the system leads to significant time costs not only for mining, but also for validation of the formed blocks. The further development of decentralized systems, increase the number of wall-distributors and full nodes forces us to look for new ways to solve a temporary problem based on using different approaches to providing authentication in decentralized systems. The paper discusses the algorithm for generating UMAC message authentication codes using a McEliece's crypto-code scheme based on the use of universal hashing functions. A reduced UMAC model (mini-UMAC) and a method for statistical analysis of the collision characteristics of the generated message authentication codes are proposed. Using the reduced UMAC model, collisional characteristics of authentication codes are investigated, it is shown that the use of cryptographic transformation (using the AES algorithm) at the final stage of UMAC leads to a violation of the universal hashing characteristics.

Keywords: mini-UMAC with a McEliece's crypto-code scheme, authentication, universal function, authenticity codes, AES algorithm.

1. INTRODUCTION

The development of decentralized systems, blockchain technology develops a range of banking services in which huge amounts of data are stored and circulated. To ensure the authenticity (identity) of the formed blocks in the blockchain technology systems, as a rule, they use the MAC – Message Authentication Code – (MDC – Manipulation Detection Code) codes used as an effective mechanism for ensuring the integrity and authenticity of information [1–7]. However, used cryptographic algorithms, including hashing functions in cryptocurrency systems, are increasingly becoming the target of cyber-terrorists and hackers. The analysis showed that the UMAC (Message Authentication Code using Universal Hashing) algorithm [5, 7] selected during the European NESSIE competition, has the highest computational efficiency for generate authentication codes using the family of universal hash functions [8, 9]. The number of collisions (collisions) of generated hash images for each entered universal hash key does not exceed a predetermined value, and UMAC cryptographic strength is provided at the level of the selected cryptoalgorithms (according to the specification, AES encryption algorithm is recommended). However, researches [10–12] of using a cryptoalgorithms for collisional

characteristics of UMAC message authentication codes showed that the provision of universal hashing characteristics in such a multilayered structure is not justified.

2. LITERATURE REVIEW

In works[13, 14], an analysis of the cryptographic strength of competing algorithms for the American standard SHA-3 showed that the winner of the competition, the Keccak algorithm does not provide the required level of security and is not considered by the developers of authentication protocols in decentralized systems as an alternative to the SHA-256 algorithm. However, the growth of users, wall allocators and full nodes in the Bitcoin system, increased functionality based on the introduction of smart contracts require fast and cryptographically stable hashing algorithms.

The analysis of the European competition NESSIE showed that the algorithms based on the use of universal hash functions provide the maximum speed of transformations and allow to evenly distribute collisions across the entire set of hash codes, which in turn can be applicable as authenticators in large databases.

Generation of message authentication codes using the UMAC algorithm. One of the first versions of the algorithm for generating message authentication codes using universal hashing (UMAC – Message Authentication Code using Universal Hashing) was presented in work [1]. Later, after some refinement [2 – 4], the UMAC algorithm was presented in the final report of the European competition NESSIE – New European Schemes for Signatures, Integrity and Encryption [5]. One of the latest electronic versions of the UMAC algorithm is available in electronic format [6]. The most detailed components of the UMAC are presented in the dissertation [7].

Let us consider the general scheme for the formation of message authenticity codes using the UMAC algorithm, analyze the main analytical relations that describe the internal structure and the transformations used in the formation of message authenticity codes.

2.1. The general scheme of the formation of message authentication codes using the UMAC algorithm

Tag the message authentication code (we denote it as) according to the specification of the UMAC algorithm is formed by calculating the following function:

$$Tag = UMAC(K, M, Nonce, Taglen) = Y \oplus Pad,$$

where: K – is the secret key whose length *Keylen* equal to the standard length of the secret key of the used symmetric block cipher (the UMAC specification recommends using the AES encryption algorithm (FIPS-197), in this case the length of the secret key {16, 24, 32}belongs to the set of valid byte values M 2^{67} an information message to be authenticated, presented in the format of a string array with a dimension from one to 2^{64} bits (*Nonce* bytes); M – non-repeating (for all input informational messages *Taglen*) eight-byte number; {4, 8, 12, 16} – an integer number from the set of valid values *Tag*, that specifies the length of the message authentication code $Hash(K, M, Taglen)$ in bytes; M – key universal hash function of an information message K using a secret key $PDF(K, Nonce, Taglen)$; *Pad* – the function of the formation of a pseudo-random pad (*Nonce*) on the entered value K and the secret key $\oplus$; "$Y = Hash(K, M, Taglen)$" – bitwise addition (XOR) of the result of the key message hash and the formed pad

$$Pad = PDF(K, Nonce, Taglen), \text{ i.e.}$$

$$Tag = Hash(K, M, Taglen) \oplus PDF(K, Nonce, Taglen).$$

The length of the hash code *Y*, the pad *Pad* and the code *Tag* belong to the set of valid values of {32, 64, 96, 128} bits. These fixed values *Taglen* correspond to the case of the formation of codes of authenticity of messages UMAC – 32, UMAC – 64, UMAC – 96 or UMAC – 128, respectively.

Consider the formation of hash codes and the pad $Pad = PDF(K, Nonce)$.

2.2. The scheme of the formation of hash codes $Y = Hash(K, M, Taglen)$

Calculating the values $Hash(K, M, Taglen)$ of key universal hashing functions of the information message using M K the secret key is performed in three stages (three levels (layers) of key hashing are used) $Hash_{L1}$, $Hash_{L2}$ and $Hash_{L3}$, respectively. The second level of hashing $Hash_{L2}$ is performed only if the length of the hashed message exceeds 1024 bytes.

The length of the hash code Y is a multiple of 32 bits, its value for any length $Y = Hash(K, M, Taglen)$ is formed by combining (concatenating) several (from one to four) sequences Y_{L3_i}

$$Y = Hash(K, M, Taglen) = Y_{L3_1} \| Y_{L3_2} \| .. \| Y_{L3_{It}}, \; It = Taglen/4,$$

where: Y_{L3_i} – is the result of multi-level hashing of the message M on the i-th iteration using the appropriate keys, $i=1, 2, ..., I_t$.

Consider the formation of a hash code Y_{L3_i} on the i-th iteration. To do this, we denote the result of multi-level hashing on an arbitrary i-th iteration as follows:

$$Y_{L3_i} = Y_{L3} = Hash_{L3}(K_{L3_1}, K_{L3_2}, Hash_{L2}(K_{L2}, Hash_{L1}(K_{L1}, M))),$$

where: $Hash_{L1}(K_{L1}, M)$, $Hash_{L2}(K_{L2}, Y_{L1})$ and $Hash_{L3}(K_{L3_1}, K_{L3_2}, Y_{L2})$ – are key hashing functions of the first, second, and third level that are controlled and dependent on the iteration number secret keys K_{L1}, K_{L2}, K_{L3_1}, K_{L3_2}, respectively.

Key sequences K_{L1}, K_{L2}, K_{L3_1}, K_{L3_2}, are formed by the entered secret key of K length $Keylen$ bytes using a special function $KDF(K, Index, Numbyte)$ (Key-Derivation Function – KDF), where $Index$ and $Numbyte$ are two positive integers not exceeding 2^{64}.

