Inv. No. 3583 For Official Use
UDC 681.325.6:006.354 Group P85

Government Standard of the U.S.S.R.

Cryptographic Protection for Data Processing Systems

Cryptographic Transformation Algorithm*

GOST 28147—89

Effective Date 01 July 90

The following standard establishes the authorized cryptographic transformation algorithm for data processing systems in computer networks, separate computing complexes, and computers that defines the rules for data enciphering and for producing message authentication codes.

The cryptographic transformation algorithm is intended for either hardware or software implementation, satisfies the cryptographic requirements, and does not place any limitations on the secrecy level of the protected information.

This standard is obligatory for organizations, companies, and offices that use cryptographic protection for data stored in or transferred through computer networks, separate computer complexes, or computers.

Definitions of the terms used in this standard are given in Appendix 1.

* Translated from the Russian by Aleksandr Malchik, Sun Microsystems Laboratories, Mountain View, California, with editorial and typographic assistance from Whitfield Diffie, Sun Microsystems, Mountain View, California.

1. STRUCTURE OF THE CRYPTOGRAPHIC TRANSFORMATION ALGORITHM.

1.1. The cryptographic transformation contains (see Figure 1):

A 256-bit key memory unit (KMU) that consists of eight 32-bit registers $(X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7)$;

four 32-bit registers (N_1, N_2, N_3, N_4);

1/2

two 32-bit registers (N_5, N_6) containing constants S_2, S_1;

two 32-bit modulo 2^{32} adders (SM_1, SM_3);

one 32-bit modulo 2 bitwise adder (SM_2);

one 32-bit modulo $(2^{32} - 1)$ adder (SM_4);

one modulo 2 adder (SM_5), with no limitation placed on the length of adder SM_5;

a substitution block (K);

a cyclic shift register, shifting 11 steps towards the higher bit.

1.2. The substitution block K consists of 8 replacement modules (subkeys) $K_1, K_2, K_3, K_4, K_5, K_6, K_7, K_8$ with 64-bits of memory each. A 32-bit vector that enters the substitution block is split into eight 4-bit vectors, each of which is transformed into a 4-bit vector by the appropriate subkey, each representing a sixteen-line table containing 4 bits of data in each line. The entering vector determines the address of the line in the table, the contents of this particular line is the output vector.

1.3. During the addition and cyclic shift of binary vectors the higher bit is the one with bigger number.

1.4. When loading the key $(W_1, W_2, \ldots, W_{256}), W_q \in \{0,1\}, 1 \leq q \leq 256$, into the KMU, the contents of W_1 are entered into the 1^{st} bit of register X_0, the contents of W_2 are entered into the 2^{nd} bit of register X_0, …, the contents of W_{32} are entered into the 32^{nd} bit of register X_0, the contents of W_{33} are entered into the 1^{st} bit of register X_1, the contents of W_{34} are entered into the 2^{nd} bit of register X_1, …, the contents of W_{64} are entered into the 32^{nd} bit of register X_1, the contents of W_{65} are entered into the

1st bit of register X_2, etc.; finally, the contents of W_{256} are entered into the 32nd bit of register X_7.

1.5. During the transfer of data, the contents of bit p of one register (or adder) are written into bit p of another register (or adder).

1.6. The values of the constants S_1, S_2 in the registers N_6, N_5 is given in the Appendix 2.

1.7. The keys that determine the contents of the KMU and the tables of the substitution block K are secret elements and are distributed only in the proper channels.

The contents of the substitution tables of block K are a long-term key element that is common throughout a network.

Organization of the different kinds of communications is achieved by building an appropriate key management system. It is possible to produce the keys (KMU contents) and encipher them in electronic codebook mode or electronic codebook mode with message authentication for transmission over communication channels or storage in computer memory. 2 | 3

1.8. The cryptographic system has four modes of operation:

enciphering (and deciphering) of data in the electronic codebook mode;

enciphering (and deciphering) of data in the output feedback mode;

enciphering (and deciphering) of data in the cipher feedback mode;

producing message authentication codes.

The flowcharts for software implementation of the cryptographic transformation algorithm are given in Appendix 3.

Figure 1

3/4

2. ELECTRONIC CODEBOOK MODE.

2.1. Enciphering of plaintext in the electronic codebook mode.

2.1.1. The operation of the enciphering algorithm in electronic codebook mode is shown in Figure 2.

Plaintext to be enciphered is split into 64-bit blocks. Input of any block $T_p = (a_1(0), a_2(0), \ldots, a_{31}(0), a_{32}(0), b_1(0), b_2(0), \ldots, b_{32}(0))$ of binary data into the registers N_1 and N_2 is done so that the contents of $a_1(0)$ are entered into the 1^{st} bit of N_1, contents of $a_2(0)$ are entered into the 2^{nd} bit of N_1 etc., contents of $a_{32}(0)$ are entered into the 32^{nd} bit of N_1; contents of $b_1(0)$ are entered into the 1^{st} bit of N_2, contents of $b_2(0)$ are entered into the 2^{nd} bit of N_2 etc., contents of $b_{32}(0)$ are entered into the 32^{nd} bit of N_2. The result is: state $(a_{32}(0), a_{31}(0), \ldots, a_2(0), a_1(0))$ of the register N_1 and state $(b_{32}(0), b_{31}(0), \ldots, b_1(0))$ of the register N_2.

