

GOST 28147-89: Encryption, Decryption,
and Message Authentication Code (MAC) Algorithms

Abstract

This document is intended to be a source of information about the Russian Federal standard for electronic encryption, decryption, and message authentication algorithms (GOST 28147-89), which is one of the Russian cryptographic standard algorithms called GOST algorithms). Recently, Russian cryptography is being used in Internet applications, and this document has been created as information for developers and users of GOST 28147-89 for encryption, decryption, and message authentication.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This is a contribution to the RFC Series, independently of any other RFC stream. The RFC Editor has chosen to publish this document at its discretion and makes no statement about its value for implementation or deployment. Documents approved for publication by the RFC Editor are not a candidate for any level of Internet Standard; see [Section 2 of RFC 5741](#).

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at <http://www.rfc-editor.org/info/rfc5830>.

Copyright Notice

Copyright (c) 2010 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document.

This document may not be modified, and derivative works of it may not be created, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	3
1.1. General Information	3
2. Applicability	3
3. Definitions and Notations	3
3.1. Definitions	3
3.2. Notation	4
4. General Statements	4
5. The Electronic Codebook Mode	6
5.1. Encryption of Plain Text in the Electronic Codebook Mode ...	6
5.2. Decryption of the Ciphertext in the Electronic Codebook Mode	9
6. The Counter Encryption Mode	10
6.1. Encryption of Plain Text in the Counter Encryption Mode ...	10
6.2. Decryption of Ciphertext in the Counter Encryption Mode ...	13
7. The Cipher Feedback Mode	13
7.1. Encryption of Plain Text in the Cipher Feedback Mode	13
7.2. Decryption of Ciphertext in the Cipher Feedback Mode	14
8. Message Authentication Code (MAC) Generation Mode	15
9. Security Considerations	17
10. Normative References	17
Appendix A . Values of the Constants C1 and C2	18
Appendix B . Contributors	19

1. Introduction

1.1. General Information

[GOST28147-89] is the unified cryptographic transformation algorithm for information processing systems of different purposes, defining the encryption/decryption rules and the message authentication code (MAC) generation rules.

This cryptographic transformation algorithm is intended for hardware or software implementation and corresponds to the cryptographic requirements. It puts no limitations on the encrypted information secrecy level.

2. Applicability

GOST 28147-89 defines the encryption/decryption model and MAC generation for a given message (document) that is meant for transmission via insecure public telecommunication channels between data processing systems of different purposes.

GOST 28147-89 is obligatory to use in the Russian Federation in all data processing systems providing public services.

3. Definitions and Notations

3.1. Definitions

The following terms are used in the standard:

Running key: a pseudo-random bit sequence generated by a given algorithm for encrypting plain texts and decrypting encrypted texts.

Encryption: the process of transforming plain text to encrypted data using a cipher.

MAC: an information string of fixed length that is generated from plain text and a key according to some rule and added to the encrypted data for protection against data falsification.

Key: a defined secret state of some parameters of a cryptographic transformation algorithm, that provides a choice of one transformation out of all the possible transformations.

Cryptographic protection: data protection using the data cryptographic transformations.

Cryptographic transformation: data transformation using encryption and (or) MAC.

Decryption: the process of transforming encrypted data to plain text using a cipher.

Initialisation vector: initial values of plain parameters of a cryptographic transformation algorithm.

Encryption equation: a correlation showing the process of generating encrypted data out of plain text as a result of transformations defined by the cryptographic transformation algorithm.

Decryption equation: a correlation showing the process of generating plain text out of encrypted data as a result of transformations defined by the cryptographic transformation algorithm.

Cipher: a set of reversible transformations of the set of possible plain texts onto the set of encrypted data, made after certain rules and using keys.

3.2. Notation

In this document, the following notations are used:

\wedge is a power operator.

(+) is a bitwise addition of the words of the same length modulo 2.

[+] is an addition of 32-bit vectors modulo 2^{32} .

[+]' is an addition of the 32-bit vectors modulo $2^{32}-1$.

1..N is all values from 1 to N.