The first level of hashing splits an array-string M of size up to 2^{64} bytes into blocks M_i of 1024 bytes with the subsequent conversion of each block by a function $NH(K_{L1}, M_i)$. The obtained results $Hash_{L1_i} = NH(K_{L1}, M_i)$ concatenated (combined) into a string $Y_{L1} = Hash_{L1}(K_{L1}, M)$, that is 128 times shorter than the information sequence. This line is the result of the first level hashing:

$$Y_{L1} = Hash_{L1}(K_{L1}, M) = NH(K_{L1}, M_0) \| NH(K_{L1}, M_1) \| ... \| NH(K_{L1}, M_{n-1}),$$

where: $n = \left\lceil \frac{Length(M)}{1024} \right\rceil$, $\lceil x \rceil$ – the integer part of number x, $Length(M)$ – information message byte length M.

The result of the function $Hash_{L1_i} = NH(K_{L1}, M_i)$ is calculated according to the following rule. The information block M_i is divided into four-byte sub-blocks so:

$$M_i = M_{i_1} \| M_{i_2} \| .. \| M_{i_t},$$

where: $t = \left\lceil \frac{Length(M_i)}{4} \right\rceil$. In this case $t = \left\lceil \frac{1024}{4} \right\rceil = 256$.

Similarly, a key sequence K_{L1} is represented as four-byte subblock sequences:

$$K_{L1} = K_{L1_1} \| K_{L1_2} \| .. \| K_{L1_t}.$$

After that (taking the initial state $Hash_{L1_i} = 0$) for all $j=1,9,17,...,t-7$ the following operations are performed:

$$Hash_{L1_i} = Hash_{L1_i} +_{64} ((M_{i_{j+0}} +_{32} K_{L1_{j+0}}) \times_{64} (M_{i_{j+4}} +_{32} K_{L1_{j+4}})),$$

$$Hash_{L1_i} = Hash_{L1_i} +_{64} ((M_{i_{j+1}} +_{32} K_{L1_{j+1}}) \times_{64} (M_{i_{j+5}} +_{32} K_{L1_{j+5}})),$$

$$Hash_{L1_i} = Hash_{L1_i} +_{64} ((M_{i_{j+2}} +_{32} K_{L1_{j+2}}) \times_{64} (M_{i_{j+6}} +_{32} K_{L1_{j+6}})),$$

$$Hash_{L1_i} = Hash_{L1_i} +_{64} ((M_{i_{j+3}} +_{32} K_{L1_{j+3}}) \times_{64} (M_{i_{j+7}} +_{32} K_{L1_{j+7}})),$$

where: +64, +32 – are addition operations modulo 2^{64} and 2^{32}, respectively ×64 – the operation of multiplication modulo 2^{64}.

Papers [1 – 7] shows, that the considered key hashing function NH belongs to the class of universal hashing functions.

The second level of hashing $Poly$, discussed in detail in papers [1 – 7]. The result of the work of this level is the calculation of the hash code $Y_{L2} = Hash_{L2}(K_{L2}, Y_{L1}) = Poly(Wordbits, Maxwordrange, k, M_P)$, i.e. the second level hashing input is given a string $Y_{L1} = Hash_{L1}(K_{L1}, M)$.

The polynomial hash function uses as input data:
$Wordbits \in [64,128]$; $Maxwordrange$ – a positive integer less than $2^{Wordbits}$; k – key-dependent K_{L2} integer from the range $[0,...,prime(Wordbits)-1]$, $prime(x)$ – the largest prime number, the least 2^x; $M_P = Y_{L1} = Hash_{L1}(K_{L1}, M)$ – data for polynomial hashing.

According to the specification of the UMAC algorithm, as $prime(x)$ the following constants are used: $prime(36) = 2^{36} - 5$, $prime(64) = 2^{64} - 59$, $prime(128) = 2^{128} - 159$. The bit length M_P is denoted by $Bytelength(M_P)$. Depending on the length M_P the following features are used in the implementation of the second level of hashing:

- if the length of the received data M_P does not exceed 2^{17} bytes, then the polynomial hashing, $Poly$ is performed with parameters $Wordbits = 64$; $Maxwordrange = 2^{64} - 2^{32}$; $k = k64$ – a string formed by the first eight bytes of the key K_{L2} and a special eight-byte mask;
- if the length of the received data M_P exceeds 2^{17} bytes (but does not exceed 2^{64} bytes), then the first $2^{Poly}(64,2^{64}-2^{32},k64,M_P)$ bytes of data are processed by the polynomial hashing function $Poly$, a and the remaining data bytes are processed by the function $Wordbits = 128$ with parameters $Wordbits = 128$; $Maxwordrange = 2^{128} - 2^{96}$; $k = k128$ – the string formed by the last 16 bytes of the key and the special 16 bytes mask.

Hashable data M_P divided into blocks by $Wordbytes = Wordbits/8$ bytes:

$$M_P = M_{P_1} \| M_{P_2} \| .. \| M_{P_n},$$

where: $n = Bytelength(M_P)/Wordbytes$.

The result of the hash is the value of the polynomial function

$$Y_{L2} = \left(M_{P_n} + kM_{P_{n-1}} + ... + k^{n-1}M_{P_1} + k^n\right) \bmod(p),$$

which is calculated by an iterative procedure (for each $i = 1,2,...,n$):

$$Poly_i = \left(kPoly_{i-1} + M_{P_i}\right) \bmod(p),\ Poly_0 = 1,\ p = prime(Wordbits)$$

Using Horner's method

$$M_{P_n} + kM_{P_{n-1}} + ... + k^{n-1}M_{P_1} + k^n = (((k + M_{P_1})k + M_{P_2})k + ... + M_{P_{n-1}})k + M_{P_n}.$$

The calculated hash value $Y_{L2} = Poly_n$ is an integer in range. $[0,..., prime(Wordbits)-1]$.
Considered the polynomial key hashing function $Poly(Wordbits, Maxwordrange, k, M_P)$ belongs to the class of universal hashing functions [1–7].

The third level of hashing $Hash_{L3}\left(K_{L3_1}, K_{L3_2}, Y_{L2}\right)$ is performed on the result of a polynomial hash and converts data up to 16 bytes sent to its input into a hash code Y of fixed length 32 bits.
The source data of the third level of hashing are two key sequences K_{L3_1} and K_{L3_2} that length are 64 and 4 bytes, respectively, as well as an input 16 bytes sequence Y_{L2}.
Hashable data Y_{L2} and key sequence K_{L3_1} are evenly divided into eight blocks, each of which is represented as an integer Y_{L2_i} and $K_{L3_{1_i}}$, $i = 1,2,...8$.
The hash value Y_{L3} is calculated as follows:

$$Y_{L3} = \left(\left(\left(\sum_{i=1}^{m} Y_{L2_i} K_{L3_{1_i}}\right) \bmod(prime(36))\right) \bmod(2^{32})\right) xor(K_{L3_2}),$$

where: $(x)xor(y)$ – where is the operation of "exclusive OR" on the values *x* and *y*.
The considered key hashing function $Y_{L3} = Hash_{L3}\left(K_{L3_1}, K_{L3_2}, Y_{L2}\right)$ belongs to the class of universal hashing functions; its properties are studied in detail in [1–7].

2.3. KDF: Key-Derivation Function

A special function $KDF(K, Index, Numbyte)$ is designed to form sequences of pseudo-random data bits, which are used at various levels of generating message authentication codes as key data of the corresponding hashing functions.

As the source data for the generation of key pseudo-random sequences, the secret key *K* whose length is $Keylen$ bytes and two positive integers $Index$ and $Numbyte$, whose value does not exceed. 2^{64}.