2.1.2. The 256 bits of key are entered into the KMU. The contents of eight 32-bit registers $X_0, X_1, \ldots, X_7$ are then:

$$
\begin{aligned}
X_0 &= (W_{32}, W_{31}, \ldots, W_2, W_1) \\
X_1 &= (W_{64}, W_{63}, \ldots, W_{34}, W_{33}) \\
&\cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \quad \cdot \\
X_7 &= (W_{256}, W_{255}, \ldots, W_{226}, W_{225})
\end{aligned}
$$

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

In the first round the initial contents of register N_1 are added modulo 2^{32} in the adder SM_1 to the contents of the register X_0. Note: the contents of register N_1 are unchanged.

The result of the addition is transformed in the substitution block K and the resulting vector enters register R, in which it is shifted, cyclically, by 11 steps towards the higher bits. The result of this shift is added bitwise modulo 2 in the adder SM_2 to the 32-bit contents of register N_2. The result produced in SM_2 is then stored in N_1. Note: the old contents of N_1 are stored in N_2. This ends the first round.

The subsequent rounds are analogous to the first one: in the 2nd round the contents of X_1 are read from the KMU, in the 3rd round the contents of X_2 are read from the KMU etc., in the 8th round the contents of X_7 are read from KMU. In rounds 9 through 16 and 17 through 24 the contents of the KMU are read in the same order:

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

In the last eight rounds from the 25th to the 32nd the contents of the KMU are read backwards:

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

Figure 2

In the 32 enciphering rounds, the registers are used in the following order:

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

In the 32nd round the output of the adder SM_2 is entered into the register N_2, and the old contents of register N_1 are unchanged.

The combined contents of registers N_1 and N_2 resulting from the 32nd round of enciphering is the block of ciphertext that corresponds to the block of plaintext.

3 3ak. 413 DSP

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

$$\begin{cases} a(j) = (a(j-1) \boxplus X_{(j-1) \pmod 8})KR \oplus b(j-1) \\ b(j) = a(j-1) \end{cases}$$

for $1 \leq j \leq 24$;

$$\begin{cases} a(j) = (a(j-1) \boxplus X_{(32-j)})KR \oplus b(j-1) \\ b(j) = a(j-1) \end{cases}$$

for $25 \leq j \leq 31$;

$$\begin{cases} a(32) = a(31) \\ b(32) = (a(31) \boxplus x_0)KR \oplus b(31) \end{cases}$$

for $j = 32$,

where $a(0) = (a_{32}(0), a_{31}(0), \ldots, a_1(0))$ — the initial contents of N_1 before the first round of enciphering;

$b(0) = (b_{32}(0), b_{31}(0), \ldots, b_1(0))$ — the initial contents of N_2 before the first round of enciphering;

$a(j) = (a_{32}(j), a_{31}(j), \ldots, a_1(j))$ — the contents of N_1 after the j^{th} round of enciphering;

$b(j) = (b_{32}(j), b_{31}(j), \ldots, b_1(j))$ — the contents of N_2 after the j^{th} round of enciphering, $1 \leq j \leq 32$.

The symbol $\oplus$ denotes the bitwise addition of 32-bit vectors modulo 2.

The symbol $\boxplus$ denotes addition of 32-bit vectors modulo 2^{32}. The rules of the addition modulo 2^{32} are given in Appendix 4;

R — cyclic shift towards the higher bits by 11 steps, as follows,

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

2.1.5. The 64-bit block of ciphertext T_c is taken out of the registers N_1, N_2 in the following order: the 1st, 2nd, …, 32nd bit of the register N_1, then the 1st, 2nd, …, 32nd bit of the register N_2, i.e.,

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

6/7 —

The remaining blocks of plaintext in electronic codebook mode are enciphered in the same fashion.

2.2. Deciphering of the ciphertext in the simple substitution mode.

2.2.1. The deciphering operation in the electronic codebook mode is the same as that for enciphering (see Figure 2). The same 256-bit key that was used for enciphering is loaded into the KMU. The ciphertext to be deciphered is divided into 64-bit blocks. The loading of a block

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

into the registers N_1 and N_2 is done in such a way that the contents of $a_1(32)$ are entered into the 1^{st} bit of N_1, the contents of $a_2(32)$ are entered into the 2^{nd} bit of N_1 and so on, the contents of $a_{32}(32)$ are entered into the 32^{nd} bit of N_1; the contents of $b_1(32)$ are entered into the 1^{st} bit of N_2 and so on, and the contents of $b_{32}(32)$ are entered into the 32^{nd} bit of N_2.

