FEDERAL AGENCY
ON TECHNICAL REGULATION AND METROLOGY

NATIONAL STANDARD OF THE RUSSIAN FEDERATION

GOST R 34.13–2015

Information technology

CRYPTOGRAPHIC DATA SECURITY

Modes of operation for block ciphers

English Version

Moscow
Standartinform
2015

Foreword

1 DEVELOPED by the Center for Information Protection and Special Communications of the Federal Security Service of the Russian Federation with participation of the Open Joint-Stock company “Information Technologies and Communication Systems” (JSC “InfoTeCS”)

2 SUBMITTED by the Technical Committee for Standardization TC 26 “Cryptography and security mechanisms”

3 APPROVED AND PUT INTO EFFECT under Decree No 750-st of the Federal Agency on Technical Regulation and Metrology dated June 19, 2015

4 REPLACES GOST R ISO/IEC 10116-93

Application of this Standard is ruled by GOST R 1.0-2012 (Section 8). Information on amendments to this Standard (as of January 1 of the current year) is published in National Standards annual information index. The text of revisions and amendments is published in National Standards monthly information indices. In case of revision (replacement) or cancellation of this Standard, a corresponding notification is published in National Standards monthly information index. Any corresponding information, notifications, and texts are also available in the public information system — on the official web-site of the Federal Agency on Technical Regulation and Metrology on the Internet (www.gost.ru).

Contents

Introduction

This Standard specifies modes of operation for block ciphers. These modes provide protection of data confidentiality, authenticity, and integrity for messages of variable length.

This Standard replaces national standard GOST R ISO/IEC 10116-93 “Information Technology. Modes of Operation for an *n*-bit Block Cipher”. The need for the development of this Standard was determined by the demand for defining modes of operation for block ciphers that meet modern requirements for cryptographic strength.

The terms and notions of this Standard comply with international standards ISO/IEC 9797-1 [1], ISO/IEC 10116 [2], ISO/IEC 10118-1 [3], ISO/IEC 18033 [4], ISO/IEC 14888-1 [5].

N o t e – The main part of this Standard is supplemented with Annex A.

NATIONAL STANDARD OF THE RUSSIAN FEDERATION

Information technology
CRYPTOGRAPHIC DATA SECURITY
Modes of operation for block ciphers

Effective date — 2016—01—01

1 Scope

Modes of operation for block ciphers specified in this Standard are recommended to use for developing, manufacturing, operation and modernization of cryptographic solutions intended for different kinds of information systems.

This Standard should be applied in case the confidential information must be protected according to legislation of the Russian Federation.

2 Terms, definitions, and symbols

For the purposes of this Standard, the following terms and definitions apply.

2.1 Terms and definitions

2.1.1

encryption algorithm: process which transforms plaintext into ciphertext
[ISO/IEC 18033–1, clause 2.19]

2.1.2

decryption algorithm: process which transforms ciphertext into plaintext
[ISO/IEC 18033–1, clause 2.14]

2.1.3

basic block cipher: block cipher which for a given key provides one particular invertible mapping of the set of fixed-length plaintext blocks into the set of ciphertext blocks of the same length

Official Publication

2.1.4

block: string of bits of a defined length
[ISO/IEC 18033–1, clause 2.6]

2.1.5

block cipher: symmetric cryptographic technique with the property that the encryption algorithm operates on a block of plaintext to yield a block of ciphertext
[ISO/IEC 18033–1, clause 2.7]

2.1.6

padding: appending extra bits to a string of bits
[ISO/IEC 10118–1, clause 3.9]

2.1.7

block chaining: encryption of information in such a way that each block of ciphertext is cryptographically dependent upon a preceding ciphertext block
[ISO/IEC 10116, clause 3.1]

2.1.8

encryption: reversible transformation of data by a cipher to produce ciphertext from plaintext
[ISO/IEC 18033–1, clause 2.18]

2.1.9

message authentication code (MAC): fixed-length string of bits computed by applying a symmetric cryptographic technique to the message and appended to that message in order to provide its integrity and authenticity of the data source.
[ISO/IEC 9797-1, clauses 3.9, 3.10]

2.1.10

key: sequence of symbols that controls the operation of a cryptographic transformation
[ISO/IEC 18033–1, clause 2.21]

N o t e – In this Standard, keys in the form of sequences of bits are only considered.

