Math-Net.Ru

Общероссийский математический портал

А. Ю. Nesterenko, Новый режим аутентифицированного шифрования для произвольного блочного шифра на основе универсальной функции хэширования, *Матем. вопр. криптогр.*, 2017, том 8, выпуск 2, 117–130

DOI: <https://doi.org/10.4213/mvk228>

Использование Общероссийского математического портала Math-Net.Ru подразумевает, что вы прочитали и согласны с пользовательским соглашением

<http://www.mathnet.ru/rus/agreement>

Параметры загрузки:

IP: 54.80.97.219

26 апреля 2023 г., 18:19:03

A new authenticated encryption mode for arbitrary block cipher based on universal hash function

A. Yu. Nesterenko

National Research University Higher School of Economics, Moscow

Получено 18.III.2016

Abstract. In this article we present a new authenticated encryption mode for arbitrary block cipher. This mode is a combination of well known XEX (Xor-Encryption-Xor) mechanism used in XTS encryption mode and universal hash function with predetermined properties from special class of functions. The bit length of authentication code being twice as much as the length of a cipher block is an important feature of our mode. The other important feature is the possibility of parallel implementation. The description, some security considerations and aspects of practical implementation are supplied.

Keywords: authenticated encryption, block cipher, universal hash function

Новый режим аутентифицированного шифрования для произвольного блочного шифра на основе универсальной функции хэширования

А. Ю. Нестеренко

Национальный исследовательский университет «Высшая школа экономики», Москва

Аннотация. В работе предлагается новый режим аутентифицированного шифрования, который является комбинацией хорошо известного режима XEX (Xor-Encryption-Xor) и универсальной функции хэширования с заданными свойствами из специального класса. Важной особенностью данного режима является тот факт, что длина кода аутентификации в два раза превосходит длину входного блока используемого шифра. Другой важной особенностью является возможность распараллеливания. В работе приводятся описание режима, обоснование его безопасности и рассматриваются вопросы практической реализации.

Ключевые слова: аутентифицированное шифрование, блочный шифр, универсальная функция хэширования

Citation: *Mathematical Aspects of Cryptography*, 2017, v. 8, № 2, pp. 117–130 (Russian)

© Академия криптографии Российской Федерации, 2017 г.

1. Description

Let w, m be two natural integers, $\mathbb{V}^w$ be the space of binary vectors of fixed finite length w and $\mathbb{V}^\infty \supset \mathbb{V}^w$ be the space of binary vectors of any finite length. In this article we define an authenticated encryption as follows

$$\text{AE} : \mathbb{V}^m \times \mathbb{V}^m \times \mathbb{V}^\infty \times \mathbb{V}^{2w} \rightarrow \mathbb{V}^\infty \times \mathbb{V}^{2w}, \quad \text{AE}(K_1, K_2, X, \text{iv}) = \{C, a\}, \quad (1)$$

where $X \in \mathbb{V}^\infty$ is a plaintext, $C \in \mathbb{V}^\infty$ is a ciphertext, $K_1 \in \mathbb{V}^m$ is a secret encryption key, $K_2 \in \mathbb{V}^m$ is a secret authentication key, $\text{iv} \in \mathbb{V}^{2w}$ is an initial vector and $a \in \mathbb{V}^{2w}$ is an authentication tag of message X .

Also, we define an inverse map

$$\text{AD} : \mathbb{V}^m \times \mathbb{V}^m \times \mathbb{V}^\infty \times \mathbb{V}^{2w} \times \mathbb{V}^{2w} \rightarrow \mathbb{V}^\infty \times \mathbb{V}^1, \quad \text{AD}(K_1, K_2, \{C, a\}, \text{iv}) = \{X, b\}, \quad (2)$$

where b is a Boolean variable, which is the result of authentication procedure. The authentication of message X is successful if the condition

$$\text{AD}(K_1, K_2, \text{AE}(K_1, K_2, X, \text{iv}), \text{iv}) = \{X, \text{«true»}\}$$

holds for any feasible values of K_1, K_2, X and iv . For detailed description of AE, AD maps we need to define some elementary operations over simple algebraic objects.

1.1. Mathematical background

For any vectors $X, Y \in \mathbb{V}^\infty$ we can define natural maps:

- $\text{len}_2(X)$ — bit length of X ,
- $X \parallel Y$ — concatenation of two vectors, where

$$\text{len}_2(X \parallel Y) = \text{len}_2(X) + \text{len}_2(Y)$$

(note that every vector $x = (x_0, x_1, \dots, x_{m-1}) \in \mathbb{V}^m$ may be represented as concatenation $x = x_0 \parallel x_1 \parallel \dots \parallel x_{m-1}$),

- $\text{lsb}(X, b)$ — least b significant bits of X ,
- $\text{msb}(X, b)$ — most b significant bits of X , for example

$$X = \text{lsb}(X \parallel Y, \text{len}_2(X)), \quad Y = \text{msb}(X \parallel Y, \text{len}_2(Y)),$$

- $E(K, X) : \mathbb{V}^m \times \mathbb{V}^w \rightarrow \mathbb{V}^w$ — block cipher which encrypts a message $X \in \mathbb{V}^w$ on a secret key $K \in \mathbb{V}^m$, for example algorithm AES [4] or Russian algorithms Magma or Kuznyechik [6]. Note that m is a bit length of secret key and w is a bit length of cipher block and takes values from the set $\{64, 128\}$.