2.2.2. The deciphering procedure uses the same algorithm as the enciphering of plaintext, with one exception: the contents of the registers $X_0, X_1, \ldots, X_7$ are read from the KMU in the deciphering rounds in the following order:

$$\begin{gathered} X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7, X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0, \\ X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0, X_7, X_6, X_5, X_4, X_3, X_2, X_1, X_0. \end{gathered}$$

2.2.3. The equations for deciphering are:

$$\begin{cases} a(32-j) = (a(32-j+1) \boxplus X_{(j-1)})KR \oplus b(32-j+1) \\ b(32-j) = a(32-j+1) \end{cases}$$

$$\text{for } 1 \leq j \leq 8;$$

$$\begin{cases} a(32-j) = (a(32-j+1) \boxplus X_{(32-j) \pmod 8})KR \oplus b(32-j+1) \\ b(32-j) = a(32-j+1) \end{cases}$$

$$\text{for } 9 \leq j \leq 31;$$

$$\begin{cases} a(0) = a(1) \\ b(0) = (a(1) \boxplus X_0)KR \oplus b(1) \end{cases}$$

for $j = 32$,

2.2.4. The contents of the registers N_1 and N_2 resulting from the 32 rounds comprise the block of plaintext

$$T_p = (a_1(0), a_2(0), \ldots, a_{32}(0), b_1(0), b_2(0), \ldots, b_{32}(0)),$$

7|8 corresponding to the block of ciphertext. Note: the contents of $a_1(0)$ from block T_p correspond to the content of the 1$^{\text{st}}$ bit of N_1, the contents of $a_2(0)$ correspond to the contents of the 2$^{\text{nd}}$ bit of N_1 and so on, the contents of $a_{32}(0)$ correspond to the contents of the 32$^{\text{nd}}$ bit of N_1; the contents of $b_1(0)$ correspond to the contents of the 1$^{\text{st}}$ bit of N_2, the contents of $b_2(0)$ corresponds to the contents of the 2$^{\text{nd}}$ bit of N_2 and so on, the contents of $b_{32}(0)$ corresponds to the contents of the 32$^{\text{nd}}$ bit of N_2.

The remaining blocks of ciphertext are deciphered in the same way.

2.3. The enciphering algorithm in electronic codebook mode of the 64-bit block T_p is denoted by A, so

$$A(T_p) = A(a(0), b(0)) = (a(32), b(32)) = T_c.$$

2.4. Electronic codebook mode is to be used for enciphering (deciphering) data only in the cases described in 1.7.

3. OUTPUT FEEDBACK MODE.

3.1. Enciphering plaintext in the output feedback mode.

3.1.1. The operation of the enciphering algorithm in output feedback mode is shown in Figure 3.

The plaintext, divided into 64-bits blocks $T_p^{(1)}, T_p^{(2)}, \ldots, T_p^{(M-1)}, T_p^{(M)}$, is enciphered in the output feedback mode by bitwise addition modulo 2 in the adder SM_5 to the keystream Γ_c, which is produced in the 64-bit blocks

$$\Gamma_c = (\Gamma_c^{(1)}, \Gamma_c^{(2)}, \ldots, \Gamma_c^{(M-1)}, \Gamma_c^{(M)}),$$

where M is determined by the quantity of data to be enciphered.

$\Gamma_c^{(i)}$ is the i^{th} 64-bit block, $1 \leq i \leq M$. If the number of bits in the block $T_p^{(M)}$ is be less than 64, the part of the keystream not used for enciphering from the block $\Gamma_c^{(M)}$ is discarded.

3.1.2. The 256 bits of key are entered into the KMU. The 64-bit binary sequence (initialization vector) $S = (S_1, S_2, \ldots, S_{64})$ is entered into the registers N_1, N_2. This sequence is the initial contents of these registers for the subsequent production of M blocks of keystream. The initialization vector is entered into N_1 and N_2 so that the value of S_1 is entered into the 1^{st} bit of N_1, value of S_2 is entered into the 2^{nd} bit of N_1, etc., the value of S_{32} is entered into the 32^{nd} bit of N_1, the value of S_{33} is entered into the 1^{st} bit of N_2, the value of S_{34} is entered into the 2^{nd} bit of N_2, etc., the value of S_{64} is entered into the 32^{nd} bit of N_2.

3.1.3. The initial contents of the registers N_1 and N_2 (the initialization vector S) is enciphered in electronic codebook mode as described in 2.1. The result of enciphering $A(S) = (Y_0, Z_0)$ is copied into the 32-bit registers N_3 and N_4 so that the contents|of N_1 are copied into N_3, and contents of N_2 8|9
into N_4.