2.1.11

starting variable: variable possibly derived from some initializing value and used in defining the starting point of the modes of operation of a block cipher
[ISO/IEC 10116, clause 3.12]

2.1.12

plaintext: unencrypted information
[ISO/IEC 10116, clause 3.11]

2.1.13

decryption: reversal of a corresponding encryption
[ISO/IEC 18033-1, clause 2.13]

2.1.14

symmetric cryptographic technique: cryptographic technique that uses the same key for both the originator's and recipient's transformation

[ISO/IEC 18033–1, clause 2.32]

2.1.15

initializing value: value transmitted over the communication channel and used in defining the starting variable

2.1.16

message: string of bits of any finite length

[ISO/IEC 14888–1 clause 3.10]

2.1.17

counter: string of bits of length equal to the cipher block length which is used in the counter mode

[ISO/IEC 10116, clause 3.4]

2.1.18

cipher: cryptographic technique used to protect the confidentiality of data, and which consists of both encryption and decryption algorithms

[ISO/IEC 18033–1, clause 2.20]

2.1.19

ciphertext: data which has been transformed from plaintext to hide its information content

[ISO/IEC 10116, clause 3.3]

2.2 Symbols

For the purposes of this Standard, the following symbols apply.

V^*	the set of all binary vector-strings of finite length (hereinafter referred to as strings), including an empty string;
V_s	the set of all binary strings of length s, where s is a non-negative integer; substrings and string components are enumerated from right (lower order) to left (higher order) starting from zero;
$\|A\|$	the number of components (the length) of $A \in V^*$; if A is an empty string, then $\|A\| = 0$;
$A\|\|B$	concatenation of the strings A, $B \in V^*$, i.e. a string from $V_{\|A\| + \|B\|}$, where the substring with higher order components from $V_{\|A\|}$ is equal to A, and the substring with lower order components from $V_{\|B\|}$ is equal to B;
0^r	a string consisting of r zero bits;
$\oplus$	bitwise addition modulo 2 of two binary strings of the same length;
$\mathbb{Z}_{2^s}$	the integer residue ring modulo 2^s;
$\boxplus_s$	the addition operation in $\mathbb{Z}_{2^s}$;

$x \bmod \ell$	a remainder of the division of the integer x by the positive integer ℓ;
$\mathrm{MSB}_s\colon V^* \setminus \bigcup_{i=0}^{s-1} V_i \to V_s$	the mapping associating a string $z_{m-1}\|\| \ldots \|\|z_1\|\|z_0$, $m \geq s$, with the string $z_{m-1}\|\| \ldots \|\|z_{m-s+1}\|\|z_{m-s}$, $z_i \in V_1$, $i = 0, 1, \ldots, m-1$;
$\mathrm{LSB}_s\colon V^* \setminus \bigcup_{i=0}^{s-1} V_i \to V_s$	the mapping associating a string $z_{m-1}\|\| \ldots \|\|z_1\|\|z_0$, $m \geq s$, with the string $z_{s-1}\|\| \ldots \|\|z_1\|\|z_0$, $z_i \in V_1$, $i = 0, 1, \ldots, m-1$;
$A \ll r$	a string obtained from A by shifting it by r positions in the direction of higher order components. If $A \in V_s$, then $A \ll r \in V_s$, where: $A \ll r = \begin{cases} \mathrm{LSB}_{s-r}(A) \,\|\|\, 0^r, & \text{if } r < s, \\ 0^s, & \text{if } r \geq s; \end{cases}$
$\mathrm{Poly}_s\colon V^s \to GF(2)[x]$	the mapping associating a string $z=(z_{s-1}\|\| \ldots \|\|z_0) \in V_s$ with the polynomial $\mathrm{Poly}_s(z) = \sum_{i=0}^{s-1} z_i x^i$;
$\mathrm{Vec}_s\colon \mathbb{Z}_{2^s} \to V_s$	the bijective mapping which for an integer from $\mathbb{Z}_{2^s}$ puts into correspondence its binary representation, i.e. for any $z \in \mathbb{Z}_{2^s}$ represented as $z = z_0 + 2 \cdot z_1 + \ldots + 2^{s-1} \cdot z_{s-1}$, where $z_i \in \{0, 1\}$, $i = 0, 1, \ldots, s-1$, the equality $\mathrm{Vec}_s(z) = z_{s-1}\|\|\ldots\|\|z_1\|\|z_0$ holds;
$\mathrm{Int}_s\colon V_s \to \mathbb{Z}_{2^s}$	the mapping inverse to the mapping, Vec_s i.e. $\mathrm{Int}_s = \mathrm{Vec}_s^{-1}$;
k	the length (in bits) of the block cipher key;
n	the block length (in bits) of a block cipher;
$\mathrm{E}\colon V_n \times V_k \to V_n$	the mapping which implements the basic block cipher, and transforms the plaintext block $P \in V_n$ with the key $K \in V_k$ into the ciphertext block $C \in V_n$: $\mathrm{E}(P,K)=C$;
$\mathrm{e}_K\colon V_n \to V_n$	the encryption function with the key $K \in V_k$, i.e. $\mathrm{e}_K(P)=\mathrm{E}(P,K)$ for all $P \in V_n$;
$\mathrm{d}_K\colon V_n \to V_n$	the decryption function with the key $K \in V_k$, i.e. $\mathrm{d}_K=\mathrm{e}_K^{-1}$.