Let $X = (x_0 \| \dots \| x_{w-1}) \in \mathbb{V}^\infty$ be a message with $\text{len}_2(X) = w$ and $x_0, \dots, x_{w-1} \in \mathbb{V}^1$. Consider a map $\text{int}_w : \mathbb{V}^w \rightarrow \mathbb{Z}_{2^w}$ and inverse of this map $\text{bin}_w : \mathbb{Z}_{2^w} \rightarrow \mathbb{V}^w$, where

$$\text{int}_w(X) = \sum_{n=0}^{w-1} x_n 2^n \in \mathbb{Z}_{2^w}, \quad \text{bin}_w(\text{int}_w(X)) = (x_0 \| \dots \| x_{w-1}).$$

Let $\gamma_0 \in \mathbb{V}^{2w}$. Define a function $T(\gamma) : \mathbb{V}^{2w} \rightarrow \mathbb{V}^{2w}$ which generates a sequence $\gamma_{n+1} = T(\gamma_n)$, $n = 0, 1, \dots$, with some large period τ . In practice we can use one of two variants of the function $T()$.

1. Let $f(x) \in \mathbb{F}_2[x]$ be a primitive polynomial of degree $2w$ and $\mathbb{F}_{2^{2w}}$ be a finite extension of $\mathbb{F}_2$ generated by $f(x)$. Let α be a root of $f(x)$ in $\mathbb{F}_{2^{2w}}$ then we can define a function T as follows

$$\gamma_{n+1} = T(\gamma_n) = \gamma_n \alpha = \gamma_0 \alpha^n. \quad (3)$$

Note that τ satisfies the equality $\tau = 2^{2w} - 1$ for $\gamma_0 \neq 0$, and $\tau = 1$ for $\gamma_0 = 0$. For practical evaluations we can use the following polynomials.

$2w$	$f(x)$
128	$f(x) = x^{128} + x^7 + x^2 + x + 1$
256	$f(x) = x^{256} + x^{10} + x^5 + x^2 + 1$

2. If $\theta_0, \theta_1 \in \mathbb{Z}_{2^w}$ are constants such that $\text{GCD}(\theta_0, 2^w - 1) = 1$, $\text{GCD}(\theta_1, 2^w) = 1$, then we can define the function $T()$ as follows

$$\gamma_{n+1} = T(\gamma_n) = \gamma_{n+1,0} \| \gamma_{n+1,1}, \quad (4)$$

where

$$\begin{aligned} \gamma_{n+1,0} &= \text{bin}_w \left(\theta_0 + \text{int}_w(\text{lsb}(\gamma_n, w)) \pmod{2^w - 1} \right), \\ \gamma_{n+1,1} &= \text{bin}_w \left(\theta_1 + \text{int}_w(\text{msb}(\gamma_n, w)) \pmod{2^w} \right). \end{aligned}$$

In this case τ satisfies an equality $\tau = 2^w(2^w - 1)$ for every $\gamma_0 \in \mathbb{V}^{2w}$.

Note that this function was firstly used in the Russian standard GOST 28147-89 for $w = 32$ and $\theta_0 = 168430012$ and $\theta_1 = 16843009$. Since $w \equiv 0 \pmod{8}$ we can define

$$\theta_1 = \sum_{i=0}^{\frac{w}{8}-1} 2^{8i}, \quad \theta_0 = \theta_1 + 3$$

for larger values of w . In this case we get different θ_0, θ_1 for different w .

At last we define a non linear map $\pi(z_1, z_2) : \mathbb{V}^w \times \mathbb{V}^w \rightarrow \mathbb{V}^{2w}$ such that $\pi(z_1, z_2)$ is a permutation on $\mathbb{V}^{2w}$. We can give two examples of map π .

1. In the first example we define a polynomial $f_\pi \in \mathbb{Z}[x_1, x_2]$

$$f_\pi(x_1, x_2) = \frac{1}{2} ((x_1 + x_2)^2 + x_1 + 3x_2) \quad (5)$$

and then define permutation $\pi : \mathbb{V}^w \times \mathbb{V}^w \rightarrow \mathbb{V}^{2w}$ as follows

$$\pi(z_1, z_2) = \begin{cases} \text{bin}_{2w}(f_\pi(x_1, x_2)), & \text{if } 0 \leq x_1 + x_2 < 2^w, \\ \text{bin}_{2w}(2^{2w} - 1 - f_\pi(2^w - 1 - x_1, 2^w - 1 - x_2)) & \text{otherwise,} \end{cases} \quad (6)$$

where $x_1 = \text{int}_w(z_1)$, $x_2 = \text{int}_w(z_2)$. In [13] we have proved that $\pi(z_1, z_2)$ is a bijective map on $\mathbb{V}^{2w}$.