To generate pseudo-random key sequences, a block symmetric cipher is used. Let us denote the procedure for encrypting a block of data *T* of length $Blocklen$ using a secret key *K* of length $Keylen$ bytes in the form of a certain function $Enchiper(K,T)$. Then the procedure for the formation of a pseudo-random key sequence $K' = KDF(K, Index, Numbyte)$ can be represented as the following iterative (for each $i = 1,2,...,n$) transformation:

$$T_i = Index \| i,$$
$$K'_i = Enchiper(K, T_i),$$
$$K' = K'_1 \| K'_2 \| .. \| K'_n,$$

where: $n = \left\lceil \frac{Numbyte}{Blocklen} \right\rceil$, $\lceil x \rceil$ – is the integer path of x, $a \| b$ – concatenation of strings *a* and *b*.

The generated sequence of pseudo-random key data bits has K' length of $Numbyte$ bytes, that is a multiple of the block byte length $Blocklen$.

2.4. Pseudo-random pad formation scheme (PDF: Pad-Derivation Function)

Function $PDF(K, Nonce, Taglen)$ is intended for the formation of a pseudo-random pad Pad, used in the final stage of the formation of the message authentication code.

As the source data, the secret key *K* whose length is $Keylen$ bytes and a non-repeating (for each input informational messages *M*) eight-byte number $Nonce$, as well as an integer $Taglen$, specifying the size (length in bytes) of the generated authentication code Tag.

The procedure for forming a pseudo-random pad $Pad = PDF(K, Nonce, Taglen)$ is to form a subkey

$$K' = KDF(K, Index, Numbyte),\ Index = 0,\ Numbyte = Keylen,$$

using the above procedure for the formation of sequences of pseudo-random key bits and the encryption of values $Nonce$ on the generated subkey K' K', i.e.:

$$Pad = PDF(K, Nonce, Taglen) = Enchiper(KDF(K, 0, Keylen), Nonce).$$

The procedure for the formation of a pseudo-random pad Pad is constructed so that the resulting value Pad has a length of $Taglen$ bytes, regardless of the values $Blocklen$ and $Nonce$.

Thus, the considered scheme for the formation of message authentication codes UMAC uses a multi-level design of universal hashing $Hash(K, M, Taglen)$ and the procedure for the formation of a pseudo-random pad Pad. The use of universal hashing makes it possible to ensure equiprobability of the formation of hash images for the entire set of key data used, which is the basis for the proof of the security of the algorithm [1 – 7]. The formation of a pseudo-random pad with a cryptographically robust algorithm (for example, using the AES block symmetric cipher) ensures the cryptographic strength of the UMAC algorithm at the persistence level of the cryptographic algorithm used [5, 7]. Consequently, the considered scheme of UMAC formation has potentially high efficiency indicators, in Figure 1 presents the results of studies of the performance of different versions of the UMAC algorithm in comparison with the known practical algorithms.

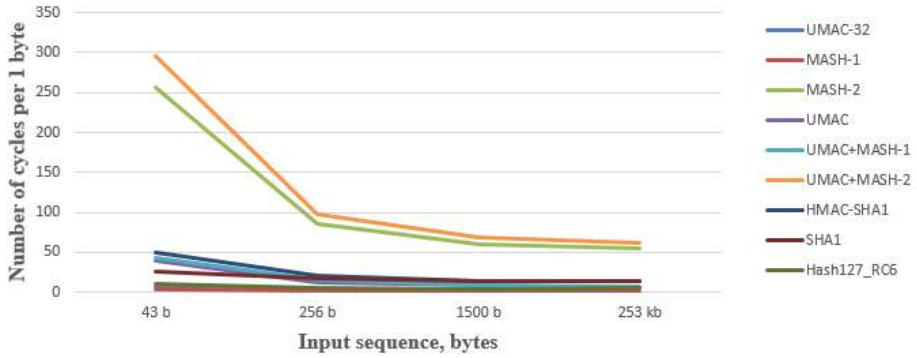

Figure 1. The results of the research performance algorithms UMAC.

At the same time, the algorithm of UMAC formation after applying the last layer of overlaying pseudo-random pads loses the property of "universality" of hashing; its collision properties significantly deteriorate.

3. MATERIAL AND METHOD

Minified model UMAC (mini-UMAC). The UMAC message authentication code generation algorithm uses several transformation layers in its structure; including a block symmetric cipher, (the AES cipher is recommended).

It was shown above that the scheme of forming UMAC codes consists of the following layers:

– three-level universal hashing for generating hash codes: $Y = Hash(K, M, Taglen)$;

– cryptographic transformations using a block symmetric cipher to form a pseudo-random pad $Pad = PDF(K, Nonce, Taglen)$;

– a final transformation to generate message authentication codes $Tag = UMAC(K, M, Nonce, Taglen) = Y \oplus Pad$.

Consider each layer of the UMAC message authentication code generation scheme for their scaling.

3.1. Mini-version of the three-level universal hash

We construct a mini-version of the three-level universal hash without changing the structure of algebraic transformations by simply reducing the dimension of the blocks and the processed data eight times.

The corresponding length of the hash-code $Y_{\min i}$ of the reduced model of the first layer will be a multiple of 4 bits; we will form its value by combining (concatenating) four sequences $Y_{\min i L3_i}$:

$$Y_{\min i} = Y_{\min i L3_1} \| Y_{\min i L3_2} \| Y_{\min i L3_3} \| Y_{\min i L3_4},$$

where: $Y_{\min i L3_i}$ – is the result of multi-level hashing of the message of the reduced model of the first layer mini-UMAC.

Consider the process of forming a single block $Y_{\min i L3_i}$ (we will not perform the second level of hashing in the reduced model):

$$Y_{\min i L3_i} = Y_{\min i L3} = Hash_{\min i L3}(K_{\min i L3_1}, K_{\min i L3_2}, Hash_{\min i L1}(K_{\min i L1}, M_{\min i})),$$

where: $K_{\min i L1}$, $K_{\min i L3_1}$, $K_{\min i L3_2}$ – mini-UMAC key sequences, $Hash_{\min i L1}$ and $Hash_{\min i L3}$ – minified versions of the first and third levels hashing, respectively.

At the first level, a 32-bit string array $M_{\min i}$ is transformed by a function $NH(K_{L1}, M_i)$. his string is the result of the first level hash: $Y_{\min i L1} = NH_{\min i}(K_{\min i L1}, M_{\min i})$.

The result of the function $NH_{\min i}(K_{\min i L1}, M_{\min i})$ is calculated according to the following rule. The information block $M_{\min i}$ is divided into eight four-bit sub-blocks:

$$M_{\min i} = M_{\min i_1} \| M_{\min i_2} \| \ldots \| M_{\min i_8}.$$

Similarly, a key sequence K_{L1} is represented as a sequence of eight four-bit sub-blocks: $K_{\min iL1} = K_{\min iL1_1} \| K_{\min iL1_2} \| .. \| K_{\min iL1_8}$.

Then (taking the initial state $Hash_{L1} = 0$) the following operations are performed:

$$Hash_{\min iL1} = Hash_{\min iL1} +_8 ((M_{\min i_0} +_4 K_{\min iL1_0}) \times_8 (M_{\min i_4} +_4 K_{\min iL1_4})),$$

$$Hash_{\min iL1} = Hash_{\min iL1} +_8 ((M_{\min i_1} +_4 K_{\min iL1_1}) \times_8 (M_{\min i_5} +_4 K_{\min iL1_5})),$$

$$Hash_{\min iL1} = Hash_{\min iL1} +_8 ((M_{\min i_2} +_4 K_{\min iL1_2}) \times_8 (M_{\min i_6} +_4 K_{\min iL1_6})),$$

$$Hash_{\min iL1} = Hash_{\min iL1} +_8 ((M_{\min i_3} +_4 K_{\min iL1_3}) \times_8 (M_{\min i_7} +_4 K_{\min iL1_7})),$$

where: $+_8$, $+_4$ – addition operations modulo 2^8 and 2^4, respectively; $\times_8$ – multiplication operation modulo 2^8.

