Internet-Draft V. Dolmatov, Ed.
Intended status: Informational Cryptocom Ltd.
Expires: June 21, 2010 December 21, 2009

GOST 28147-89
encryption, decryption and MAC algorithms
draft-dolmatov-cryptocom-gost2814789-08

Status of This Memo

Copyright Notice

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, message authentication.

Table of Contents

1. Introduction

1.1. General information

GOST 28147-89 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 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 required for use in the Russian Federation by all data processing systems providing public services.

3. Definitions and notations

3.1. Definitions

The following terms are used in the standard:

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

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

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

3.1.4 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.

3.1.5 Cryptographic protection: data protection using the data cryptographic transformations.

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

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

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

3.1.9 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.

3.1.10 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.

3.1.11 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:

^   is a power operator

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

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

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

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

4 General Statements

4.1. 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, N6) containing constants C2, C1;

- 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).

4.2. 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 4 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].

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

4.4. When writing a key (W1, W2, ..., W256), Wq = 0..1, q = 1..256, in the KDS the value W1 is written into the 1-st bit of the register X0, the value W2 is written into the 2-nd bit of the register X0, ..., the value W32 is written into the 32-nd bit of the register X0; the value W33 is written into the 1-st bit of the register X1, the value W34 is written into the 2-nd bit of the register X1, ..., the value W64 is written into the 32-nd bit of the register X1; the value W65 is written into the 1-st bit of the register X2 etc.; the value W256 is written into the 32-nd bit of the register X7.

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

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

4.7. 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].

4.8 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;

- the MAC generation mode.

[RFC4357] describes also 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

5.1.1. The plain text to be encrypted is split into 64-bit blocks. Input of a binary data block Tp = (a1(0), a2(0), ... , a31(0), a32(0), b1(0), b2(0), ..., 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 2_nd 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), ..., a2(0), a1(0)) of the register N1 and the state (b32(0), b31(0), ..., b1(0)) of the register N2.

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

```
X0 = W32, W31, ... , W2, W1

X1 = W64, W63, ... , W34, W33

. . . . . . . . . . . . . . .

X7 = W256, W255 ..., W226, W225
```

5.1.3. 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 is 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 the 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.

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

5.1.4. The equations for enciphering in the electronic codebook mode are:

```
|a(j) = (a(j-1) [+] X(j-1)(mod 8))*K*R (+) b (j-1)
|                                                        j = 1..24;
|b(j) = a(j-1)

|a(j) = (a(j-1) [+] X(32-j))*K*R (+) b(j-1)
|                                           j = 25..31; a32 = a31;
|b(j) = a(j-1)

b(32) = (a(31) [+] X0)*K*R (+) b(31)                        j=32,
```

where a(0) = (a32(0), a31(0), ..., a1(0)) is the initial contents of N1 before the first round of encryption;

b(0) = (b32(0), b31(0), ..., b1(0)) is the initial contents of N2 before the first round of encryption;

a(j) = (a32(j), a31(j), ..., a1(j)) is the contents of N1 after the j-th round of encryption;

b(j) = (b32(j), b31(j), ..., b1(j)) is 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(r32, r31, r30, r29, r28, r27, r26, r25, r24, r23, r22, r21, r20, ..., r2, r1) =

(r21, r20, ..., r2, r1, r32, r31, r30, r29, r28, r27, r26, r25, r24, r23, r22)

5.1.5. The 64-bit block of ciphertext Tc 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.,

Tc = a1(32), a2(32), ..., a32(32), b1(32), b2(32), ..., b32(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

5.2.1 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

Tc = (a1(32), a2(32), ..., a32(32), b1(32), b2(32), ..., b32(32)) into the registers N1 and N2 is done in such a way that the contents of a1(32) are written into the first bit of N1, the contents of a2(32) are written into the second bit of N1 and so on, the contents of a32(32) are written into the 32nd bit of N1; the contents of b1(32) are written into the first bit of N2 and so on, and the contents of b32(32) are written into the 32nd bit of N2.