2. In the second example we fix some permutation $\psi : \mathbb{V}^8 \rightarrow \mathbb{V}^8$ (for example taken from Kuznyechik [6]), finite extension $\mathbb{F}_{2^w}$ generated by the irreducible polynomial $h(x)$, $\deg h(x) = w$, and matrix

$$\Delta = \begin{pmatrix} \delta_0 & 1 \\ 1 & \delta_1 \end{pmatrix},$$

where $\delta_0, \delta_1 \in \mathbb{F}_{2^w}$ and $\delta_0\delta_1 \neq 0, 1$.

We represent every $\zeta \in \mathbb{V}^{2w}$ as a vector $\zeta = (\zeta_0 \parallel \dots \parallel \zeta_{2w/8-1})$, where $\zeta_i \in \mathbb{V}^8$, $i = 0, \dots, 2w/8 - 1$, and define a map $S : \mathbb{V}^{2w} \rightarrow \mathbb{V}^{2w}$ as follows

$$S(\zeta) = (\psi(\zeta_0) \parallel \dots \parallel \psi(\zeta_{2w/8-1})).$$

Also we represent every $\zeta \in \mathbb{V}^{2w}$ as a vector $\zeta = (\zeta_0 \parallel \zeta_1)$, where $\zeta_0, \zeta_1 \in \mathbb{V}^w$, and define a map $L : \mathbb{V}^{2w} \rightarrow \mathbb{V}^{2w}$ as follows

$$L(\zeta) = \Delta \zeta = \begin{pmatrix} \delta_0 & 1 \\ 1 & \delta_1 \end{pmatrix} \begin{pmatrix} \zeta_0 \\ \zeta_1 \end{pmatrix},$$

where multiplication is executed in $\mathbb{F}_{2^w}$. Now we define

$$\pi(z_1, z_2) = L(S(z_1 \parallel z_2)).$$

Since $\det \Delta = \delta_0\delta_1 - 1 \neq 0$ the permutation π is invertible and hence is a bijective map on $\mathbb{V}^{2w}$. In practice we can use the following parameters¹:

w	64	128
$h(x)$	$x^{64} + x^4 + x^3 + x + 1$	$x^{128} + x^7 + x^2 + x + 1$
δ_0	$(0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel 0 \parallel 1 \parallel 0 \parallel \dots \parallel 0) = 0x80$	$0x80$
δ_1	$(0 \parallel 0 \parallel 0 \parallel 0 \parallel 1 \parallel 0 \parallel \dots \parallel 0) = 0x10$	$0x10$

¹ The values of $h(x)$ are taken from GOST R 34.13-2015 [7, § 5.6.1]

1.2. AE algorithm

Authenticated encryption of plain message may be represented as simple algorithm that uses following subroutines: INIT(), STEP() and UPDATE().

The first subroutine INIT() set the values of two internal variables:

1. variable $\gamma \in \mathbb{V}^{2w}$ is the secret round key generated from authentication key K_2 and initial vector iv ;
2. variable $a \in \mathbb{Z}_{2^{2w}}$ is a current value of authentication code.

These variables are common for all subroutines used in authenticated encryption mode.

Algorithm 1. The INIT(K_2 , iv) subroutine

Input: AUTHENTICATION KEY $K_2 \in \mathbb{V}^m$ AND INITIAL VECTOR $iv \in \mathbb{V}^{2w}$.

Result: THE INITIAL STATE OF INTERNAL VARIABLES γ, s, l, a .

- 1 REPRESENT THE INITIAL VECTOR iv AS CONCATENATION $iv = iv_1 \parallel iv_2$, WHERE $\text{len}_2(iv_1) = \text{len}_2(iv_2) = w$;
- 2 DEFINE $\gamma = E(K_2, iv_1) \parallel E(K_2, iv_2)$;
- 3 IF $\gamma = (0 \parallel \dots \parallel 0)$ THEN
- 4 CALCULATE $iv_1 = \text{bin}_w(1 + \text{int}_w(iv_1))$ AND $iv_2 = \text{bin}_w(\text{int}_w(iv_2) + 3)$;
- 5 GO TO STEP 2.
- 6 DEFINE $a = 0$ AND STOP THE SUBROUTINE.

With very small probability we can get a value $\gamma = (0 \parallel \dots \parallel 0)$. If we use a function $T()$ defined by (3) such value may later lead to equalities $\gamma_0 = \gamma_1 = \dots = \gamma_n$ for every n compromising the authenticated encryption mode. Hence we must check equality $\gamma = (0 \parallel \dots \parallel 0)$ and choose a new value of initialization vector iv . When we use a function $T()$ defined by (4) steps 3–5 may be omitted.

Algorithm 2. The STEP(K , X) subroutine

Input: PLAIN MESSAGE X WITH FIXED LENGTH AND ENCRYPTION KEY K , WHERE $X \in \mathbb{V}^{2w}$, $K \in \mathbb{V}^m$.

Result: THE ENCRYPTED MESSAGE C , CORRESPONDING TO X , AND NEW STATES OF INTERNAL VARIABLES γ, a .