As a result of calculations, an eight-bit value $Y_{\min iL1} = Hash_{\min iL1}$ is formed.

The third level of hashing converts the eight-bit data $Y_{\min iL1}$ submitted to its input into a hash code $Y_{\min iL3}$ of length 4 bits. The key sequences are $K_{\min iL3_1}$ and $K_{\min iL3_2}$ which length are 16 and 4 bits, respectively.

Hash able data $Hash_{\min iL1}$ and key sequence $K_{\min iL3_1}$ are evenly divided into four blocks, each of which is represented as an integer $Y_{\min iL2_i}$ and $K_{\min iL3_{1_i}}$, $i = 1,2,...,4$.

The hash value $Y_{\min iL3}$ is calculated as follows:

$$Y_{\min iL3} = \left(\left(\left(\sum_{i=1}^{4} Y_{\min iL2_i} K_{\min iL3_{1_i}} \right) \bmod(17) \right) \bmod(2^4) \right) xor(K_{\min iL3_2}),$$

where: $(x)xor(y)$ – is the operation of "exclusive OR" on the values *x* and *y*.

3.2. The mini-version of the block symmetric AES

The mini-version of the block symmetric AES cipher for the formation of a pseudo-random pad is discussed in detail in papers [14–18]. The simplest to implement is the mini version of the AES (Baby-Rijndael) cipher, which was proposed by K. Bergman [16]. We briefly consider this reduced cipher model and justify its use for the formation of a pseudo-random pad in mini-UMAC.

The size of the plaintext block is 16 bits, which we denote by four hexadecimal numbers h_0, h_1, h_2, h_3. Note that $h_0 h_0$ consists of the first four bits of the input stream. However, when h_0 h_0 is treated as a hexadecimal digit, the first bit is treated as a higher order bit. For example, the input block 1000 1100 0111 0001 will be represented by $h_0 = 8, h_1 = c, h_2 = 7, h_3 = 1$.

The key size is also 16 bits. Denote it as 4 hexadecimal numbers k_0, k_1, k_2, k_3.

Cipher steps are applied to a state – an array of 2×2 hexadecimal digits. However, for the operation $\tilde{\sigma}$ considered below, the state will be represented as an array of 8×2 bits, i.e. each hexadecimal digit will be treated as a column of 4 bits with a higher order bit on top.

The input block is loaded into the state by displaying h_0, h_1, h_2, h_3 in $\begin{bmatrix} h_0 & h_2 \\ h_1 & h_3 \end{bmatrix}$. For example, the input block 1000 1100 0111 0001 will be loaded as:

$\begin{bmatrix} 8 & 7 \\ c & 1 \end{bmatrix}$, where: 8×2 matrix will be $\begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 0 & 1 \\ 0 & 1 \\ 1 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}$.

Baby-Rijndael includes several rounds identical in structure (there are 4 by default). Before encryption, the input block is loaded into the state as described above and round keys are calculated. Encryption has a general structure:

$$E(a) = r_4 \circ r_3 \circ r_2 \circ r_1 \circ (a \oplus k_0),$$

where a denotes the state, k_0, k_1, k_2, k_3, k_4 – round keys and $r_i(a) = (t \cdot \tilde{\sigma}(S(a))) \oplus k_i$ except r_4, where the multiplication is omitted by t. At the end of the cipher, the state is loaded into a 16-bit block in the same order in which it was loaded.

Now we will describe the individual components of the cipher.

SubBytes: Operation S is a sample table that is applied to each hexadecimal state number:

$$\begin{bmatrix} h_0 & h_2 \\ h_1 & h_3 \end{bmatrix} \xrightarrow{S} \begin{bmatrix} S(h_0) & S(h_2) \\ S(h_1) & S(h_3) \end{bmatrix}?$$

where: the function S is given by the following table 1.

Table 1. Sample table that implements the Baby-Rijndael S-box.

x	*0*	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*a*	*b*	*c*	*d*	*e*	*f*
S(x)	*a*	*4*	*3*	*b*	*8*	*e*	*2*	*c*	*5*	*7*	*6*	*f*	*0*	*1*	*9*	*d*

ShiftRows: The operation $\tilde{\sigma}$ simply changes the entries in the second status bar:

$$\begin{bmatrix} h_0 & h_2 \\ h_1 & h_3 \end{bmatrix} \xrightarrow{\tilde{\sigma}} \begin{bmatrix} h_0 & h_2 \\ h_3 & h_1 \end{bmatrix}$$

MixColumns: The t matrix is the next 8 × 8 bit matrix:

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

For this transformation, the state is considered as an 8×2 bit matrix. The state is multiplied on the left by t, using matrix multiplication modulo 2: $a = ta$.

KeySchedule: At the beginning of the cipher and at the end of each round, the state is bit-wise (i.e., modulo 2) with a round key. The columns of the round keys are defined recursively as follows:

$$w_0 = \begin{pmatrix} k_0 \\ k_1 \end{pmatrix}, \; w_1 = \begin{pmatrix} k_2 \\ k_3 \end{pmatrix},$$

$$w_{2i} = w_{2i-2} \oplus S(reverse(w_{2i-2})) \oplus r_i,$$

$$w_{2i+1} = w_{2i-1} \oplus w_{2i}$$

for each $i = 1,2,3,4$, where $r_i = \begin{pmatrix} 2^{i-1} \\ 0 \end{pmatrix}$, and *reverse* function replaces the two entries in the column. Function S is the same as described above.

It should be noted that all additions are performed bitwise modulo 2. Finally, for $i = 1,2,3,4$ the round key k_i is the matrix whose columns are w_{2i} and w_{2i}.

The use of the considered reduced model of the AES block symmetric cipher makes it possible to carry out experimental studies of the collisional characteristics of the pseudo-random pads being formed over the entire set of secret keys. Thus, a pseudo-random mini-UMAC pad $Pad_{\min i}$ is formed by encrypting a number *Nonce* that does not repeat for each information message $M_{\min i}$. The resulting value $Pad_{\min i}$ has a length of 16 bits, as well as the corresponding length of the hash code $Y_{\min i}$.

3.3. The mini-version of the final transform

The mini-version of the final transform for generating the mini-UMAC message authentication codes consists of modulo 2 values $Y_{\min i}$ and $Pad_{\min i}$: $Tag_{\min i} = Y_{\min i} \oplus Pad_{\min i}$.

Thus, the scaling of the applied transformations on the respective layers of the formation of the message authentication codes makes it possible to construct a reduced model of UMAC, experimentally investigate the collision properties of the generated images (codes). When developing a mini-model of UMAC, the scaling factor was chosen so that the length of generated hash codes *Y*, pseudo-random pads Pad and message authentication codes $Tag = Y \oplus Pad$ was equal to the length of the mini-block block of the AES symmetric block cipher [16, 19], i.e. 16 bits The choice of such a scaling factor allows, on the one hand, preserving the algebraic structure of the basic transformations of the UMAC algorithm, including the AES algorithm included in its scheme, on the other hand, it makes it possible to conduct experimental studies using the methods of statistical testing of hypotheses and mathematical statistics, considering a limited set of elements Y, Pad and $Tag = Y \oplus Pad$, and the corresponding results for estimating the number of collisions as a sample from the general population.