5.2.2. 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:

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

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

5.2.3. The decryption equations are:

```
|a(32-j) = (a(32-j+1) [+] X(j-1))*K*R (+) b(32-j+1)
|                                                        j = 1..8;
|b(32-1) = a(32-j+1)

|a(32-j) = (a(32-j+1) [+] X(j-1)(mod 8))*K*R (+) b(32-j+1)
|                                                       j = 9..31;
|b(32-1) = a(32-j+1)

|a(0) = a(1)
|                                                            j=32.
|b(0) = (a(1) [+] X0)*K*R (+) b1
```

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

```
Tp = (a1(0), a2(0), ... , a32(0), b1(0), b2(0), ..., b32(0))
```

corresponding to the encrypted data block, and 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 bot of N2.

The remaining blocks of encrypted data are decrypted similarly.

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

```
A(Tp) 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

6.1.1 The plain text divided into 64-bit blocks Tp(1), Tp(2), ..., 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), ..., 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 bit 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.

6.1.2 256 bit 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, ..., 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 as 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.

6.1.3 The initial filling of the registers N1 and 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.

6.1.4 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 the 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), ..., t63(1), t64(1)).

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

Tc(1) = (tau1(1), tau2(1), ..., tau63(1), tau64(1)).

The value of tau1(1) of the block Tc(1) is the result of addition modulo 2 in the CM5 the value t1(1) of the block Tp(1) to the value of the first bit of N1, the value of tau2(1) of the block Tc(1) is the result of addition modulo 2 in the CM5 the value of t2(1) from the block Tp(1) to the value of the second bit of N1 etc., the value of tau64(1) of the block Tc(1) is the result of addition modulo 2 in the CM5 of the value t64(1) of the block Tp(1) to the value of the 32nd bit of N2.

6.1.5 To get the next 64-bit block of the running key Gc(2) the filling of N4 is added modulo (2^32-1) in the adder CM4 with the constant C1 from N6, the filling of N3 is added modulo 2^32 in the adder CM3 with the constant C2 from N5. The new filling of N3 is copied into N1, the new 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 the section 5.1. The resulting encrypted filling of N1 and N2 is the second 64-bit block of the running key Gc(2), this block is bitwise added modulo 2 in the adder CM5 with the first 64-bit block of the plain text Tp(2). The remaining running key blocks Gc(3), Gc(4), ..., Gc(M) are generated and the plain text blocks Tp(3), Tp(4), ..., Tp(M) are encrypted similarly. If the length of the last M-th block of the plain text is less than 64 bit then only the corresponding number of bit from the last M-th block of the running key is uses, remaining bit are discarded.

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

6.1.7 The encryption equation is:

```
   Tc(i) = A(Y[i-1] [+] C2, Z[i-1]) [+]' C1) (+) Tp(i)
         = Gc(i) (+) Tp(i)     i=1..M
```

where:

Y[i] is the contents of the register N3 after encrypting the i-th block of the plain text Tp(i);

Z(i) is the contents of the register N4 after encrypting the i-th block of the plain text Tp(i);

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

6.2. Decryption of ciphertext in the counter encryption mode

6.2.1 256 bit 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 sections 6.1.2 - 6.1.5 M 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.

6.2.2 The decryption equation is:

```
   Tp(i) = A (Y[i-1] [+] C2, Z[i-1] [+]' C1) (+) Tc(i)
         = Gc(i) (+) Tc(i)     i = 1..M
```

7. The cipher feedback mode

7.1. Encryption of plain text in the cipher feedback mode

7.1.1 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), ..., Gc(M)), where M is defined by the length of the plain text, Gc(i) is the i-th 64-bit block, $i=\overline{1,M}$. The number of bits in the block Tp(M) may be less than 64.

7.1.2 256 bit of key are put into the KDS. The 64-bit initialisation vector S = (S1, S2, ... S64) is put into N1 and N2 as described in the section 6.1.2.

7.1.3 The initial filling of N1 and N2 is encrypted in the electronic codebook mode in accordance with the requirements in section 6.1. The 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), ..., t64(1)).

The result is 64-bit block of encrypted data