- 1 DEFINE $x_1 = \text{lsb}(X, w)$ AND $x_2 = \text{msb}(X, w)$;
- 2 CALCULATE THE NEW VALUE $\gamma = T(\gamma)$;
- 3 DEFINE $k_1 = \text{lsb}(\gamma, w)$ AND $k_3 = E(K, k_1 \oplus x_1)$;
- 4 DEFINE $k_2 = \text{msb}(\gamma, w)$ AND $k_4 = E(K, k_2 \oplus x_2)$;
- 5 DEFINE $c_1 = k_1 \oplus k_3$ $c_2 = k_2 \oplus k_4$ AND $C = c_1 \parallel c_2$;
- 6 DEFINE $z_1 \equiv \text{int}_w(k_1 \oplus x_1) + \text{int}_w(k_4) \pmod{2^w}$;
- 7 DEFINE $z_2 \equiv \text{int}_w(k_2 \oplus x_2) + \text{int}_w(k_3) \pmod{2^w}$;
- 8 CALCULATE $a \equiv a + \pi(z_1, z_2) \pmod{2^{2w}}$.

The second subroutine STEP() is a key point of authenticated encryption algorithm. This subroutine transforms a part of plain message into a part of encrypted message and updates the values of internal variables.

The STEP() subroutine has two stages. The first stage consists in encryption of plain message $x_1 \| x_2$ by applying Xor-Encryption-Xor procedure, i. e

$$c_1 = E(K_1, x_1 \oplus k_1) \oplus k_1, \quad c_2 = E(K_1, x_2 \oplus k_2) \oplus k_2,$$

where k_1, k_2 are secret tweaks². This way is similar to the XTS mode of encryption defined by NIST Special Publication 800-38E [15].

The second stage is an evaluation of a new state of internal variable a which contains a temporary value of authentication code. We can write this part of STEP() subroutine as follows

$$a \equiv a + \pi(z_1, z_2) \pmod{2^{2w}},$$

where

$$\begin{aligned} z_1 &\equiv \text{int}_w(x_1 \oplus k_1) + \text{int}_w(E(K_1, x_2 \oplus k_2)) \pmod{2^w}, \\ z_2 &\equiv \text{int}_w(x_2 \oplus k_2) + \text{int}_w(E(K_1, x_1 \oplus k_1)) \pmod{2^w}. \end{aligned}$$

Thus every argument of map $\pi(\cdot, \cdot)$ depends on the whole message $x_1 \| x_2$ and secret tweaks k_1, k_2 . Since z_1 has only linear dependence on x_1 and similarly z_2 has only linear dependence on x_2 we use a non-linear bijective map $\pi(\cdot, \cdot)$.

A graphic scheme of STEP() subroutine is presented in the next picture.

² By term “tweak” we mean pseudorandom secret value of fixed length, see [10].

The third subroutine UPDATE() encrypts a plain message of arbitrary length and evaluate a new states of internal variables.

Algorithm 3. The UPDATE(K, X) subroutine

Data: THE PLAIN MESSAGE $X \in \mathbb{V}^\infty$ AND ENCRYPTION KEY $K \in \mathbb{V}^m$.

Result: THE ENCRYPTED MESSAGE $C \in \mathbb{V}^\infty$, CORRESPONDING TO X , AND NEW STATES OF INTERNAL VARIABLES γ, a .

- 1 SET THE ENCRYPTED MESSAGE C AS A ZERO VECTOR;
- 2 REPRESENT A PLAIN MESSAGE AS $X = X_1 \parallel \dots \parallel X_{n-1} \parallel X_n$, WHERE $\text{len}_2(X_i) = 2w$ FOR EVERY $i = 1, \dots, n-1$ AND $0 \leq \text{len}_2(X_n) < 2w$;
- 3 FOR $i = 1, \dots, n-1$ DO
- 4 USE THE STEP(K, X_i) SUBROUTINE TO CALCULATE CIPHER TEXT C_i CORRESPONDING TO X_i , AND NEW VALUES OF INTERNAL VARIABLES;
- 5 DEFINE ENCRYPTED MESSAGE C AS $C = C \parallel C_i$.
- 6 IF $\text{len}_2(X_n) > 0$ THEN
- 7 DEFINE $X^* = X_n \parallel 1 \parallel 0 \parallel \dots \parallel 0$, WHERE $\text{len}_2(X^*) = 2w$;
- 8 USE THE STEP(K, X^*) SUBROUTINE TO CALCULATE CIPHERTEXT C^* CORRESPONDING TO X^* , AND NEW VALUES OF INTERNAL VARIABLES;
- 9 DEFINE ENCRYPTED MESSAGE C AS $C = C \parallel C^*$.
- 10 DEFINE A TEMPORARY VALUE $s \equiv \sum_{i=1}^n \text{int}_w(X_i) + \text{int}_w(X^*) \pmod{2^w}$.
- 11 DEFINE A TEMPORARY MESSAGE $S = \text{bin}_w(\text{len}_2(X)) \parallel \text{bin}_w(s)$.
- 12 USE THE STEP(K_1, S) SUBROUTINE TO CALCULATE NEW VALUES OF INTERNAL VARIABLES.