3 General

This Standard defines the following modes of operation for block ciphers:

- Electronic Codebook (ECB) mode;
- Counter Mode (CTR) mode;
- Output Feedback (OFB);
- Cipher Block Chaining (CBC) mode;
- Cipher Feedback (CFB) mode;
- Message Authentication Code (MAC) algorithm.

These modes could be applied for block ciphers with arbitrary block length n.

4 Auxiliary procedures

4.1 Padding

Some modes of operation (CTR, OFB, CFB) may provide cryptographic transformation of messages of arbitrary length. For the other modes (ECB, CBC) the message length should be a multiple of a given value ℓ. In the latter case, for messages of arbitrary length padding is required. This Standard specifies three padding methods.

Let $P \in V^*$ be the original message to encrypt.

4.1.1 Method 1

Let $|P| \equiv r \bmod \ell$. Set

$$P^* = \begin{cases} P, & \text{if } r = 0, \\ P \,||\, 0^{\ell-r}, & \text{otherwise.} \end{cases}$$

N o t e. In some cases, the specified method does not provide unambiguous recovery of the original message. For example, the padded versions of P_1 such that $|P_1| = \ell \cdot q - 1$ (for a certain q) and of $P_2 = P_1 || 0$ are the same. In this case, to provide unambiguous message recovery the length of the original message must be available.

4.1.2 Method 2

Let $|P| \equiv r \bmod \ell$. Set

$$P^* = P \,||\, 1 \,||\, 0^{\ell-r-1}.$$

N o t e. This method provides unambiguous recovery of the original message. If the length of the original message is a multiple of ℓ, the length of the padded message shall be increased.

4.1.3 Method 3

Let $|P| \equiv r \bmod \ell$.

Depending on the value of r, the following cases are possible:

- if $r = n$, then the last block remains unchanged: $P^* = P$;
- if $r < n$, then padding Method 2 is applied.

N o t e. This method shall be used with the MAC algorithm specified in this Standard (see Clause 5.6) and is not recommended for the other modes (see Clauses 5.1-5.5).

N o t e. The choice of a particular padding method is left to the information system developer and/or is regulated by other normative documents.

4.2 Starting variable derivation

Some modes of operation use starting variables derived from the initializing value *IV*. Let I_m: $V_{|IV|} \rightarrow V_m$ be a starting variable derivation procedure, which is called the initialization procedure, where *m* is the length of the staring variable. The initialization procedure is called trivial if $\mathrm{I}_{|IV|} = IV$. Unless stated otherwise, the trivial initialization procedure is assumed with the initializing value of suitable length.