We justify the method of statistical estimation of the collision properties of the formed elements (we denote them for simplicity $h(x)$), consider the main conditions and limitations when conducting experiments.

3.4. Methods of statistical studies of collision properties

Experimental studies of the collision properties of UMAC message authentication codes will be carried out along the appropriate transformation layers:

1. At the first stage, we investigate the collision properties of the mini-version of universal hashing. To do this, it is necessary to confirm in the course of the experiment theoretical estimates of the number of collisions arising from the generated hash codes $Y_{\min i}$;

2. At the second stage, we will conduct experimental studies of the collisional characteristics of pseudorandom pads $Pad_{\min i}$ based on the analysis of the properties of the reduced Baby-Rijndael cipher model. Such studies in the available literature are not described and, apparently, conducted by us for the first time;

3. At the third stage, we will conduct experimental studies of the collisional characteristics of message authentication codes $Tag_{\min i} = Y_{\min i} \oplus Pad_{\min i}$ generated using mini-UMAC. This is the most important part of the research, since it will allow us to answer the question of preserving the properties of universal hashing after applying the layer of cryptographic transformation of information.

We will estimate the number of collisions of the formed elements, focusing on the collision properties of universal hashing. As a matter of fact, we need to confirm or refute the hypothesis about the preservation of the collision properties of universal hashing at all stages of the formation of message authentication codes mini-UMAC.

The idea of universal hashing is to define such a set of elements of a finite set *H* of hash functions $h: A \to B$, $|A| = a$, $|B| = b$, so that the random choice of the function x_1 and x_2 probability that $h(x_1) = h(x_2)$ (probability of collision, collision) must not exceed some predetermined value ε:

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

and the probability of a collision can be calculated as

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

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

We give two definitions of universal hashing [8, 9].

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

2. Let $0<\varepsilon<1$. H is ε – *is a strictly universal hash class (abbreviated* $\varepsilon - SU(H,A,B)$*)* if the following conditions are true:
– for each $x_1 \in A$ and for each $y_1 \in B$,

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

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

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

The definition of a universal class of hash functions is equivalent to the definition of an authentication code generation algorithm, in which the number of different authentication code generation rules (number of keys) for which there is a collision (matching authentication codes) for two arbitrary input sequences is limited. The number of such keys cannot exceed the value $P_{coll} \cdot |H|$, where P_{coll} – is the probability of a collision, $|H|$ – is the number of all rules (keys).

The definition of a strictly universal class of hash functions is equivalent to the definition of such an algorithm for the formation of authentication codes, in which the following rules will be executed:
1. The number of rules for the formation of the authentication code (the number of keys) for which the value of the authentication code does not change for an arbitrary input sequence is limited. The number of such keys cannot exceed the value $|H|/|B|$, where $|H|$ – is the number of all keys, $|B|$ – is the number of possible states of the authentication code;
2. The number of rules for the formation of an authentication code (the number of keys) for which the corresponding values of the authentication code for two arbitrary input sequences do not change is limited. The number of such keys cannot exceed the value $P_{coll} \cdot |H|$, where P_{coll} – is the probability of a collision, $|H|$ – is the number of all keys.

The probability of collision of authentication codes in a scheme with strictly universal hashing is defined as $P_{coll} \leq \varepsilon$.

The basis of the proposed methodology for the statistical study of collisional characteristics of formed elements $h(x)$ is an empirical estimate of the maxima of the number of keys (hashing rules) for which:

1. For arbitrary $x_1, x_2 \in A$, $x_1 \neq x_2$ equality holds:

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

2. For arbitrary $x_1 \in A$ and $y_1 \in B$ equality holds:

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

3. For arbitrary $x_1, x_2 \in A$, $x_1 \neq x_2$ and $y_1, y_2 \in B$ equality holds:

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

The evaluation by the first criterion corresponds to the verification of the fulfillment of the condition for the universal class of hash functions, the evaluation by the second and third criterion – the conditions for the strictly universal class of hash functions.
We introduce the following notation:

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

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

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

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

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

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

Since the number of keys for which equalities (1), (2) and (3) can hold, should not exceed the corresponding value $P_{coll} \cdot |H|$, $|H|/|B|$ and $P_{coll}|H|/|B|$ we will estimate the maximum number of such keys for each of the considered sets of elements.

We confine ourselves to studying the statistical characteristics of the maxima of these quantities, and then compare the results obtained with $P_{coll} \cdot |H|$ (for the first criterion), with $|H|/|B|$ (for the second criterion) and with $P_{coll} \cdot |H|$ (for the third criterion).

Thus, it is proposed to use as statistical values for evaluating the collision properties for which we will conduct experimental studies:

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

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

We will evaluate the collision properties by the above criteria in the average statistical sense. In other words, when setting up an experiment, we will use a limited set of elements $x_1, x_2 \in A$, $x_1 \neq x_2$ and corresponding hash images $y_1, y_2 \in B$, considering the corresponding results as a sample from the general population.

The natural estimate for the mathematical expectation m of a random variable X is the arithmetic average of its observed values (or statistical average) [15]

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

where: N – is the number of realizations of the random variable X.

The variance estimate of the random variable X is determined by the expression:

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

By virtue of the central limit theorem of probability theory, for large values of the number of realizations N, the arithmetic average will have a distribution close to the normal law with the mathematical expectation

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

and standard deviation

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

where: σ – is the standard deviation of the estimated parameter.

In this case, the probability that the estimate deviates $\tilde{m}$ from its mathematical expectation by less than ε by (confidence probability) is equal to [15]:

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

where: $\Phi(x)$ – is the Laplace function, determined by the expression:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}}\int_0^x e^{-\frac{t^2}{2}} dt$$

Thus, when conducting experimental studies of collisional characteristics, we will use statistical methods for testing hypotheses and mathematical statistics.

1. From the general population of the random variable X, we form the sample as follows:

- for the average estimate of the mathematical expectation $m(n_1)$ and variance $D(n_1)$ as a random variable, the maximum is the maximum $n_1(x_1, x_2)$ at which equality $h(x_1) = h(x_2)$, takes place, therefore, a sample size *N:* X_1, X_2, ..., X_N is 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 evaluates $x_1, x_2 \in A$, $x_1 \neq x_2$, i.e. the total volume of the formed pairs of elements, $x_1, x_2 \in A$, $x_1 \neq x_2$ will be *NM*;
- for the average estimate $m(n_2)$ and $D(n_2)$ as a random variable, the maximum $n_2(x_1, y_1)$ appears when the equalities are fulfilled and, therefore, the sample that size is *N:* X_1, X_2, ..., X_N is formed by selecting *N* sets, each of which contains *M* pairs of elements $x_1 \in A$, $y_1 \in B$ and is estimated as $n_2(x_1, y_1)$. The total volume of the formed pairs of elements $x_1 \in A$, $y_1 \in B$ will be *NM*;
- for the average estimate $m(n_3)$ and $D(n_3)$ as a random variable, the maximum appears $n_3(x_1, x_2, y_1, y_2)$ when the equalities $y_1 = h(x_1)$ and $y_2 = h(x_2)$ are fulfilled, therefore, the sample that size is *N:* X_1, X_2, ..., X_N is formed by selecting *N* sets, each of which contains *M* quadruple elements $x_1, x_2 \in A$, $x_1 \neq x_2$, $y_1, y_2 \in B$ and is estimated as $n_3(x_1, x_2, y_1, y_2)$, the total volume of the formed fours will be *NM*.
-

2. In experimental studies of the collision properties of hashing, we will estimate the arithmetic average $\tilde{m}(n_i)$ of the observed values of the maxima n_i and variance $\tilde{D}(n_i)$, $i = 1,2,3$.