4. General Statements

The structure model of the cryptographic transformation algorithm (a cryptographic model) contains:

- a 256-bit key data store (KDS) consisting of eight 32-bit registers (X0, X1, X2, X3, X4, X5, X6, X7);
- four 32-bit registers (N1, N2, N3, N4);
- two 32-bit registers (N5 and N6) containing constants C1 and C2;
- two 32-bit adders modulo 2^{32} (CM1, CM3);

- a 32-bit adder of bitwise sums modulo 2 (CM2);
- a 32-bit adder modulo $(2^{32}-1)$ (CM4);
- an adder modulo 2 (CM5), with no limitation to its width;
- a substitution box (K);
- a register for a cyclic shift of 11 steps to the top digit (R).

A substitution box (S-box) K consists of eight substitution points K1, K2, K3, K4, K5, K6, K7, K8, with 64-bit memory. A 32-bit vector coming to the substitution box is divided into eight successive 4-bit vectors, and each of them is transformed into a 4-bit vector by a corresponding substitution point. A substitution point is a table consisting of 16 lines, each containing four bits. The incoming vector defines the line address in the table, and the contents of that line is the outgoing vector. Then, these 4-bit outgoing vectors are successively combined into a 32-bit vector.

Remark: the standard doesn't define any S-boxes. Some of them are defined in [RFC4357].

When adding and cyclically shifting binary vectors, the registers with larger numbers are considered the top digits.

When writing a key (W1, W2, ..., W256), $W_q = 0..1$, $q = 1..256$, in the KDS the value:

- W1 is written into the 1st bit of the register X0;
- the value W2 is written into the 2nd bit of the register X0 (etc.);
- the value W32 is written into the 32nd bit of the register X0;
- the value W33 is written into the 1st bit of the register X1;
- the value W34 is written into the 2nd bit of the register X1 (etc.);
- the value W64 is written into the 32nd bit of the register X1;
- the value W65 is written into the 1st bit of the register X2 (etc.);
- the value W256 is written into the 32nd bit of the register X7.

When rewriting the information, the value of the p -th bit of one register (adder) is written into the p -th bit of another register (adder).

The values of the constants $C1$ and $C2$ in the registers $N5$ and $N6$ are in the Appendix 1.

The keys defining fillings of KDS and the substitution box K tables are secret elements and are provided in accordance with the established procedure.

The filling of the substitution box K is described in GOST 28147-89 as a long-term key element common for a whole computer network. Usually, K is used as a parameter of algorithm, some possible sets of K are described in [RFC4357].

The cryptographic model contemplates four working modes:

- data encryption (decryption) in the electronic codebook (ECB) mode,
- data encryption (decryption) in the counter (CNT) mode,
- data encryption (decryption) in the cipher feedback (CFB) mode, and
- the MAC generation mode.

[RFC4357] also describes the CBC mode of GOST 28147-89, but this mode is not a part of the standard.

5. The Electronic Codebook Mode

5.1. Encryption of Plain Text in the Electronic Codebook Mode

The plain text to be encrypted is split into 64-bit blocks. Input of a binary data block $T_p = (a1(0), a2(0), \dots, a31(0), a32(0), b1(0), b2(0), \dots, b32(0))$ into the registers $N1$ and $N2$ is done so that the value of $a1(0)$ is put into the first bit of $N1$, the value of $a2(0)$ is put into the second bit of $N1$, etc., and the value of $a32(0)$ is put into the 32nd bit of $N1$. The value of $b1(0)$ is put into the first bit of $N2$, the value of $b2(0)$ is put into the 2nd bit of $N2$, etc., and the value of $b32(0)$ is input into the 32nd bit of $N2$.

The result is the state $(a32(0), a31(0), \dots, a2(0), a1(0))$ of the register $N1$ and the state $(b32(0), b31(0), \dots, b1(0))$ of the register $N2$.

The 256 bits of the key are entered into the KDS. The contents of eight 32-bit registers $X0, X1, \dots, X7$ are:

$X0 = W32, W31, \dots, W2, W1$

$X1 = W64, W63, \dots, W34, W33$

\dots

$X7 = W256, W255, \dots, W226, W225$

The algorithm for enciphering 64-bit blocks of plain text in the electronic codebook mode consists of 32 rounds.