```
   Tc(1) = (tau1(1), tau2(1), ..., tau64(1)).
```

7.1.4 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 tau2(1) is written into the second bit of N1, etc., the value of tau32(1) is written into the 32nd bit of N1; the value of tau33(1) is written into the first bit of N2, the value of tau34(1) is written into the second bit of N2 etc., the value of tau64(1) is written into the 32nd bit of N2.

The filling of N1, N2 is encrypted in the electronic codebook mode in accordance with the requirements in the section 6.1. The encrypted filling N1, 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) is performed similarly. If the length of the last M-th block of the plain text is less than 64 bit, only the corresponding number of bits of the M-th block of the running key Gc(M) is used, remaining bits are discarded.

7.1.5. The encryption equations in the cipher feedback mode are:

```
|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.
```

7.1.6 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

7.2.1 256 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 similarly to 6.1.2.

7.2.2 The initial filling of N1, 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).

7.2.3 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.

7.2.4. The decryption equations in the cipher feedback mode are:

```
|Tp(1) = A(S) (+) Tc(1) = Gc(1) (+) Tc(1)
|
|Tp(1) = A(Tc(i-1)) (+) Tc(i) = Gc(i) (+) Tc(i), i=2..M
```

8. Message authentication code (MAC) generation mode

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

8.2. The first block of plain text

```
Tp(1) = (t1(1), t1(2), ..., t64(1)) = (a1(1)[0], a2(1)[0], ...,
        a32(1)[0], b1(1)[0], b2(1)[0], ..., 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.

8.3. 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's 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], ..., a32(1)[16], b1(1)[16], b2(1)[16], ..., b32(1)[16]), is added in CM5 modulo 2 to the second block Tp(2) = (t1(2), t2(2), ..., t64(2)).

The result of this addition

```
(a1(1)[16](+)t1(2), a2(1)[16](+)t2(2), ..., a32(1)[16](+)t32(2),
b1(1)[16](+)t33(2), b2(1)[16](+)t34(2), ..., b32(1)[16](+)t64(2))

=

(a1(2)[0], a2(2)[0] ..., a32(2)[0], b1(2)[0], b2(2)[0], ...,
b32(2)[0])
```

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), ..., 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], ..., a32(M-1)[16], b1(M-1)[16], b2(M-1)[16], ..., b32(M-1)[16]).

The result of the addition

```
   (a1(M-1)[16](+)t1(M), a2(M-1)[16](+)t2(M), ..., a32(M-1)[16](+)
   t32(M), b1(M-1)[16](+)t33(M), b2(M-1)[16](+)t34(M), ...,
   b32(M-1)[16](+)t64(M))

 =

   (a1(M)[0], a2(M)[0] ..., a32(M)[0], b1(M)[0], b2(M)[0], ...,
   b32(M)[0])
```

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], ..., 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).

8.4. 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. IANA Considerations

This document has no actions for IANA.

11. Normative references

[GOST28147] "Cryptographic Protection for Data Processing System", GOST 28147-89, Gosudarstvennyi Standard of USSR, Government Committee of the USSR for Standards, 1989. (In Russian)

[RFC4357] RFC 4357. V.Popov, I.Kurepkin, 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

Appendix 1. Values of the constants C1, 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

Authors' Addresses

Vasily Dolmatov, Ed.
Cryptocom Ltd.
Kedrova st., 14, bld.2
Moscow, 117218, Russian Federation

EMail: dol@cryptocom.ru

Dmitry Kabelev
Cryptocom Ltd.
Kedrova st., 14, bld.2
Moscow, 117218, Russian Federation

EMail: kdb@cryptocom.ru

Igor Ustinov
Cryptocom Ltd.
Kedrova st., 14, bld.2
Moscow, 117218, Russian Federation

EMail: igus@cryptocom.ru

Irene Emelianova
Cryptocom Ltd.
Kedrova st., 14, bld.2
Moscow, 117218, Russian Federation

EMail: irene@cryptocom.ru