3.1.4. The contents of the register N_4 are added modulo $(2^{32} - 1)$ in the adder SM_4 with the 32-bit constant S_1 from the register N_6; the result is copied into N_4. The rules for addition modulo $(2^{32} - 1)$ are given in Appendix 4. The contents of the register N_3 are added modulo 2^{32} in the adder SM_3 with the 32-bit constant S_2 from the register N_5; the result is copied into N_3.

Figure 3

The contents of N_4 are copied into N_1, and the contents of N_3 into N_2, so that the contents of N_3 and N_4 are preserved.

The contents of N_1 and N_2 are enciphered in electronic codebook mode

as described in 2.1. The results of enciphering the contents of N_1, N_2 9 | 10
comprise the first 64-bit block of the keystream $\Gamma_c^{(1)}$, which is then added bitwise modulo 2 in the adder SM_5 to the first 64-bit block of plaintext

$$T_p^{(1)} = (t_1^{(1)}, t_2^{(1)}, \ldots, t_{63}^{(1)}, t_{64}^{(1)}).$$

The result is the 64-bit block of ciphertext $T_c^{(1)} = (\tau_1^{(1)}, \tau_2^{(1)}, \ldots, \tau_{63}^{(1)}, \tau_{64}^{(1)})$.

The value $\tau_1^{(1)}$ of the $T_c^{(1)}$ block is the result of addition modulo 2 in SM_5 of the value $\tau_1^{(1)}$ from the $T_p^{(1)}$ block with the value of the 1$^{\text{st}}$ bit of N_1; the value $\tau_2^{(1)}$ of the $T_c^{(1)}$ block is the result of addition modulo 2 in SM_5 of the value $\tau_2^{(1)}$ from the $T_p^{(1)}$ block with the value of the 2$^{\text{nd}}$ bit of N_1 and so on, the value of $\tau_{64}^{(1)}$ of the $T_p^{(1)}$ block — addition of $\tau_{64}^{(1)}$ with the value of the 32$^{\text{nd}}$ bit of N_2.

3.1.5. For the next 64-bit block of the keystream $\Gamma_c^{(2)}$ the contents of N_4 are added modulo $(2^{32} - 1)$ in the adder SM_4 with the constant S_1 from N_6, the contents N_3 are added modulo 2^{32} in the adder SM_3 with the constant S_3 from N_5. The new contents of N_3 are copied into N_1, and the new contents of N_4 are copied into N_2, so the contents of N_3, N_4 are unchanged.

The combined contents of N_1 and N_2 are enciphered in the simple substitution mode as described in 2.1. The result of enciphering the contents of N_1, N_2 comprises the second 64-bit block of keysteam $\Gamma_c^{(2)}$, which is then added bitwise modulo 2 in the adder SM_5 with the second block of plaintext $T_p^{(2)}$. The blocks of keystream $\Gamma_c^{(3)}, \Gamma_c^{(4)}, \ldots, \Gamma_c^{(M)}$ are produced and the blocks of plaintext $T_p^{(3)}, T_p^{(4)}, \ldots, T_p^{(M)}$ are enciphered in the same way.

If the length of the last plaintext block M is less than 64 bits, then from the last, M^{th}, block of the keystream $\Gamma_c^{(M)}$ only the corresponding number of bits of keystream are used for enciphering, the remaining bits are discarded.

3.1.6. The initialization vector S and the blocks of ciphertext $T_c^{(1)}, T_c^{(2)}, \ldots, T_c^{(M)}$. are transferred to the communication channel or to the memory of the computer.

3.1.7. The equation for enciphering is:

$$T_c^{(i)} = A(Y_{i-1} \boxplus S_2, Z_{i-1} \boxplus' S_1) \oplus T_p^{(i)} = \Gamma_c^{(i)} \oplus T_p^{(i)},$$
$$1 \leq i \leq M$$

where $\boxplus'$ denotes the addition modulo $(2^{32}-1)$ of the 32-bit contents; $\frac{10}{11}$

$\oplus$ — bitwise addition modulo 2 of the two contents;

Y_i — the contents of the register N_3 after enciphering the i^{th} block of plaintext $T_p^{(i)}$;

Z_i — the contents of the register N_4 after enciphering the i^{th} block of plaintext $T_p^{(i)}$;

$$(Y_0, Z_0) = A(S).$$

3.2. Deciphering of ciphertext in the output feedback mode.

3.2.1. The deciphering operation is the same as the enciphering operation (see Figure 3). The 256 bits of key that were used for enciphering the data $T_p^{(1)}, T_p^{(2)}, \ldots, T_p^{(M)}$ are entered into the KMU. The initialization vector S is entered into the registers N_1 and N_2, and, in analogy to 3.1.2–3.1.5 the process of the M blocks of the keystream $\Gamma_c^{(1)}, \Gamma_c^{(2)}, \ldots, \Gamma_c^{(M))}$ production is performed. The blocks of ciphertext $T_c^{(1)}, T_c^{(2)}, \ldots, T_c^{(M)}$. are added bitwise modulo 2 in the adder SM_5 with the blocks of the keystream. As the result, blocks of plaintext $T_p^{(1)}, T_p^{(2)}, \ldots, T_p^{(M)}$. are recovered, and $T_p^{(M)}$ may contain less than 64 bits.