If binary length of plain message X is not exactly divided by $2w$ then the condition $\text{len}_2(X_n) > 0$ holds. In this case we use a second padding method from GOST R 34.13-2015, see [7, § 4.1.2], and $\text{len}_2(C) \equiv 0 \pmod{2w}$. Hence UPDATE() subroutine increases a length of encrypted message X for every message such that $\text{len}_2(X) \not\equiv 0 \pmod{2w}$.

The UPDATE() subroutine is presented in the next picture.

Now, we can define authenticated encryption mode as following algorithm.

Algorithm 4. Authenticated encryption mode

Data: THE PLAIN MESSAGE $X \in \mathbb{V}^\infty$, ENCRYPTION KEY $K_1 \in \mathbb{V}^m$, AUTHENTICATION KEY $K_2 \in \mathbb{V}^m$ AND INITIAL VECTOR $iv \in \mathbb{V}^{2w}$.

Result: THE ENCRYPTED MESSAGE $C \in \mathbb{V}^\infty$ AND AUTHENTICATION CODE $M \in \mathbb{V}^{2w}$.

- 1 SET THE ENCRYPTED MESSAGE C AS A ZERO VECTOR;
- 2 USE $\text{INIT}(K_2, iv)$ SUBROUTINE TO SET A VALUES OF INTERNAL VARIABLES;
- 3 USE $\text{UPDATE}(K_1, X)$ SUBROUTINE TO CALCULATE CIPHER TEXT C , CORRESPONDING TO X , AND NEW VALUES OF INTERNAL VARIABLES;
- 4 USE $\text{STEP}(K_2, \text{bin}_{2w}(a))$ TO CALCULATE AN ENCRYPTED TEXT M , CORRESPONDING TO a ;
- 5 RETURN M AS AUTHENTICATION CODE OF MESSAGE X .

1.3. AEAD feature

It is simple to add to the algorithm 4 the feature of authentication of non-encrypted messages. Let $A \in \mathbb{V}^\infty$ be a message which needs to be authenticated but not to be encrypted. In this case we can rewrite algorithm 4 as follows.

Note that in the step 3 of algorithm 5 we use the subroutine $\text{UPDATE}()$ only for evaluating new values of internal variable a . The encrypted text corresponding to A is not needed.

Algorithm 5. Authenticated encryption mode with authenticated data

Data: THE PLAIN MESSAGE $X \in \mathbb{V}^\infty$ WHICH NEEDS TO BE ENCRYPTED AND AUTHENTICATED, PLAIN DATA $A \in \mathbb{V}^\infty$ WHICH NEEDS TO BE AUTHENTICATED, ENCRYPTION KEY $K_1 \in \mathbb{V}^m$, AUTHENTICATION KEY $K_2 \in \mathbb{V}^m$ AND INITIAL VECTOR $iv \in \mathbb{V}^{2w}$.

Result: THE ENCRYPTED MESSAGE $C \in \mathbb{V}^\infty$ CORRESPONDING TO X , AND AUTHENTICATION CODE $M \in \mathbb{V}^{2w}$ CORRESPONDING TO A AND X .

- 1 SET THE ENCRYPTED MESSAGE C AS A ZERO VECTOR;
- 2 USE $\text{INIT}(K_2, iv)$ SUBROUTINE TO SET A VALUES OF INTERNAL VARIABLES;
- 3 USE $\text{UPDATE}(K_2, A)$ SUBROUTINE TO CALCULATE NEW VALUES OF INTERNAL VARIABLES;
- 4 USE $\text{UPDATE}(K_1, X)$ SUBROUTINE TO CALCULATE CIPHER TEXT C CORRESPONDING TO X AND NEW VALUES OF INTERNAL VARIABLES;
- 5 USE $\text{STEP}(K_2, \text{bin}_{2w}(a))$ TO CALCULATE AN ENCRYPTED TEXT M CORRESPONDING TO a ;
- 6 RETURN M AS AUTHENTICATION CODE OF MESSAGES A AND X .

2. Security analysis

Firstly, we give a full algebraic form for AE transformation (algorithm 4).

Let $\gamma_0 = E(K_2, iv_1) \parallel E(K_2, iv_2)$ and $\gamma_n = T(\gamma_{n-1})$, $n = 1, 2, \dots$, where function $T()$ is defined by (3) or (4).

We can consider a sequence

$$x_1, x_2, \dots, x_{2t}, x_{2t+1}, x_{2(t+1)} \in \mathbb{V}^w, \quad (7)$$

where $x_1, \dots, x_{2t}$ is a plain message X with last block that may be padded by the procedure described in algorithm 3, $x_{2t+1} = \text{len}_2(X)$ and $x_{2(t+1)} = s$, where s was defined in algorithm 3.