In the first round, the initial value of register N1 is added modulo 2^{32} in the adder CM1 to the contents of the register X0. Note: the value of register N1 is unchanged.

The result of the addition is transformed in the substitution block K, and the resulting vector is put into the register R, where it is cyclically shifted by 11 steps towards the top digit. The result of this shift is added bitwise modulo 2 in the adder CM2 to the 32-bit contents of the register N2. The result produced in CM2 is then written into N1, and the old contents of N1 are written in N2. Thus, the first round ends.

The subsequent rounds are similar to the first one:

- in the second round, the contents of X1 are read from the KDS;
- in the third round, the contents of X2 are read from the KDS, etc.;
- in the 8th round, the contents of X7 are read from the KDS.
- in rounds 9 through 16 and 17 through 24, the contents of the KDS are read in the same order:

$X0, X1, X2, X3, X4, X5, X6, X7$.

- in the last eight rounds from the 25th to the 32nd, the contents of the KDS are read backwards:

$X7, X6, X5, X4, X3, X2, X1, X0$.

Thus, during the 32 rounds of encryption, the following order of choosing the registers' contents is implemented:

$X0, X1, X2, X3, X4, X5, X6, X7, X0, X1, X2, X3, X4, X5, X6, X7,$

$X0, X1, X2, X3, X4, X5, X6, X7, X7, X6, X5, X4, X3, X2, X1, X0$

In the 32nd round, the result in the adder CM2 is written into the register N2, and the old contents of register N1 are unchanged.

After the 32nd round, the contents of the registers N1 and N2 are an encrypted data block corresponding to a block of plain text.

The equations for enciphering in the electronic codebook mode are:

$$\begin{aligned} | a(j) &= (a(j-1) [+] X(j-1)(\text{mod } 8)) * K * R (+) b(j-1) \\ | & \\ | b(j) &= a(j-1) \end{aligned} \quad j = 1..24;$$

$$\begin{aligned} | a(j) &= (a(j-1) [+] X(32-j)) * K * R (+) b(j-1) \\ | & \\ | b(j) &= a(j-1) \end{aligned} \quad j = 25..31; a_{32} = a_{31};$$

$$b(32) = (a(31) [+] X_0) * K * R (+) b(31) \quad j=32,$$

where:

$a(0) = (a_{32}(0), a_{31}(0), \dots, a_1(0))$ constitutes the initial contents of N1 before the first round of encryption;

$b(0) = (b_{32}(0), b_{31}(0), \dots, b_1(0))$ constitutes the initial contents of N2 before the first round of encryption;

$a(j) = (a_{32}(j), a_{31}(j), \dots, a_1(j))$ constitutes the contents of N1 after the j -th round of encryption;

$b(j) = (b_{32}(j), b_{31}(j), \dots, b_1(j))$ constitutes the contents of N2 after the j -th round of encryption, $j = 1..32$.

R is the operation of cyclic shift towards the top digit by 11 steps, as follows:

$$R(r_{32}, r_{31}, r_{30}, r_{29}, r_{28}, r_{27}, r_{26}, r_{25}, r_{24}, r_{23}, r_{22}, r_{21}, r_{20}, \dots, r_2, r_1) =$$

$$(r_{21}, r_{20}, \dots, r_2, r_1, r_{32}, r_{31}, r_{30}, r_{29}, r_{28}, r_{27}, r_{26}, r_{25}, r_{24}, r_{23}, r_{22})$$

The 64-bit block of ciphertext T_c is taken out of the registers N1, N2 in the following order:

the first, second, ..., 32nd bit of the register N1, then the first, second, ..., 32nd bit of the register N2, i.e.,

$$T_c = a_1(32), a_2(32), \dots, a_{32}(32), b_1(32), b_2(32), \dots, b_{32}(32).$$

The remaining blocks of the plain text in electronic codebook mode are encrypted in the same fashion.