3.2.2. The equation for deciphering is:

$$T_p^{(i)} = A(Y_{i-1} \boxplus S_2, Z_{i-1} \boxplus' S_1) \oplus T_c^{(i)} = \Gamma_c^{(i)} \oplus T_c^{(i)},$$
$$1 \leq i \leq M$$

4. CIPHER FEEDBACK MODE.

4.1. Enciphering plaintext in cipher feedback mode.

4.1.1. The operation of the enciphering algorithm in cipher feedback mode is shown in Figure 4.

The plaintext, divided into 64-bit blocks $T_p^{(1)}, T_p^{(2)}, \ldots, T_p^{(M)}$, is enciphered in cipher feedback mode by bitwise addition modulo 2 in the adder SM_5 with the keystream Γ_c, which is produced as 64-bit blocks, $(\Gamma_c^{(1)}, \Gamma_c^{(2)}, \ldots, \Gamma_c^{(M)})$, where M is determined by the quantity of plaintext and $\Gamma_i^{(i)}$ is the i^{th} 64-bit block, $1 \leq i \leq M$. The number of bits in the block $T_p^{(M)}$ may be less than 64.

4.1.2. The 256 bits of key are entered into the KMU. The 64-bit initialization vector, $S = (S_1, S_2, \ldots, S_{64})$, is entered into N_1 and N_2 as described in 3.1.2.

$\frac{11}{12}$

Figure 4

4.1.3. The initial contents of N_1 and N_2 are enciphered in the simple substitution mode as described in 2.1. The results of enciphering the contents N_1 and N_2 comprise the first 64-bit block of the keystream $\Gamma_c^{(1)} = A(S)$, which is added bitwise modulo 2 in the adder SM_5 to the first 64-bit block of plaintext $T_p^{(1)} = (t_1^{(1)}, t_2^{(1)}, \ldots, t_{64}^{(1)})$.

The result is the 64-bit block of ciphertext

$$T_c^{(1)} = (\tau_1^{(1)}, \tau_2^{(1)}, \ldots, \tau_{64}^{(1)}).$$

4.1.4. The block of ciphertext $T_c^{(1)}$ is, simultaneously, the initial contents of N_1, N_2 for the production of the second block of the keystream $\Gamma_c^{(2)}$ and, by feedback, is loaded into the registers noted above. The contents of $\tau_1^{(1)}$ are entered into the 1^{st} bit of N_1, the contents of $\tau_2^{(1)}$ are entered into

the 2^{nd} bit of N_1, and so on; the contents of $\tau_{32}^{(1)}$ into the 32^{nd} bit of N_1;
12|13 $\tau_{33}^{(1)}$ into the 1^{st} bit of N_2, $\tau_{34}^{(1)}$ 2^{nd} bit of N_2, and so on; $\tau_{64}^{(1)}$ 32^{nd} bit of N_2.

The contents of N_1, N_2 are enciphered in the electronic codebook mode according to the requirements of 2.1. The result of enciphering the contents of N_1, N_2 comprises the second 64-bit block of the keystream $\Gamma_c^{(2)}$, which is then added bitwise modulo 2 in the adder SM_5 with the second block of plaintext $T_c^{(2)}$.

The rest of the blocks of the keystream $\Gamma_c^{(i)}$ are produced and the corresponding blocks of plaintext $T_p^{(i)}(3 \le i \le M)$ are encrypted analogously. If the length of the last, M^{th}, block of plaintext $T_p^{(M)}$ is less than 64 bits, then from $\Gamma_c^{(M)}$ only the corresponding number of keystream bits are used and the rest are discarded.

4.1.5. The equations for the cipher feedback mode are:

$$\begin{aligned} T_c^{(1)} &= A(S) \oplus T_p^{(1)} = \Gamma_c^{(1)} \oplus T_p^{(1)} \\ T_c^{(i)} &= A(T_c^{(i-1)}) \oplus T_p^{(i)} = \Gamma_c^{(i)} \oplus T_p^{(i)} \qquad 1 \le i \le M \end{aligned}$$

4.1.6. The initialization vector S and blocks of ciphertext $T_c^{(1)}, T_c^{(2)}, \ldots, T_c^{(M)}$, are transferred into the communications channel or the memory of the computer.

4.2. Deciphering of ciphertext in the cipher feedback mode.

4.2.1. The deciphering operation is the same as the enciphering operation (see Figure 4).

The same 256 bits of key that were used for enciphering $T_p^{(1)}, T_p^{(2)}, \ldots, T_p^{(M)}$. are entered into the KMU. The initialization vector S is entered into N_1 and N_2 just as described in 3.1.2.