None of the modes of operation specified in this Standard require the initializing value to be kept secret. However, the initializing value generation procedure shall comply with one of the following requirements.

- Initializing values for the CBC and CFB modes shall be chosen randomly, uniformly and independently of each other from the set of all possible values. In other words, the value of *IV* must be unpredictable (random or pseudorandom). This means that given any number of *IV*s the value of a new *IV* cannot be determined with probability exceeding $2^{-|IV|}$.
- All of the initializing values used for encryption under the same key in the CTR mode shall be unique, i.e. all of them shall be pairwise distinct. A deterministic counter can be used to generate initializing values in this case.
- Initializing values for the OFB mode shall be either unpredictable (random or pseudorandom) or unique.

N o t e. The ECB mode does not assume initializing values.

4.3 Truncation procedure

In some modes of operation, strings of *n* bits may be truncated to strings of *s* bits, $s \le n$, by the function $\mathrm{T}_s = \mathrm{MSB}_s$. Therefore the truncation procedure takes higher order (leftmost) bits.

5 Modes of operation for block ciphers

5.1 Electronic Codebook (ECB) mode

The bit length of the messages encrypted in the ECB mode shall be a multiple of the block length *n* of the basic block cipher. Therefore, padding shall be applied to the original message prior to encryption if necessary.

Encryption (decryption) in the ECB mode is the encryption (decryption) of each message block with the basic block cipher.

5.1.1 Encryption

The (padded if needed) plaintext $P \in V^*$, $|P| = n \cdot q$, is spit into blocks: $P = P_1||P_2|| \ldots ||P_q$. The ciphertext blocks are calculated as follows:

$$C_i = e_K(P_i), \quad i=1,\ldots,q. \tag{1}$$

The resulting ciphertext is

$$C = C_1||C_2||\ldots||C_q.$$

The encryption procedure in the ECB mode is shown in Figure 1.

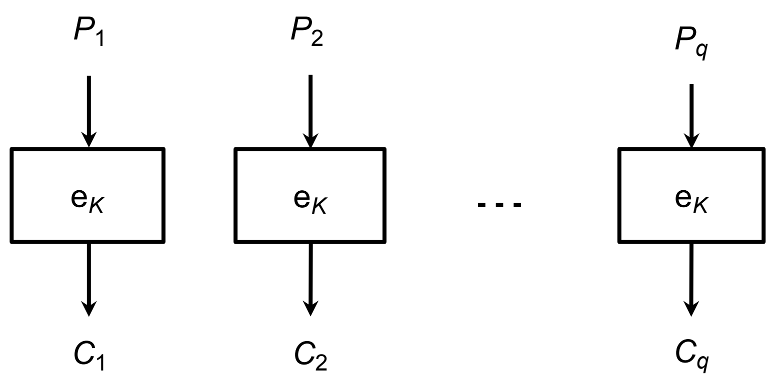

Fig. 1 – Encryption in ECB mode

5.1.2 Decryption

The ciphertext is split into blocks: $C = C_1||C_2||\ldots||C_q$, $C_i \in V_n$, $i = 1, 2, \ldots, q$. The plaintext blocks are calculated as follows:

$$P_i = d_K(C_i), \quad i=1,\ldots,q. \tag{2}$$

The padded plaintext is

$$P = P_1||P_2||\ldots||P_q.$$

N o t e. If the original plaintext was padded prior to encryption, the inverse procedure shall be applied after decryption. For unambiguous message recovery the length of the original message may be required.

The decryption procedure in the ECB mode is shown in Figure 2.

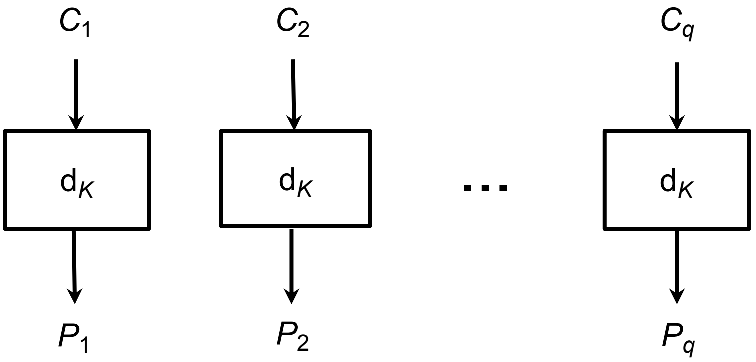

Fig. 2 – Decryption in ECB mode

5.2 Counter (CTR) mode

The CTR mode of operation is defined by an integer parameter s, $0 < s \leq n$.