5.2. Decryption of the Ciphertext in the Electronic Codebook Mode

The same 256-bit key that was used for encryption is loaded into the KDS, the encrypted data to be deciphered is divided into 64-bit blocks. The loading of any binary information block

$$T_c = (a_1(32), a_2(32), \dots, a_{32}(32), b_1(32), b_2(32), \dots, b_{32}(32))$$

into the registers N1 and N2 is done in such a way that:

- the contents of $a_1(32)$ are written into the first bit of N1;
- the contents of $a_2(32)$ are written into the second bit of N1 (and so on);
- the contents of $a_{32}(32)$ are written into the 32nd bit of N1;
- the contents of $b_1(32)$ are written into the first bit of N2 (and so on);
- and the contents of $b_{32}(32)$ are written into the 32nd bit of N2.

The decryption procedure uses the same algorithm as the encryption of plain text, with one exception: the contents of the registers X0, X1, ..., X7 are read from the KDS in the decryption rounds in the following order:

$$X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0,$$

$$X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0, X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0.$$

The decryption equations are:

$$\begin{array}{l|l} a(32-j) = (a(32-j+1) [+] X(j-1)) * K * R (+) b(32-j+1) & j = 1..8; \\ b(32-j) = a(32-j+1) & \\ \hline a(32-j) = (a(32-j+1) [+] X(j-1)(\text{mod } 8)) * K * R (+) b(32-j+1) & j = 9..31; \\ b(32-j) = a(32-j+1) & \\ \hline a(0) = a(1) & j=32. \\ b(0) = (a(1) [+] X_0) * K * R (+) b_1 & \end{array}$$

The fillings of the adders N1 and N2 after 32 working rounds are a plain text block.

$$Tp = (a1(0), a2(0), \dots, a32(0), b1(0), b2(0), \dots, b32(0))$$

corresponding to the encrypted data block:

- the value of $a1(0)$ of the block Tp corresponds to the contents of the first bit of $N1$;
- the value of $a2(0)$ corresponds to the contents of the second bit of $N1$ (etc.);
- the value of $b1(0)$ corresponds to the contents of the first bit of $N2$;
- the value of $b2(0)$ corresponds to the contents of the second bit of $N2$ (etc.);
- the value of $b32(0)$ corresponds to the contents of 32nd bit of $N2$;
- the remaining blocks of encrypted data are decrypted similarly.

The encryption algorithm in the electronic codebook mode of a 64-bit block Tp is denoted by A , that is:

$$A(Tp) \text{ is } A(a(0), b(0)) = (a(32), b(32)) = Tc.$$

6. The Counter Encryption Mode

6.1. Encryption of Plain Text in the Counter Encryption Mode

The plain text divided into 64-bit blocks $Tp(1), Tp(2), \dots, Tp(M-1), Tp(M)$ is encrypted in the counter encryption mode by bitwise addition modulo 2 in the adder CM5 with the running key Gc produced in 64-bit blocks, that is:

$$Gc = (Gc(1), Gc(2), \dots, Gc(M-1), Gc(M))$$

where M is defined by the size of the plain text being encrypted. $Gc(i)$ is the i -th 64-bit block where $i=1..M$, the number of bits in a block $Tp(M)$ can be less than 64. In this case, the unused part of the running key block $Gc(M)$ is discarded.

256 bits of the key are put into the KDS. The registers N1 and N2 accept a 64-bit binary sequence (an initialisation vector) $S = (S1, S2, \dots, S64)$, that is, the initial filling of these registers for subsequent generation of M blocks of the running key. The initialisation vector is put into the registers N1 and N2 so:

- the value of S1 is written into the first bit of N1;
- the value of S2 is written into the second bit of N1 (etc.);
- the value of S32 is written into the 32nd bit of N1;
- the value of S33 is written into the first bit of N2;
- the value of S34 is written into the 33th bit of N2 (etc.);
- the value of S64 is written into the 32nd bit of N2.

The initial filling of the registers N1 and N2 (the initialisation vector S) is encrypted in the electronic codebook mode in accordance with the requirements from [section 5.1](#). The result of that encryption $A(S) = (Y0, Z0)$ is rewritten into the 32-bit registers N3 and N4 so as the contents of N1 are written into N3, and the contents of N2 are written into N4.