4.2.2. The initial contents of N_1, N_2 (initialization vector S) is enciphered in the electronic codebook mode according to 2.1. The result of enciphering the contents of N_1, N_2 comprise the first block of the keystream $\Gamma_c^{(1)} = A(S)$, which is then added bitwise modulo 2 in the adder SM_5 with the block of ciphertext $T_c^{(1)}$. The result is the first block of plaintext $T_p^{(1)}$.

4.2.3. Block of ciphertext $T_c^{(1)}$ is the initial contents of N_1, N_2 for the production of the second block of keystream $\Gamma^{(2)}$. Block $T_c^{(1)}$ is recorded into N_1, N_2 according to the requirements of 4.1.4. The received contents of N_1, N_2 are enciphered in the electronic codebook mode according to the requirements of 2.1, received in the result block $\Gamma_c^{(2)}$ is added bitwise modulo 13 | 14
2 in the adder SM_5 with the second block of ciphertext $T_c^{(2)}$. The result is plaintext block $T_c^{(2)}$.

In the same way, the blocks of ciphertext $T_c^{(2)}, T_c^{(3)}, \ldots, T_c^{(M-1)}$, from which the blocks of keystream $\Gamma_c^{(3)}, \Gamma_c^{(4)}, \ldots, \Gamma_c^{(M)}$ are derived in electronic codebook mode, are entered into N_1, N_2, one by one.

The blocks of keystream are added bitwise by modulo 2 addition in the adder SM_5 with blocks of enciphered data $T_c^{(3)}, T_c^{(4)}, \ldots, T_c^{(M)}$ producing blocks of plaintext data $T_p^{(3)}, T_p^{(4)}, \ldots, T_p^{(M)}$. The last block $T_p(M)$ may contain fewer than 64 bits.

4.2.4. The equations of decipherment in CFB have the form

$$T_p^{(1)} = A(S) \oplus T_c^{(1)} = \Gamma_c^{(1)} \oplus T_c^{(1)}$$

$$T_p^{(i)} = A(T_c^{(i-1)}) \oplus T_c^{(i)} = \Gamma_c^{(i)} \oplus T_c^{(i)}, 2 \leq i \leq M$$

$$T_p^{(1)} = A(S) \oplus T_c^{(1)} = \Gamma_c^{(1)} \oplus T_c^{(1)}$$

5. MESSAGE AUTHENTICATION MODE

5.1. To insure the authenticity of the plaintext, consisting of M 64-bit blocks $T_p^{(1)}, T_p^{(2)}, \ldots, T_p^{(M)}, M \geq 2$, the additional l-bit block (message authentication code I) is produced. The process of producing the message authentication code is the same for all modes of enciphering.

5.2. The first block of plaintext

$$T_p^{(1)} = \left(t_1^{(1)}, t_2^{(1)}, \ldots, t_{64}^{(1)}\right) = \left(a_1^{(1)}(0), a_2^{(1)}(0), \ldots, a_{32}^{(1)}(0),\right.$$
$$\left. b_1^{(1)}(0), b_2^{(1)}(0), \ldots, b_{32}^{(1)}(0)\right)$$

is stored in N_1 and N_2, so that the contents of $t_1^{(0)} = a_1^{(0)}(0)$ are entered into the 1st bit of N_1, the contents of $t_2^{(0)} = a_2^{(0)}(0)$ are entered into the 2nd bit of N_1 and so on; the contents of $t_{32}^{(0)} = a_{32}^{(0)}(0)$ are entered into the 32nd bit of N_1; the contents of $t_{33}^{(0)} = b_1^{(0)}(0)$ are entered into the 1st bit of N_2, etc.; the contents of $t_{64}^{(0)} = b_{32}^{(0)}(0)$ are entered into the 32nd bit of N_2.

5.3. N_1 and N_2 are transformed according to the first 16 rounds of enciphering in electronic codebook mode (see Section 2.1). Note: The K-MU contains the same key that was used to encipher the blocks of plaintext $T_p^{(1)}, T_p^{(2)}, \ldots, T_p^{(M)}$ into the corresponding blocks of the ciphertext $T_c^{(1)}, T_c^{(2)}, \ldots, T_c^{(M)}$.