The CTR mode does not require padding of the plaintext.

For each plaintext to be encrypted under a particular key a unique initializing value $IV \in V_{\frac{n}{2}}$ is required.

Encryption in the CTR mode is performed by bitwise addition modulo 2 of the plaintext with a keystream which is generated block-wise by s bits. The keystream is produced by encrypting a sequence of counter values $CTR_i \in V_n$, i=1, 2,…, by the basic block cipher followed by truncation. The starting variable is $CTR_1 = \mathrm{I}_n(IV) = IV \,||\, 0^{\frac{n}{2}}$. The sequence of counter values is generated by the function Add: $V_n \rightarrow V_n$ as follows:

$$CTR_{i+1} = \mathrm{Add}(CTR_i) = \mathrm{Vec}_n(\mathrm{Int}_n(CTR_i) \boxplus_n 1). \qquad (3)$$

5.2.1 Encryption

The plaintext $P \in V^*$ is split into blocks: $P = P_1||P_2||...||P_q$, $P_i \in V_s$, i = 1, 2, …, q-1, $P_q \in V_r$, $r \leq s$. The ciphertext blocks are calculated as follows:

$$\begin{cases} C_i = P_i \oplus \mathrm{T}_s(e_K(CTR_i)), & i = 1, 2, \ldots, q\text{-}1, \\ C_q = P_q \oplus \mathrm{T}_r(e_K(CTR_q)). \end{cases} \qquad (4)$$

The resulting ciphertext is:

$$C = C_1||C_2||...||C_q.$$

The encryption procedure in the CTR mode is shown in Figure 3.

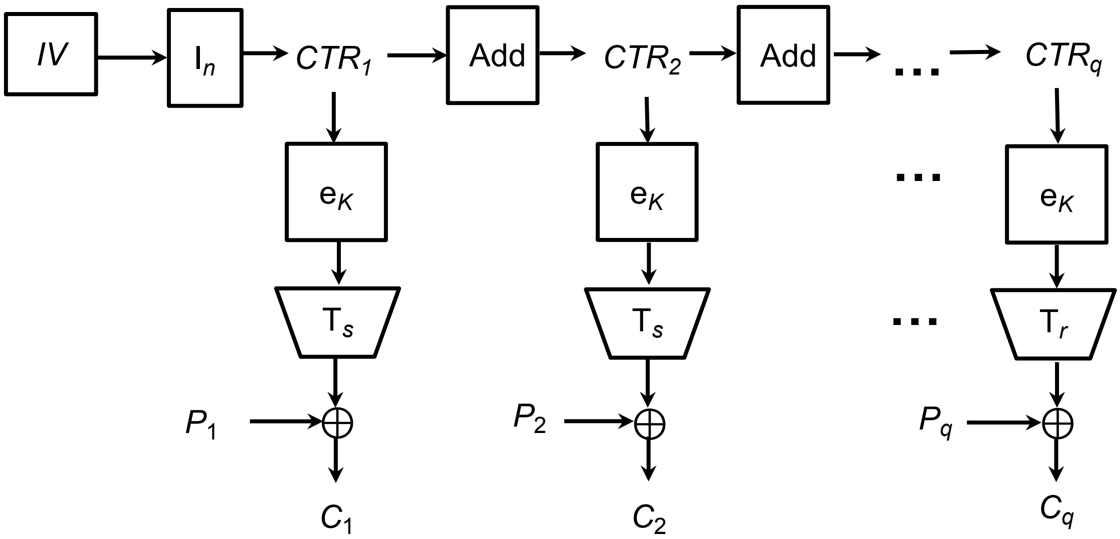

Fig. 3 – Encryption in CTR mode

5.2.2 Decryption

The ciphertext is split into blocks $C = C_1||C_2||...||C_q$, $C_i \in V_s$, $i = 1, 2, \ldots, q\text{-}1$, $C_q \in V_r$, $r \le s$. The plaintext blocks are calculated as follows:

$$\begin{cases} P_i = C_i \oplus \mathrm{T}_s(e_K(CTR_i)),\ i = 1, 2, \ldots, q\text{-}1, \\ P_q = C_q \oplus \mathrm{T}_r(e_K(CTR_q)). \end{cases} \quad (5)$$