The filling of the register N4 is added modulo $(2^{32}-1)$ in the adder CM4 to the 32-bit constant C1 from the register N6; the result is written into N4. The filling of the register N3 is added modulo 2^{32} in the adder CM3 with the 32-bit constant C2 from the register N5; the result is written into N3.

The filling of N3 is copied into N1, and the filling of N4 is copied into N2, while the fillings of N3 and N4 are kept.

The filling of N1 and N2 is encrypted in the electronic codebook mode according to the requirements of [section 5.1](#). The resulting encrypted filling of N1 and N2 is the first 64-bit block of the running key $Gc(1)$, this block is bitwise added modulo 2 in the adder CM5 with the first 64-bit block of the plain text:

$$Tp(1) = (t1(1), t2(1), \dots, t63(1), t64(1)).$$

The result of this addition is a 64-bit block of the encrypted data:

$$Tc(1) = (\tau1(1), \tau2(1), \dots, \tau63(1), \tau64(1)).$$

The value of $\tau_{1}(1)$ of the block $T_c(1)$ is the result of the addition modulo 2 in the CM5 the value $t_1(1)$ of the block $T_p(1)$ to the value of the first bit of N_1 ; the value of $\tau_2(1)$ of the block $T_c(1)$ is the result of addition modulo 2 in the CM5 the value of $t_2(1)$ from the block $T_p(1)$ to the value of the second bit of N_1 , etc.; the value of $\tau_{64}(1)$ of the block $T_c(1)$ is the result of addition modulo 2 in the CM5 of the value $t_{64}(1)$ of the block $T_p(1)$ to the value of the 32nd bit of N_2 .

To get the next 64-bit block of the running key $G_c(2)$, the filling of N_4 is added modulo $(2^{32}-1)$ in the adder CM4 with the constant C_1 from N_6 ; the filling of N_3 is added modulo 2^{32} in the adder CM3 with the constant C_2 from N_5 . The new filling of N_3 is copied into N_1 ; the new filling of N_4 is copied into N_2 ; the fillings of N_3 and N_4 are kept.

The filling of N_1 and N_2 is encrypted in the electronic codebook mode according to the requirements of [section 5.1](#). The resulting encrypted filling of N_1 and N_2 is the second 64-bit block of the running key $G_c(2)$; this block is bitwise added modulo 2 in the adder CM5 with the first 64-bit block of the plain text $T_p(2)$. The remaining running key blocks $G_c(3)$, $G_c(4)$, ..., $G_c(M)$ are generated and the plain text blocks $T_p(3)$, $T_p(4)$, ..., $T_p(M)$ are encrypted similarly. If the length of the last M -th block of the plain text is less than 64 bits, then only the corresponding number of bits from the last M -th block of the running key is used; remaining bits are discarded.

The initialisation vector S and the blocks of encrypted data $T_c(1)$, $T_c(2)$, ..., $T_c(M)$ are transmitted to the telecommunication channel or to the computer memory.

The encryption equation is:

$$\begin{aligned} T_c(i) &= A(Y[i-1] [+ C_2, Z[i-1]] [+ C_1] (+) T_p(i)) \\ &= G_c(i) (+) T_p(i) \quad i=1..M \end{aligned}$$

where:

$Y[i]$ is the contents of the register N_3 after encrypting the i -th block of the plain text $T_p(i)$;

$Z[i]$ is the contents of the register N_4 after encrypting the i -th block of the plain text $T_p(i)$;

$$(Y[0], Z[0]) = A(S).$$

6.2. Decryption of Ciphertext in the Counter Encryption Mode

256 bits of the key that was used for encrypting the data $Tp(1)$, $Tp(2)$, ..., $Tp(M)$ are put into the KDS. The initialisation vector S is put into the registers $N1$ and $N2$ and, like in the [section 6.1](#) blocks of the running key, $Gc(1)$, $Gc(2)$, ..., $Gc(M)$ are generated. The encrypted data blocks $Tc(1)$, $Tc(2)$, ..., $Tc(M)$ are added bitwise modulo 2 in the adder $CM5$ with the blocks of the running key, and this results in the blocks of plain text $Tp(1)$, $Tp(2)$, ..., $Tp(M)$, and $Tp(M)$ may contain less than 64 bit.