The result after 16 cycles of enciphering N_1 and N_2, $(a_1^{(1)}(16), a_2^{(1)}(16), \ldots, a_{32}^{(1)}(16), b_1^{(1)}(16), b_2^{(1)}(16), \ldots, b_{32}^{(1)}(16))$, | is added in SM_5 by modulo 2 addition with the second block $T_p^{(2)} = \left(t_1^{(2)}, t_2^{(2)}, \ldots, t_{64}^{(2)}\right)$ 14 | 15

The result of the addition

$$\left(a_1^{(1)}(16) \oplus t_1^{(2)}, a_2^{(1)}(16) \oplus t_2^{(2)}, \ldots, a_{32}^{(1)}(16) \oplus t_{32}^{(2)},\right.$$
$$\left.b_1^{(1)}(16) \oplus t_{33}^{(2)}, b_2^{(1)}(16) \oplus t_{34}^{(2)}, \ldots, b_{32}^{(1)}(16) \oplus t_{64}^{(2)}\right) =$$
$$= \left(a_1^{(2)}(0), a_2^{(2)}(0), \ldots, a_{32}^{(2)}(0), b_1^{(2)}(0), b_2^{(2)}(0), \ldots, b_{32}^{(2)}(0)\right)$$

is loaded into N_1 and N_2 and is transformed by the first 16 cycles of enciphering in electronic codebook mode.

The resulting contents of N_1 and N_2 are added modulo 2 in the adder SM_5 with the third block $T_p^{(3)}$ and so on, until the last block $T_p^{(M)} = (t_1^{(M)}, t_2^{(M)}, \ldots, t_{64}^{(M)})$, which must be padded to a full 64-bit block with zeros is added modulo 2 in SM_5 to the registers N_1, N_2 $\left(a_1^{(M-1)}(16), a_2^{(M-1)}(16), \ldots, a_{32}^{(M-1)}(16), b_1^{(M-1)}(16), b_2^{(M-1)}(16), \ldots, b_{32}^{(M-1)}(16)\right)$.

The result of the addition

$$\left(a_1^{(M-1)}(16) \oplus t_1^{(M)}, a_2^{(M-1)}(16) \oplus t_2^{(M)}, \ldots, a_{32}^{(M-1)}(16) \oplus t_{32}^{(M)},\right.$$
$$\left.b_1^{(M-1)}(16) \oplus t_{33}^{(M)}, b_2^{(M-1)}(16) \oplus t_{34}^{(M)}, \ldots, b_{32}^{(M-1)}(16) \oplus t_{64}^{(M)}\right)$$
$$= \left(a_1^{(M)}(0), a_2^{(M)}(0), \ldots, a_{32}^{(M)}(0), b_1^{(M)}(0), b_2^{(M)}(0), \ldots, b_{32}^{(M)}(0)\right)$$

is written in N_1, N_2 and is enciphered in electronic codebook mode using the first 16 cycles of the algorithm. From the resulting registers N_1 and N_2

$$\left(a_1^{(M)}(16), a_2^{(M)}(16), \ldots, a_{32}^{(M)}(16), b_1^{(M)}(16), b_2^{(M)}(16), \ldots, b_{32}^{(M)}(16)\right)$$

the l-bit segment I_l (message authentication code) is chosen to be:

$$I_l = [a^{(M)}_{32-l+1}(16), a^{(M)}_{32-l+2}(16), \ldots, a^{(M)}_{32}(16)]$$

This message authentication code I_l is transfered into the communications channel or the memory of the computer at the end of the enciphered data: $T_c^{(1)}, T_c^{(2)}, \ldots, T_c^{(M)}, I_l$.

5.4. The received ciphertext $T_c^{(1)}, T_c^{(2)}, \ldots, T_c^{(M)}$ is deciphered and from the resulting blocks of plaintext the message authentication code I_l' is produced (see 5.3), which then is compared with the message authentication code I_l, received with the enciphered data from the communications channel or the computer memory. In case of a discrepancy between the two message authentication codes, the received blocks of plaintext $T_p^{(1)}, T_p^{(2)}, \ldots, T_p^{(M)}$ are considered inauthentic.

$\frac{15}{16}$

—

Production of the message authentication code I_1 (I_1') can be done either before encipherment (after decipherment) of the whole message, or in parallel with the encipherment (decipherment) blockwise. The first blocks of the plaintext that take part in production of the message authentication code may contain auxiliary information (address part, time mark, initialization vector, etc.) and may not be enciphered. The value of the parameter l (the number of bits in the message authentication code) is determined by the cryptographic security requirements and makes the probability of accepting inauthenic data 2^{-l}.

Appendix 1
Explanatory

TERMS USED IN THIS STANDARD AND THEIR DEFINITIONS.

Algorithm	GOST 19781
Authentication	Protection of the system of enciphered communication from introduction of inauthentic data.
Coding	Process of enciphering or deciphering.
Communication channel	GOST 17657
Cryptographic protection	Protection of data by cryptographic transformation.
Cryptographic transformation	Transformation of the data by enciphering and (or) generating a message authentication code.
Cryptosystem	A set of invertable transformations of the set of possible plaintexts into the set of possible ciphertexts, performed according to specified rules using the keys.
Data	GOST 15971
Encipherment of data	Process of transformating the plaintext into ciphertext using cryptographic algorithm.
Key	Specified secret setting of some parameters of a cryptographic algorithm that provides the choice of one transformation out of the set of possible transformations.
Keystream	Pseudo-random binary sequence generated by a specified algorithm for enciphering plaintext and deciphering ciphertext.
Message Authentication Code	A segment of information of fixed length, generated by a special rule from the plaintext and a key, and attached to the enciphered data, providing authentication.