3. The reliability of the average estimates obtained is justified as follows. Let us fix the accuracy ε and calculate the values of the Laplace function, which, in accordance with expression (4), will give the corresponding confidence probabilities:

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

When reversing the problem, i.e. for a fixed confidence probability P_o with a sample size of *N* with a sample size of

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

where: t_ρ – solution of an equation $2\Phi(t_\rho) = P_\partial$.

Thus, the proposed method, using reduced models of individual transformation layers, based on an estimate of the collision distribution of the images being formed, allows us to experimentally investigate the collision properties of message authentication codes.

4. EXPERIMENTAL RESULTS

Using the developed miniature UMAC model (mini-UMAC) and the method of statistical study of the collision properties of message authentication codes, we will experimentally estimate the distribution of the number of collisions of the images being formed.

Since the UMAC scheme discussed above uses the family of universal hashing functions that were studied in detail in [1–7] on the first layer (when generating the hash code $Y_{\min i}$), we will conduct statistical studies only on the second layer (when forming a pseudo-random pad $Pad_{\min i}$) and at the final stage of formation authentication codes (after performing the summation $Tag_{\min i} = Y_{\min i} \oplus Pad_{\min i}$). It is at these stages, according to our assumption, that the universality properties of the generated authentication codes are violated.

To form a minified UMAC model (mini-UMAC), we use:

1st layer:

Let us denote the procedure for encrypting a block of data T of length *Blocklen* bytes using a secret key K of length *Keylen* bytes in the form of a certain function *Keylen*. Then the procedure for the formation of a pseudo-

random key sequence $K' = KDF(K, Index, Numbyte)$ can be represented as the following iterative (for each $i = 1, 2, ..., n$) transformation:

$$T_i = Index \| i,$$
$$K'_i = Enchiper(K, T_i),$$
$$K' = K'_1 \| K'_2 \| ... \| K'_n,$$

where: $n = \left\lceil \frac{Numbyte}{Blocklen} \right\rceil$.

The generated sequence of pseudo-random key data bits K' has length $Numbyte$ bytes, that is a multiple of the block length $Blocklen$ bytes.

$$n = \left\lceil \frac{Numbyte}{Blocklen} \right\rceil = \frac{1024 + 16 \times 3}{32} = \frac{1072}{32} = 33{,}5 \approx 33$$
$$\Rightarrow i = 1, 2, ..., 33$$
$$T_i = Index \| i$$

For the first layer $Index$=1, => T_i:

$T_1 = 1 \| 1 = 00000001$
$T_2 = 1 \| 2 = 00000001\ 00000010$
$T_3 = 1 \| 3 = 00000001\ 00000011$
$T_4 = 1 \| 4 = 00000001\ 00000100$
$T_5 = 1 \| 5 = 00000001\ 00000101$
$T_6 = 1 \| 6 = 00000001\ 00000110$
$T_7 = 1 \| 7 = 00000001\ 00000111$
$T_8 = 1 \| 8 = 00000001\ 00001000$
$T_9 = 1 \| 9 = 00000001\ 00001001$
$T_{10} = 1 \| 10 = 00000001\ 00001010$
$T_{11} = 1 \| 11 = 00000001\ 00001011$
$T_{12} = 1 \| 12 = 00000001\ 00001100$
$T_{13} = 1 \| 13 = 00000001\ 00001101$
$T_{14} = 1 \| 14 = 00000001\ 00001110$
$T_{15} = 1 \| 15 = 00000001\ 00001111$
$T_{16} = 1 \| 16 = 00000001\ 00010000$
$T_{17} = 1 \| 17 = 00000001\ 00010001$
$T_{18} = 1 \| 18 = 00000001\ 00010010$
$T_{19} = 1 \| 19 = 00000001\ 00010011$
$T_{20} = 1 \| 20 = 00000001\ 00010100$
$T_{21} = 1 \| 21 = 00000001\ 00010101$
$T_{22} = 1 \| 22 = 00000001\ 0010110$
$T_{23} = 1 \| 23 = 00000001\ 0010111$
$T_{24} = 1 \| 24 = 00000001\ 0011000$
$T_{25} = 1 \| 25 = 00000001\ 0011001$
$T_{26} = 1 \| 26 = 00000001\ 0011010$
$T_{27} = 1 \| 27 = 00000001\ 0011011$
$T_{28} = 1 \| 28 = 00000001\ 0011100$
$T_{29} = 1 \| 29 = 00000001\ 0011101$
$T_{30} = 1 \| 30 = 00000001\ 0011110$
$T_{31} = 1 \| 31 = 00000001\ 0011111$
$T_{32} = 1 \| 32 = 00000001\ 00100000$
$T_{33} = 1 \| 33 = 00000001\ 00100001$

$$K' = K'_1 \| K'_2 \| ... \| K'_n$$

2-nd layer:

$$n = \left\lceil \frac{Numbyte}{Blocklen} \right\rceil = \frac{24 \times 4}{32} = \frac{96}{32} = 3$$
$$\Rightarrow i = 1, 2, 3$$
$$T_i = Index \| i$$

For the second layer $Index$=2, => T_i:

$T_1 = 2 \| 1 = 00000010\ 00000001$ =>K1
$T_2 = 2 \| 2 = 00000010\ 00000010$ =>K2
$T_3 = 2 \| 3 = 00000010\ 00000011$ =>K3
$K_{L2} = K1 \| K2 \| K3$

3-rd layer:

$$n = \left\lceil \frac{Numbyte}{Blocklen} \right\rceil = \frac{64 \times 4}{32 * 4} = 2$$
$$\Rightarrow i = 1, 2$$
$$T_i = Index \| i$$

For the third layer =3, => : $Index$ =3, => T_i:

$T_1 = 3 \| 1 = 00000010\ 00000001$ =>K1
$T_2 = 3 \| 2 = 00000010\ 00000010$ =>K2
$K_{L31} = K1 \| K2$

When conducting statistical studies of the collision properties of the values $Pad_{\min i}$ and $Tag_{\min i}$ for each experiment, the mathematical expectations $m(n_1)$, $m(n_2)$ and $m(n_3)$, variances , $D(n_1)$, $D(n_2)$ and $D(n_3)$, and, as well as for fixed accuracy ε = 0,1 the corresponding confidence probability $P(|\tilde{m}(n_i) - m(n_i)| < \varepsilon)$ was calculated. Researches were conducted on the sample, size of $N = 100$, for the formation of each element of the sample was calculated by a maximum of a set of $M = 1000$ tuples of elements. Thus, the total volume of the formed sets was $NM = 10^5$.
The results of experimental studies are summarized in Table 2.