The decryption equation is:

$$\begin{aligned} Tp(i) &= A(Y[i-1] [+], C2, Z[i-1] [+], C1) (+) Tc(i) \\ &= Gc(i) (+) Tc(i) \quad i = 1..M \end{aligned}$$

7. The Cipher Feedback Mode

7.1. Encryption of Plain Text in the Cipher Feedback Mode

The plain text is divided into 64-bit blocks $Tp(1)$, $Tp(2)$, ..., $Tp(M)$ and encrypted in the cipher feedback mode by bitwise addition modulo 2 in the adder $CM5$ with the running key Gc generated in 64-bit blocks, i.e., $Gc(i) = (Gc(1), Gc(2), \dots, Gc(M))$, where M is defined by

the length of the plain text, $Gc(i)$ is the i -th 64-bit block, $i=1..M$. The number of bits in the block $Tp(M)$ may be less than 64.

256 bits of the key are put into the KDS. The 64-bit initialisation vector $S = (S1, S2, \dots, S64)$ is put into $N1$ and $N2$ as described in [section 6.1](#).

The initial filling of $N1$ and $N2$ is encrypted in the electronic codebook mode in accordance with the requirements in [section 6.1](#). If resulting encrypted filling $N1$ and $N2$ is the first 64-bit block of the running key $Gc(1) = A(S)$, then this block is added bitwise modulo 2 with the first 64-bit block of plain text $Tp(1) = (t1(1), t2(1), \dots, t64(1))$.

The result is a 64-bit block of encrypted data

$$Tc(1) = (\tau1(1), \tau2(1), \dots, \tau64(1)).$$

The block of encrypted data $Tc(1)$ is simultaneously the initial state of $N1$ and $N2$ for generating the second block of the running key $Gc(2)$ and is written on feedback in these registers. Here:

- the value of $\tau1(1)$ is written into the first bit of $N1$;

- the value of $\tau_{24}(1)$ is written into the second bit of $N1$, etc.;
- the value of $\tau_{32}(1)$ is written into the 32nd bit of $N1$;
- the value of $\tau_{33}(1)$ is written into the first bit of $N2$;
- the value of $\tau_{34}(1)$ is written into the second bit of $N2$, etc.;
- the value of $\tau_{64}(1)$ is written into the 32nd bit of $N2$.

The filling of $N1$ and $N2$ is encrypted in the electronic codebook mode in accordance with the requirements in the [section 6.1](#). The encrypted filling of $N1$ and $N2$ makes the second 64-bit block of the running key $Gc(2)$, this block is added bitwise modulo 2 in the adder CM5 to the second block of the plain text $Tp(2)$.

The generation of subsequent blocks of the running key $Gc(i)$ and the encryption of the corresponding blocks of the plain text $Tp(i)$ ($i = 3..M$) are performed similarly. If the length of the last M -th block of the plain text is less than 64 bits, only the corresponding number of bits of the M -th block of the running key $Gc(M)$ is used; remaining bits are discarded.

The encryption equations in the cipher feedback mode are:

$$\begin{aligned} &| Tc(1) = A(S) (+) Tp(1) = Gc(1) (+) Tp(1) \\ &| \\ &| Tc(i) = A(Tc(i-1)) (+) Tp(i) = Gc(i) + Tp(i), i = 2..M. \end{aligned}$$

The initialisation vector S and the blocks of encrypted data $Tc(1)$, $Tc(2)$, ..., $Tc(M)$ are transmitted into the telecommunication channel or to the computer memory.

7.2. Decryption of Ciphertext in the Cipher Feedback Mode

256 bits of the key used for the encryption of $Tp(1)$, $Tp(2)$, ..., $Tp(M)$ are put into the KDS. The initialisation vector S is put into $N1$ and $N2$ similar to 6.1.

The initial filling of $N1$ and $N2$ (the initialisation vector S) is encrypted in the electronic codebook mode in accordance with the [subsection 6.1](#). The encrypted filling of $N1$, $N2$ is the first block of the running key $Gc(1) = A(S)$, this block is added bitwise modulo 2 in the adder CM5 with the encrypted data block $Tc(1)$. This results in the first block of plain text $Tp(1)$.