Output Feedback	Process of masking the plaintext with the keystream generated by feeding the output of the cryptographic system back to its input.
Deciphering of data	Process of transformation of the ciphertext into plaintext using a cryptosystem. 16 17
Initialization vector	Values of the initial open parameters of the cryptographic transformation algorithm.
Equation of enciphering	Expression of the generation process of the ciphertext from the plaintext as the result of transformations defined by the cryptographic transformation algorithm.
Equation of deciphering	Expression of the generation process of the plaintext from the enciphered data as the result of transformations defined by the algorithm of the cryptographic transformation.

Appendix 2
Obligatory

THE CONSTANTS S_1, S_2

1. Constant S_1 is

Bit of Register N_6	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Value of Bit	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bit of Register N_6	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Value of Bit	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0

2. Constant S_2 is

Bit of Register N_5	32	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17
Value of Bit	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

Bit of Register N_5	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
Value of Bit	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

17 18

Appendix 3
Explanatory

FLOWCHARTS FOR SOFTWARE IMPLEMENTATION OF THE CRYPTOGRAPHIC TRANSFORMATION ALGORITHM

1. Flowchart for one round of enciphering

Figure 5

— $\frac{18}{19}$

2. Flowchart for 32 rounds of enciphering

Figure 6

3. Flowchart for 32 rounds of deciphering

Figure 7

— $\frac{19}{20}$

4. Flowchart for enciphering algorithm in electronic codebook mode

Figure 8

5. Flowchart for deciphering algorithm in electronic codebook mode

Figure 9

$\frac{20}{21}$

—

6. Flowchart for enciphering algorithm in output feedback mode

Figure 10

7. Flowchart for deciphering algorithm in output feedback mode

Figure 11

$\frac{21}{22}$

—

8. Flowchart for enciphering algorithm in cipher feedback mode

Figure 12

9. Flowchart for deciphering algorithm in cipher feedback mode

Figure 13

22/23

10. Flowchart for cryptographic transformation
for generating the message authentication code

Figure 14

23/24

Appendix 4
Explanatory

RULES OF ADDITION MODULO 2^{32}, AND $2^{32} - 1$

1. Two whole numbers a, b, where $0 < a, b < 2^{32} - 1$, represent binary numbers

$$a = (a_{32}, a_{31}, \ldots, a_2, a_1), b = (b_{32}, b_{31}, \ldots, b_2, b_1),$$

that is, $a = a_{32} \cdot 2^{31} + a_{31} \cdot 2^{30} + \cdots + a_2 \cdot 2 + a_1, b = b_{32} \cdot 2^{31} + b_{31} \cdot 2^{30} + \cdots + b_2 \cdot 2 + b_1$. are added modulo 2^{32} (operation $\boxplus$) by the following rule:

$$a \boxplus b = a + b, \text{ if } a + b < 2^{32},$$

$$a \boxplus b = a + b - 2^{32}, \text{ if } a + b \geq 2^{32},$$

where the operation $+(-)$ is the arithmetic sum (difference) of 2 whole numbers.

2. Two whole numbers a, b, where $0 < a, b < 2^{32} - 1$, represent binary numbers $a = (a_{32}, a_{31}, \ldots, a_2, a_1), b = (b_{32}, b_{31}, \ldots, b_2, b_1)$, are added modulo $(2^{32} - 1)$ (operation $(\boxplus')$ by the following rule:

$$a \boxplus b = a + b, \text{ if } a + b < 2^{32},$$

$$a \boxplus b = a + b - 2^{32} + 1, \text{ if } a + b \geq 2^{32},$$

24 | 25

Contents

—

$\boxed{\frac{25}{26}}$

Information About the Standard

1. Authors of the standard

   I. A. Zabotin (project leader), G. P. Glazkov, V.B. Isaeva

2. Accepted and introduced into use by the action of the State Standards Committee of the USSR on 2 June 89 as No. 1409.

3. To be reviewed in 1993

4. Initial release.

5. Technical-standardization documents cited

Technical standards documents cited	Number of the section, subsection, or appendix
GOST 19781—83	Appendix 1
GOST 15971—84	Appendix 1
GOST 17657—79	Appendix 1

Editor: G. A. Ivashina
Technical editor: L. Ya. Mitrofanova
Proof reader: A. I. Zyuban

Submitted for typesetting 28 June 89, Approved for publication 29 Sep 89, 1.73 USL, P. L. 1., 1.73 USL, P. L. kr.-ott. 1.54 uch.-izd. l.
800 Copies printed Price 10 Kopecs

Order of the Sign of Honor Standards printing office 123557 Moscow GSP
Novopresmeiski Passage 3
Kaluzhskaja Standards Typesetting Office Moscow St. 256 Order No. 413 DSP