Consider the sequence of temporary tweaks $k_1, k_2, k_3, k_4, \dots, k_{4t+1}, k_{4t+2}, k_{4t+3}, k_{4(t+1)}$, where

$$k_{4n-3} = \text{lsb}(\gamma_n, w), \quad k_{4n-2} = \text{msb}(\gamma_n, w), \quad (8)$$

$$k_{4n-1} = E(K_1, x_{2n-1} \oplus k_{4n-3}), \quad k_{4n} = E(K_1, x_{2n} \oplus k_{4n-2}), \quad n = 1, 2, \dots, t+1. \quad (9)$$

Now we can define an encrypted message as a sequence $c_1, c_2, \dots, c_{2t} \in \mathbb{V}^w$, where

$$c_{2n-1} = k_{4n-3} \oplus k_{4n-1}, \quad c_{2n} = k_{4n-2} \oplus k_{4n}, \quad n = 1, 2, \dots, t,$$

and authentication code

$$M = STEP\left(K_2, \sum_{n=1}^{t+1} \pi(z_{2n-1}, z_{2n}) \pmod{2^{2w}}\right),$$

where

$$z_{2n-1} \equiv \text{int}_w(x_{2n-1} \oplus k_{4n-3}) + \text{int}_w(k_{4n}) \pmod{2^w}, \quad (10)$$

$$z_{2n} \equiv \text{int}_w(x_{2n} \oplus k_{4n-2}) + \text{int}_w(k_{4n-1}) \pmod{2^w}. \quad (11)$$

and $STEP$ is an encryption procedure defined by algorithm 1.

For defence against known attacks on authenticated encryption modes we use the following assumption.

- We must use two independent secret keys — the key K_1 is a key for the message encryption and the key K_2 is a key for the message authentication. In fact we use the second key to generate a sequence of secret tweaks.
- The authentication code M should be a result of universal hash function based on additive construction with non-linear permutations.

- The value of universal hash function should be encrypted against attack on last secret tweaks conducted by the inverse of bijective map π .
- Every block of plain message should have a unique identifier used in calculations. Hence changing the position of every block holds the changing the value of authenticated code. As these unique identifiers we use secret tweaks depending on authentication key K_2 and position number of block of plain message.
- The cipher text cannot be used directly for the authentication code calculation. This property makes the information used in calculations unknown to the adversary.
- The authentication code should depend on the length of plain message and on simple control sum of plain message blocks.

From the other side AE transformation may be represented as combination of two transformations: encryption procedure generating the encrypted message $c_1, c_2, \dots, c_{2t}$ and procedure calculating an authentication code M . We consider these transformations separately.

2.1. Encryption procedure

As we have shown earlier the encryption procedure is the same as in XTS encryption mode for arbitrary block cipher. The XTS encryption mode is an instantiation of Rogaway's XEX (Xor-Encrypt-Xor) tweakable block cipher [18]. This encryption mode is specified in IEEE Std. 1619-2007 [8] and NIST SP 800-38E [15]. The proof of security of XTS in game-theoretic model may be found in [10, 18].

If the condition $\text{len}_2(X) \equiv 0 \pmod{2w}$ holds, then the presented mode coincides with the XTS mode. In other cases we have a difference in the encryption of the last block. The main reason of this difference consists in some simplification of practical implementation of our AE transformation. When a length of plain message satisfies an inequality $0 < \text{len}_2(X) \pmod{2w} < w$ the standard cipher text stealing procedure [16] should be executed on two separate fragments of plain message with length $2w$. This may be too complex for practical implementations of presented authenticated encryption mode.

We note that XTS mode and presented encryption mode have a following property. If the authentication key K_2 is known to adversary, then the sequence $k_1, k_2, k_5, k_6, k_9, k_{10}, \dots$, is also known to adversary, hence presented encryption mode coincides with the ECB encryption mode [7, 14]. If key K_2 is unknown, then our padding procedure gives to the adversary a chance to take values $c_i = E(K_1, k_i) \oplus k_i$ for some unknown values of k_i and encryption key K_1 .

2.2. Authentication

A notion of universal hash function was introduced by Carter and Wegman [3, 21] and has practical applications in many message authentication functions such as UMAC [1], Badger [2] etc. based on special additive construction. This construction does not allow to realize attacks using a polynomial properties of message authentication function, similar to the attack on GCM mode [11, 19].

In [12] Nandi had proposed a security properties for this additive construction. In [13] the author submitted a wide class of universal hash functions which satisfy the requirements from [12]. Now we can give a brief description of this class.

Let $x_1, x_2, \dots, x_{2t} \in \mathbb{Z}_{2^w}$ be a plain message and $k_1, k_2, \dots, k_{2st} \in \mathbb{Z}_{2^w}$ be a key sequence. Consider a map

$$g(k_1, \dots, k_s, x_1) : \mathbb{Z}_{2^w} \times \dots \times \mathbb{Z}_{2^w} \rightarrow \mathbb{Z}_{2^w},$$

with the following properties.

1. If values $k_1, \dots, k_s \in \mathbb{Z}_{2^w}$ are fixed, then map

$$g(k_1, \dots, k_s, x_1) = g(x_1) : \mathbb{Z}_{2^w} \rightarrow \mathbb{Z}_{2^w}$$

is a permutation on $\mathbb{Z}_{2^w}$.