The block of encrypted data $Tc(1)$ makes the initial filling of $N1$, $N2$ for generating the second block of the running key $Gc(2)$. The block $Tc(1)$ is written in $N1$ and $N2$ in accordance with the requirements in the subsection 6.1, the resulted block $Gc(2)$ is added bitwise modulo 2 in the adder CM5 to the second block of the encrypted data $Tc(2)$. This results in the block of plain text $Tc(2)$.

Similarly, the blocks of encrypted data $Tc(2)$, $Tc(3)$, ..., $Tc(M-1)$ are written in $N1$, $N2$ successively, and the blocks of the running key $Gc(3)$, $Gc(4)$, ..., $Gc(M)$ are generated out of them in the electronic codebook mode. The blocks of the running key are added bitwise modulo 2 in the adder CM5 to the blocks of the encrypted data $Tc(3)$, $Tc(4)$, ..., $Tc(M)$, this results in the blocks of plain text $Tp(3)$, $Tp(4)$, ..., $Tp(M)$; here, the number of bits in the last block of the plain text $Tp(M)$ can be less than 64 bit.

The decryption equations in the cipher feedback mode are:

$$\begin{aligned} &| Tp(1) = A(S) (+) Tc(1) = Gc(1) (+) Tc(1) \\ &| \\ &| Tp(i) = A(Tc(i-1)) (+) Tc(i) = Gc(i) (+) Tc(i), i=2..M \end{aligned}$$

8. Message Authentication Code (MAC) Generation Mode

To provide the protection from falsification of plain text consisting of M 64-bit blocks $Tp(1)$, $Tp(2)$, ..., $Tp(M)$, $M \geq 2$, an additional 1-bit block is generated (the message authentication code $I(1)$). The process of MAC generation is the same for all the encryption/decryption modes.

- The first block of plain text:

$$Tp(1) = (t1(1), t1(2), \dots, t64(1)) = (a1(1)[0], a2(1)[0], \dots, a32(1)[0], b1(1)[0], b2(1)[0], \dots, b32(1)[0])$$

is written to the registers $N1$ and $N2$;

- the value of $t1(1) = a1(1)[0]$ is written into the first bit of $N1$;
- the value of $t2(1) = a2(1)[0]$ is written into the second bit of $N1$, etc.;
- the value of $t32(1) = a32(1)[0]$ is written into the 32nd bit of $N1$;
- the value of $t33(1) = b1(1)[0]$ is written into the first bit of $N2$, etc.;
- the value of $t64(1) = b32(1)[0]$ is written into the 32nd bit of $N2$.

The filling of N1 and N2 is transformed in accordance with the first 16 rounds of the encryption algorithm in the electronic codebook mode (see the subsection 6.1). In the KDS, there exists the same key that is used for encrypting the blocks of plain text $Tp(1)$, $Tp(2)$, ..., $Tp(M)$ in the corresponding blocks of encrypted data $Tc(1)$, $Tc(2)$, ..., $Tc(M)$.

The filling of N1 and N2 after the 16 working rounds, looking like $(a1(1)[16], a2(1)[16], \dots, a32(1)[16], b1(1)[16], b2(1)[16], \dots, b32(1)[16])$, is added in CM5 modulo 2 to the second block $Tp(2) = (t1(2), t2(2), \dots, t64(2))$.

The result of this addition

$$\begin{aligned} & (a1(1)[16](+)t1(2), a2(1)[16](+)t2(2), \dots, a32(1)[16](+)t32(2), \\ & b1(1)[16](+)t33(2), b2(1)[16](+)t34(2), \dots, b32(1)[16](+)t64(2)) \\ & = \\ & (a1(2)[0], a2(2)[0] \dots, a32(2)[0], b1(2)[0], b2(2)[0], \dots, \\ & b32(2)[0]) \end{aligned}$$

is written into N1 and N2 and is transformed in accordance with the first 16 rounds of the encryption algorithm in the electronic codebook mode.