Table 2. The results of experimental researches of the collision properties of authentication codes generated using mini-AES and mini-UMAC, mini MASH-1, MASH-2 and mini-UMAC

	mini-AES, $Pad_{\min i}$	mini-UMAC, $Tag_{\min i}$	MASH-1	MASH-2
$\tilde{m}(n_1)$	-	4,23	41,42	0
$\tilde{D}(n_1)$	-	0,18	42,74	0
$P_\partial = P(\|\tilde{m}(n_1) - m(n_1)\| < \varepsilon)$	-	0,98	0,98	H 1
$\tilde{m}(n_2)$	6,68	4,78	3,99	1
$\tilde{D}(n_2)$	0,42	0,42	0,01	0
$P_\partial = P(\|\tilde{m}(n_2) - m(n_2)\| < \varepsilon)$	0,88	0,88	0,99	H 1
$\tilde{m}(n_3)$	0,19	5,31	0,26	0,31
$\tilde{D}(n_3)$	0,15	0,24	0,21	0,22
$P_\partial = P(\|\tilde{m}(n_3) - m(n_3)\| < \varepsilon)$	0,99	0,96	0,97	0,97

When examining the collision properties of authentication codes generated using the mini version of the AES cipher, the number of keys for which equality $h(x_1) = h(x_2)$ is performed was zero for all tests, i.e. $n_1(x_1, x_2) = 0$ in all $N = 100$ experiments. This result is explained by the following property. The AES cipher (like its mini version) implements a bijective mapping of a set of plaintexts into a set of cipher programs, i.e. for a fixed key, the generated ciphertexts corresponding to different plaintext will be different. The experimental research conducted by the first criterion entered was precisely to count the number of keys in which there is a collision (collision) of two ciphertexts corresponding to two different plaintexts, which is impossible by definition of a bijective cipher. In this regard, the statistical data on the first criterion for the mini version of the AES cipher in the table. 2 are not listed as non-informative.

The analysis of the data in Table 2 suggests that the results obtained are adequate and that they correspond to the statistical properties of the entire population of data. For a fixed accuracy ε = 0,1, high confidence values were obtained, which indicates the validity and reliability of the experimental results obtained.

Let us analyze the results of statistical studies of the collision properties of message authentication codes, compare the obtained results of the average estimates of mathematical expectations, $m(n_1)$, $m(n_2)$ and $m(n_3)$ the number of hashing rules for which equalities (1), (2) and (3) are satisfied, respectively, with theoretical estimates: with $P_{coll} \cdot |H|$ (for the first criteria), with $|H|/|B|$ (for the second criterion) and with $P_{coll} \cdot H$ (for the third criterion).

Consider the first criterion by which we estimate the number of hash rules for which there is a collision (coincidence of authentication codes) for two arbitrary input sequences. In accordance with

theoretical estimates, this value is bounded above $P_{coll} \cdot |H|$ by a number. Let us specify this (theoretical) estimate for authentication codes generated using mini-AES and mini-UMAC.
The power of the key set for mini-AES and mini-UMAC is $|H| = 2^{16}$, the power of the set of generated authentication codes is also $|B| = 2^{16}$. If we use the upper estimate of the probability of collisions as the inverse of the power of the generated authentication codes $P_{coll} = 2^{-16}$, we get $n_1(x_1, x_2) \le P_{кол} \cdot |H| = 1$. For the mini version of the AES cipher, this condition is met (justified by the bijectivity of the encryption transform), but the collisional characteristics of mini-UMAC are significantly inferior to this upper theoretical estimate. In fact, the number of collisions is more than four times higher than the theoretical limit and this position is confirmed with a high confidence level $P_{\partial} = P(|\tilde{m}(n_1) - m(n_1)| < 0{,}1) > 0{,}98$.

Consider the second criterion by which the hash rules are evaluated, for which for an arbitrary input sequence the value of the authentication code does not change. According to theoretical estimates, this value for authentication codes generated using mini-AES and mini-UMAC is bounded above by $|H|/|B| = 1$. The experimental results obtained indicate that the collisional characteristics of authentication codes generated using mini-AES and mini-UMAC do not satisfy the second criterion, the number of keys that do not change the authentication code several times for an arbitrary input sequence than the theoretical estimate for universal hashing.

In accordance with the *third criterion*, the number of hashing rules is estimated, for which, for two arbitrary input sequences, the corresponding values of the authentication code do not change. The theoretical estimate of this value for universal hashing is bounded above by $P_{coll} \cdot |H|$, which, using the upper estimate of the probability of collisions $P_{coll} = 2^{-16}$ gives $n_3(x_1, x_2, y_1, y_2) \le P_{coll} \cdot |H| = 1$. The values given in Table 2 indicate that the collisional characteristics of authentication codes generated using mini-AES satisfy the third criterion. At the same time, the number of mini-UMAC keys, for which the corresponding authentication code values do not change for two arbitrary input sequences, is more than five times higher than the upper theoretical estimate.

Thus, to ensure the universality of the UMAC hash code, it is necessary to “replace” the pad based on the AES blocks symmetric cipher with a crypto-resistant sequence, which ensures the crypto-resistance of the entire hashing system. In [10, 11, 20], as a “replacement”, it is proposed to use the MASH-1 and MASH-2 hashing algorithms, which provide the required level of cryptographic strength. Evaluation of cryptographic strength is given in Table 3.

However, the results presented in Table 2 shows that the implementation of the key hashing scheme based on the MASH-1 algorithm, while changing the values of the initialization vector by the secret key, does not allow for high collision properties. The number of collisions occurring significantly higher than the upper theoretical boundaries, both in the first and in the second criteria, therefore, this construction is not a universal and, moreover, strictly universal hashing scheme. This result was obtained with a high confidence level for high accuracy. Thus, for the first criterion, the confidence interval was (confidence level 0.98), for the second criterion (confidence level 0.99), for the third

Table 3. Results of experimental researches of the statistical security of hashing algorithms using the NIST STS package

Generator	The number of tests where testing has passed > 99%	The number of tests where testing has passed > 96%
MASH-1	101 (53%)	188 (99%)
MASH-2	126 (67%)	189 (100%)
MASH(EC)	141 (74%)	189 (100%)
HMAC-SHA-256	134 (71%)	187 (98%)
EMAC	138 (73%)	189 (100%)
RIPEMD-160	129 (68%)	189 (100%)

UMAC+MASH-2	173 (91%)	189 (100%)

(confidence level 0.97). The considered key hashing scheme based on the MASH-1 algorithm, when the initialization vector values change with a secret key, satisfies only the third criterion.

Using key hashing based on the MASH-2 algorithm when changing the values of the initialization vector with a secret key, on the contrary, provides high collision characteristics of universal hashing. However, their use as a "pad" significantly reduces the rate of conversion and the formation of a hash code. A promising direction is the use of crypto-code structures proposed in [21–24]. In Figure 2 shows a block diagram of a hashing algorithm, taking into account the use, as a pad, of a McEliece's crypto-code scheme on elliptic (EC) modified elliptic (shortened / extended) codes.

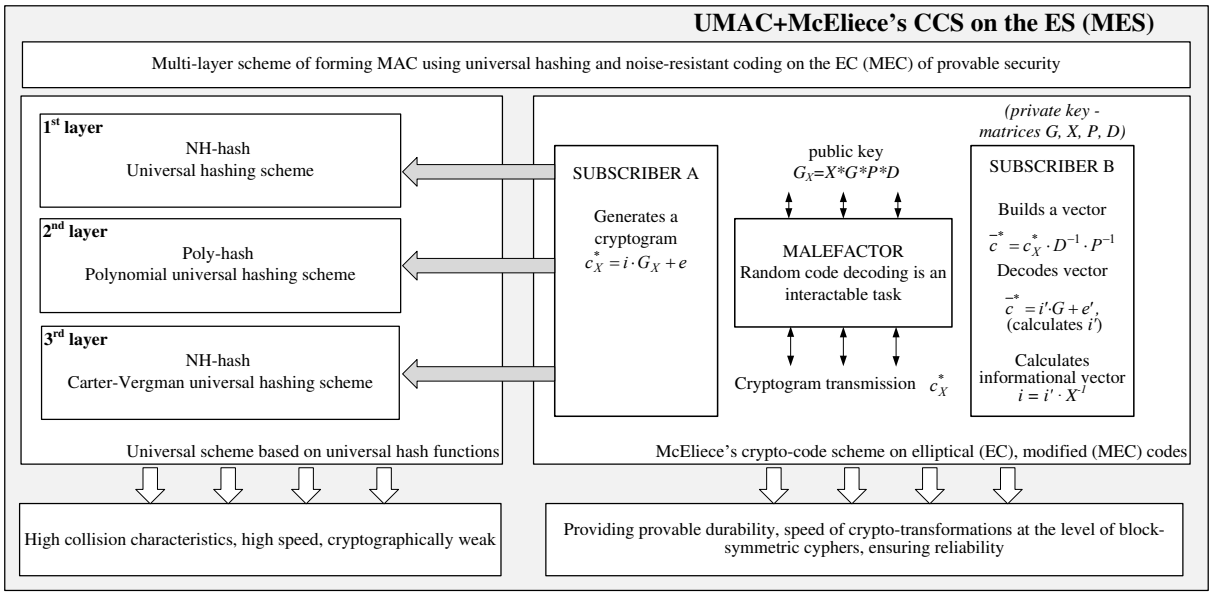

Figure 2. Block diagram of the UMAC algorithm with the McEliece's CCS on the EC (MEC)