2. The equation $g(k_1, \dots, k_s, x_1) = z$ has exactly $2^{w(s-1)}$ different solutions $(k_1, \dots, k_s)$ for fixed value $x_1 \in \mathbb{Z}_{2^w}$ and any $z \in \mathbb{Z}_{2^w}$.
3. Consider a fixed pair $x_1, y_1 \in \mathbb{Z}_{2^w}$. If $z_1, u_1 \in \mathbb{Z}_{2^w}$ takes all possible values then the system

$$\begin{cases} g(k_1, \dots, k_s, x_1) \equiv z_1 \pmod{2^w}, \\ g(k_1, \dots, k_s, y_1) \equiv u_1 \pmod{2^w}, \end{cases}$$

has exactly 2^{ws} solutions $(k_1, \dots, k_s)$.

Also we consider a bijective map $\pi(\cdot, \cdot) : \mathbb{Z}_{2^w} \times \mathbb{Z}_{2^w} \rightarrow \mathbb{Z}_{2^{2w}}$. Following [13, Theorems 1, 2] we can say that the function

$$h(x_1, \dots, x_{2t}, k_1, \dots, k_{2st}) \equiv \sum_{n=1}^t \pi(z_{2n-1}, z_{2n}) \pmod{2^{2w}},$$

where $z_n = g(k_{s(n-1)+1}, \dots, k_{sn}, x_n)$, is an universal hash function, i.e. for any $a \in \mathbb{Z}_{2^{2w}}$ and any fixed values $x_1, \dots, x_{2t}$ we can find exactly $2^{2w(sl-1)}$ values of $(k_1, \dots, k_{st})$ satisfying the condition

$$h(x_1, \dots, x_{2t}, k_1, \dots, k_{st}) \equiv a \pmod{2^{2w}}. \quad (12)$$

On the other hand, for any $a \in \mathbb{Z}_{2^{2w}}$ and any fixed values $k_1, \dots, k_{st}$ we can find exactly $2^{2w(l-1)}$ values of $(x_1, \dots, x_{2t})$ satisfying the condition (12). Hence a collision attacks on function h based on the nonequiprobability of secret keys or plain messages are inapplicable.

For the presented authenticated encryption mode we use $s = 2$ and function g defined as follows, see [13]:

$$g(k_1, k_2, x_1) \equiv \text{int}_w(k_1 \oplus x_1) + k_2 \pmod{2^w}.$$

The main difference from [13] consists in two facts.

- In the presented authentication mode we consider a message X as modified sequence of integers (7) with padding data, bit length and simple control sum of message X .
- The sequence of keys $k_1, k_2, k_3, k_4, \dots$ does depend on message X since condition (9) holds.

These facts leads us to the problem of constructing collisions for short messages X with $\text{len}_2(X) < 2w$. Let $X = x_1 \| x_2$ be a fixed message X after padding procedure; then the authentication code of X is equal to $\text{STEP}(a)$, where

$$a \equiv \xi_1 + \xi_2 \pmod{2^{2w}}, \quad \begin{aligned} \xi_1 &= \pi(g(k_1, k_4, x_1), g(k_2, k_3, x_2)), \\ \xi_2 &= \pi(g(k_5, k_8, \text{len}_2(X)), g(k_6, k_7, x_1 + x_2)), \end{aligned}$$

and

$$\begin{aligned} k_3 &= E(K_1, x_1 \oplus k_1), & k_4 &= E(K_1, x_2 \oplus k_2), \\ k_7 &= E(K_1, \text{len}_2(X) \oplus k_5), & k_8 &= E(K_1, (x_1 + x_2) \oplus k_6). \end{aligned}$$

In this case a collision problem consists in finding a message $Y = y_1 \| y_2$ with the same value of a . Consider a pair $\zeta_1, \zeta_2 \in \mathbb{Z}_{2^{2w}}$ such that $\zeta_1 + \zeta_2 \equiv a \pmod{2^{2w}}$. Since π is a bijective map there exists unique values $z_1, z_2, z_3, z_4 \in \mathbb{Z}_{2^w}$ such that

$$\zeta_1 = \pi(z_1, z_2), \quad \zeta_2 = \pi(z_3, z_4).$$

Hence the solution Y of the collision problem should satisfy the following system:

$$\begin{cases} z_1 \equiv k_1 \oplus y_1 + E(K_1, y_2 \oplus k_2) \pmod{2^w}, \\ z_2 \equiv k_2 \oplus y_2 + E(K_1, y_1 \oplus k_1) \pmod{2^w}, \\ z_3 \equiv k_5 \oplus \text{len}_2(y_1 \| y_2) + E(K_1, (y_1 + y_2) \oplus k_6) \pmod{2^w}, \\ z_4 \equiv k_6 \oplus (y_1 + y_2) + E(K_1, \text{len}_2(y_1 \| y_2) \oplus k_5) \pmod{2^w}. \end{cases}$$

If we know both secret keys K_1 and K_2 , then for known values of k_1, k_2, k_5, k_6 we can find no more than one solution of this system in variables y_1, y_2 . Moreover, the special form of this system does not allow to find a simple method of solving other than brute force testing of variables y_1, y_2 . The solution of this problem is impractical since secret keys K_1, K_2 are unknown.