The resulting filling of N1 and N2 is added in the CM5 modulo 2 with the third block $Tp(3)$, etc., the last block $Tp(M) = (t1(M), t2(M), \dots, t64(M))$, padded if necessary to a complete 64-bit block by zeros, is added in CM5 modulo 2 with the filling N1, N2 $(a1(M-1)[16], a2(M-1)[16], \dots, a32(M-1)[16], b1(M-1)[16], b2(M-1)[16], \dots, b32(M-1)[16])$.

The result of the addition

$$\begin{aligned} & (a1(M-1)[16](+)t1(M), a2(M-1)[16](+)t2(M), \dots, a32(M-1)[16](+) \\ & t32(M), b1(M-1)[16](+)t33(M), b2(M-1)[16](+)t34(M), \dots, \\ & b32(M-1)[16](+)t64(M)) \\ & = \\ & (a1(M)[0], a2(M)[0] \dots, a32(M)[0], b1(M)[0], b2(M)[0], \dots, \\ & b32(M)[0]) \end{aligned}$$

is written into N1, N2 and encrypted in the electronic codebook mode after the first 16 rounds of the algorithm's work. Out of the resulting filling of the registers N1 and N2:

(a1(M)[16], a2(M)[16] ..., a32(M)[16], b1(M)[16], b2(M)[16], ..., b32(M)[16])

an l-bit string I(l) (the MAC) is chosen:

$$I(l) = [a(32-l+1)(M)[16], a(32-l+2)(M)[16], \dots, a32(M)[16]].$$

The MAC I(l) is transmitted through the telecommunication channel or to the computer memory attached to the end of the encrypted data, i.e., Tc(1), Tc(2), ..., Tc(M), I(l).

The encrypted data Tc(1), Tc(2), ..., Tc(M), when arriving, are decrypted, out of the resulting plain text blocks Tp(1), Tp(2), ..., Tp(M). The MAC I'(l) is generated as described in the [subsection 5.3](#) and compared with the MAC I(l) received together with the encrypted data from the telecommunication channel or from the computer memory. If the MACs are not equal, the resulting plain text blocks Tp(1), Tp(2), ..., Tp(M) are considered false.

The MAC I(l) (I'(l)) can be generated either before encryption (after decryption, respectively) of the whole message or simultaneously with the encryption (decryption) in blocks. The first plain text blocks, used in the MAC generation, can contain service information (the address section, a time mark, the initialisation vector, etc.) and they may be unencrypted.

The parameter l value (the bit length of the MAC) is defined by the actual cryptographic requirements, while considering that the possibility of imposing false data is 2^{-l} .

9. Security Considerations

This entire document is about security considerations.

10. Normative References

- [GOST28147-89] "Cryptographic Protection for Data Processing System", GOST 28147-89, Gosudarstvennyi Standard of USSR, Government Committee of the USSR for Standards, 1989. (In Russian)
- [RFC4357] Popov, V., Kurepkin, I., and S. Leontiev, "Additional Cryptographic Algorithms for Use with GOST 28147-89, GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 Algorithms", [RFC 4357](#), January 2006.

Appendix A. Values of the Constants C1 and C2

The constant C1 is:

The bit of N6	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18
---------------	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

The bit value	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

The bit of N6	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
---------------	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---

The bit value	1	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

The constant C2 is:

The bit of N6	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18
---------------	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

The bit value	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

The bit of N6	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
---------------	----	----	----	----	----	----	----	----	---	---	---	---	---	---	---	---	---

The bit value	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
---------------	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Appendix B. Contributors

Dmitry Kabelev
Cryptocom, Ltd.
14 Kedrova St., Bldg. 2
Moscow, 117218
Russian Federation

EMail: kdb@cryptocom.ru

Igor Ustinov
Cryptocom, Ltd.
14 Kedrova St., Bldg. 2
Moscow, 117218
Russian Federation

EMail: igus@cryptocom.ru

Irene Emelianova
Cryptocom Ltd.
14 Kedrova St., Bldg. 2
Moscow, 117218
Russian Federation

EMail: irene@cryptocom.ru

Author's Address

Vasily Dolmatov, Ed.
Cryptocom, Ltd.
14 Kedrova St., Bldg. 2
Moscow, 117218
Russian Federation

EMail: dol@cryptocom.ru