This approach ensures the preservation of the properties of the universal hash algorithm, as well as the required level of security, efficiency and reliability in the formation of the hash code.

5. CONCLUSION

Thus, the results of statistical studies of the collision properties of hash codes based on mini-AES and mini-UMAC, as well as MASH-1 and MASH-2, allow us to state:

- the cryptographic layer of the formation of message authentication codes (mini-AES) satisfies the properties of universal hashing, the probability of a collision of generated hash images does not exceed a predetermined value. However, this transformation layer does not satisfy the properties of strictly universal hashing;
- the result of the formation of message authentication codes using the mini-UMAC scheme does not satisfy the properties of both universal hashing and, all the more, the properties of strictly universal hashing. This is because that the scheme with simple sum modulo two (XOR) of the two universal hash results does not always preserve the properties of the universal hash;
- the use of the MASH-1 algorithm as a hash-code does not meet the requirements for security and speed, the MASH-2 algorithm does not satisfy the speed of transformations, which does not allow their practical use in decentralized cryptocurrency systems.

The proposed application of the McEliece's crypto-code scheme on elliptical (EU), modified elliptical (shortened / elongated) codes retain the universality property on the first two layers of the UMAC algorithm, provides the required level of security, efficiency and reliability when generating the

message hash code. Thus, this algorithm allows to increase the levels of basic hashing functions in the blockchain technology in Bitcoin protocols.

REFERENCES

1. Black, J., Halevi, S., Krawczyk, H., Krovetz, T. and P. Rogaway, "UMAC: Fast and provably secure message authentication", Advances in Cryptology, CRYPTO '99, LNCS, Vol. 1666, Pages 216-233, 1999.

2. Krovetz, T. and Rogaway, P., "Fast universal hashing with small keys and no preprocessing, work in progress", http://www.cs.ucdavis.edu/~rogaway/umac, October 12, 2000.

3. Krovetz, T., Black, J., Halevi, S., Hevia, A., Krawczyk, H. and Rogaway, P., "UMAC -Message authentication code using universal hashing. IETF Internet Draft, draft-krovetz-umac-01.txt.", http://www.cs.ucdavis.edu/~rogaway/umac, November 15, 2000.

4. Krovetz T., "UMAC-Message authentication code using universal hashing. IETF Internet Draft, draft-krovetz-umac-02.txt.", http://www.cs.ucdavis.edu/~rogaway/umac, February 2, 2004.

5. "Final report of European project number IST-1999-12324, named New European Schemes for Signatures, Integrity and Encryption", Version 0.15 (beta), Springer-Verlag, April 19, 2004.

6. Krovetz T., "UMAC-Message authentication code using universal hashing", http://www.cs.ucdavis.edu/~rogaway/umac, June 23, 2006.

7. Krovetz T., "Software-Optimized Universal Hashing and Message Authentication. Dissertation submitted in partial satisfaction of the requirements for the degree of doctor of philosophy", University Of California Davis, California, September 2000.

8. Carter, J. L. and Wegman, M. N., "Universal classes of hash functions", Computer and System Scince, No. 18, Pages 143–154, 1979.

9. Wegman, M. N. and Carter, J. L., "New hash functions and their use in authentication and set equality", Computer and System Scince, No. 22, Pages 265–279, 1981.

10. Korol, O. G., Parhuts, L. T. and Yevseiev, S. P., "Investigation of properties of modular transformations and methods of hashing information on their basis", Information Processing Systems, No. 4(111), Pages 106–110, 2013.

11. Korol, O. G. and Yevseiev, S. P., "The method of universal hashing on the basis of modular transformations, Information processing systems", Information Technology and Computer Engineering, No. 7(97), Pages 131–132, 2011.

12. Korol, O. G., Yevseiev, S. P. and Dorokhov, A. V., "Mechanisms and protocols for protecting information in computer networks and systems", Scientific Journal of the Ministry of Defense of Republic of Serbia. Military Technical Gazette, Belgrade, No. 4, Pages 15–30, 2011.

13. Korol, O.G. and Yevseiev, S. P., "Results of the statistical test security hash algorithms-candidates tender to select standard hash algorithm SHA-3", News of higher technical educational institutions of Azerbaijan, No. 2, Pages 73–78, 2012.

14. Regenscheid, Andrew, Perlner, Ray, Chang, Shu-jen, Kelsey, John, Nandi, Mridul and Paul, Souradyuti, "Status Report on the First Round of the SHA-3 Cryptographic Hash Algorithm Competition", http://www.nist.gov/index.html, March 3, 2005.

15. Chung-Wei Phan Raphael, “Mini Advanced Encryption Standard (Mini-AES): A testbed for Cryptanalysis Students”, Cryptologia, XXVI (4), Pages 283–306, 2002.

16. A Description of Baby Rijndael, ISU CprE/Math 533; NTU ST765-U, 2003.

17. Lisitskaya, I. V., Grinenko, T. A. and Bessonov, S. Yu., “Analysis of the differential and linear properties of ciphers rijndael, serpent, threefish with 16-bit inputs and outputs”, East European Journal of Advanced Technologies, Pages 50-54, 2015.

18. Yevseiev, S. P., Ostapov, S. E. and Korolev, R. V., “Use of mini-versions for evaluation of the stability of block-symmetric ciphers”, Scientific and Technical Journal “Information Security”, Vol.23, No. 2, Pages 100–108, 2017.

19. Yevseiev, S. P., Yokhov, O. Y. and Korol, O. G., “Data Gaining in Information Systems: monograph”. pub. KhNUE, Kharkiv, 2013.

20. Yevseiev, S., Rzayev, H. and Tsyganenko, A., “Analysis of the software implementation of direct and inverse transformations using the non-binary balanced coding method”, Science and Technology Journal “Security Without Information”, Vol. 22, No. 2, Pages 196–203, 2016.

21. Yeseiev, S., “The use of flawed codes in crypto-code systems”, Information processing systems, No. 5 (151), Pages 109–121, 2017.

22. Yevseiev, S. and Bilodid, I., “The use of unprofitable codes in hybrid crypto-code designs”, Fifth International Scientific and Technical Conference “Problems of Informatization”, Cherkasy – Baku – Bielsko-Biala – Poltava, Page 11, 2017.

23. Hryshchuk, R., Yevseiev, S. and Shmatko, A., “Construction methodology of information security system of banking information in automated banking systems: monograph”, Pages 134–156, Premier Publishing s. r. o., Vienna, 2018.