3. Performance analysis

For practical implementation of presented authenticated encryption mode we use a block cipher Kuznyechik described in [6]. We use a fast implementation of this cipher based on the code by M.-J. O. Saarinen [20], which used a three tables with total size of 12888 bytes. Also we try a slow implementation which uses only 1792 bytes of memory and is based on direct implementation of linear shift register. The average speed of fast implementation is 122000 KB per second on iCore 5 with 4GB memory and average speed of slow implementation is 15500 KB per second. Since authentication code length has 256 bits, our implementations used a libGMP for big integer arithmetic [5].

As an alternative we implement two independent procedures — CBC encryption and message authentication code (MAC) algorithm described by GOST R 34.13-2015 [7]. The last algorithm is similar to CMAC algorithm described by ISO/IEC 9797-1. Comparison of average times computed by 10^7 encryptions of 1MB messages is presented below.

Algorithm	Average Time
Slow Kuznyechik with CBC + CMAC	0.132s
Slow Kuznyechik with AE	0.072s
Fast Kuznyechik with CBC + CMAC	0.015s
Fast Kuznyechik with AE	0.014s

Since CBC + CMAC mode uses a double encryption we see that additional average time for authentication code evaluation is 0.006s per 1MB. To decrease this time we can use an assembler code of bijective map π or use a more simple bijective map. Note that we have not performed a comparison between the presented authenticated encryption mode and GCM or similar modes because these modes are inapplicable in Russian Federation since GOST R 34.13-2015 exists.

References

- [1] Black J., Halevi S., Krawczyk H., Krovetz T., Rogaway P., “UMAC: Fast and provably secure message authentication”. In: “*CRYPTO '99*”, Lect. Notes Comput. Sci., **1666**, 1999, 216–233.
- [2] Boesgaard M., Scavenius O., Pedersen T., Christensen T., Zenner E., “Badger — a fast and provably secure MAC”. In: “*Appl. Crypt. Network Secur., ACNS 2005*”, Lect. Notes Comput. Sci., **3531**, 2005, 176–191.
- [3] Carter L., Wegman M., “Universal classes of hash functions”, *J. Comput. Syst. Sci.*, **18**:2 (1979), 143–154.
- [4] *FIPS-197. Announcing the Advanced Encryption Standard (AES)*: NIST, 2001, 47 pp.
- [5] *GMPLIB. The GNU Multiple Precision Arithmetic Library*, 2016, <https://gmplib.org/>.
- [6] *GOST R 34.12–2015. Information technology. Cryptographic data security. Block ciphers*, Moscow: Standardinform, 2015 (in Russian).
- [7] *GOST R 34.13–2015. Information technology. Cryptographic data security. Modes of operation for block ciphers*, Moscow: Standardinform, 2015 (in Russian).
- [8] *IEEE Std 1619-2007, The XTS-AES Tweakable Block Cipher*: Inst. Electr. Electron. Eng., Inc., 2008.
- [9] Lebedev P. A., Nesterenko A. Yu., “Authenticated encryption mode”, *Systemy vysokoi dostupnosti*, **9**:3 (2013), 6–13 (in Russian).
- [10] Lyskov M., Rivest R., Wagner D., “Tweakable Block Ciphers”, *J. Cryptol.*, **24** (2011), 588–613.
- [11] McGrew D., Viega J., “The security and performance of the Galois/Counter Mode (GCM) of operation”. In: “*INDOCRYPT 2004*”, Lect. Notes Comput. Sci., **3348**, 343–355.
- [12] Nandi M., *FSE 2014*, Lect. Notes Comput. Sci., **8540**, 2014.
- [13] Nesterenko A. Yu., “On a family of universal hash functions”, *Mathematical Aspects of Cryptography*, **6**:3 (2015), 135–151 (in Russian).
- [14] *NIST Special Publication 800-38A. Recommendation for Block Cipher Modes of Operation*, 2001, <http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf>.
- [15] *NIST Special Publication 800-38E. Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices*, 2010, <http://csrc.nist.gov/publications/nistpubs/800-38E/nist-sp-800-38E.pdf>.
- [16] *Addendum to NIST Special Publication 800-38A. Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode*, 2010, <http://csrc.nist.gov/publications/nistpubs/800-38a/addendum-to-nist-sp800-38A.pdf>.
- [17] Preneel B., *Analysis and Design of Cryptographic Hash Functions.*, Leuven: Katholieke Univ. Leuven, 1993.
- [18] Rogaway P., “Efficient instantiations of tweakable blockciphers and refinements to modes OCB and PMAC”. In: “*ASIACRYPT 2004*”, Lect. Notes Comput. Sci., **3329**, 2004, 16–31.
- [19] Saarinen M.-J. O., “Cycling attacks on GCM, GHASH and other polynomial MACs and hashes”. In: “*FSE 2012*”, Lect. Notes Comput. Sci., **7549**, 2012, 216–225.
- [20] Saarinen M.-J. O., *The implemetation of Russian GOST R 34/12-2015 cipher “Kuznetchik”*, 2015, <https://github.com/mjosaarinen/kuznechik>.
- [21] Wegman M. N., Carter J. L., “New hash functions and their use in authentication and set equality”, *J. Comput. Syst. Sci.*, **22**:3 (1981), 265–279.