The original plaintext is

$$P = P_1||P_2||...||P_q.$$

The decryption procedure in the CTR mode is shown in Figure 4.

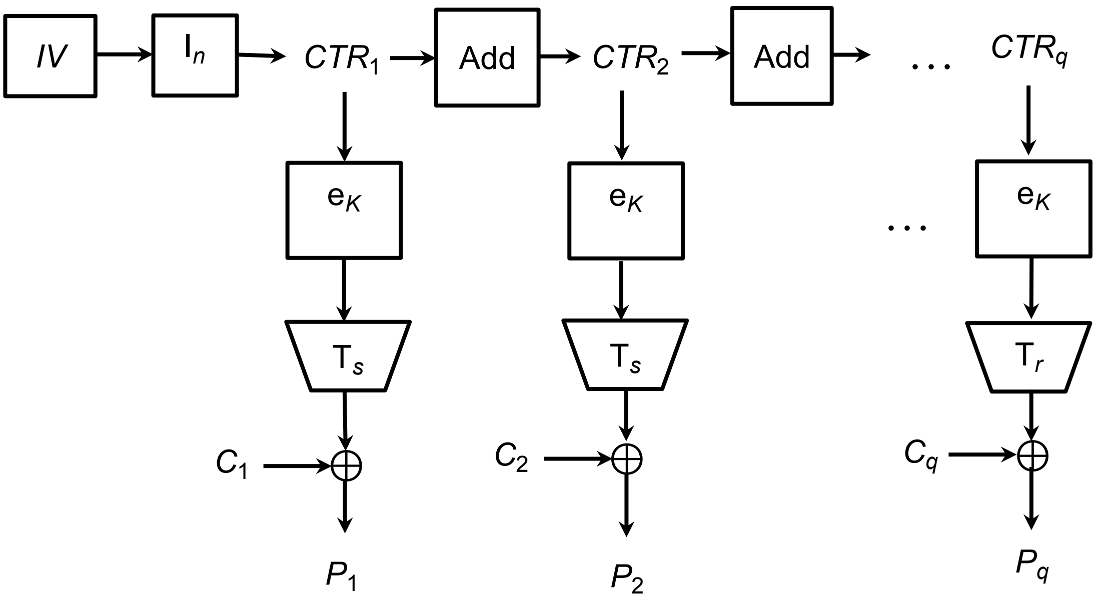

Fig. 4 – Decryption in CTR mode

5.3 Output Feedback (OFB) mode

The OFB mode of operation is defined by integer parameters s and m, $0 < s \le n$, $m = n \cdot z$, where $z \ge 1$ is also an integer.

The OFB mode does not require padding of the plaintext.

For each plaintext to be encrypted under a particular key a unique or unpredictable (random or pseudorandom) initializing value $IV \in V_m$ shall be used.

Encryption in the OFB mode is implemented using a binary shift register R of m bits. The starting variable loaded in the register is the initializing value $IV \in V_m$.

Encryption in the OFB mode is performed by bitwise addition modulo 2 of the plaintext with a keystream which is generated block-wise by s bits. While computing a block of keystream n higher order bits of the shift register are encrypted by the basic block cipher. Then the register is shifted by n positions in the direction of higher order bits, and the output of the basic block cipher is inserted in the lower order positions of the register. The block of keystream is obtained by truncating the output of the basic block cipher.

5.3.1 Encryption

The plaintext $P \in V^*$ is split into blocks: $P = P_1||P_2||...||P_q$, $P_i \in V_s$, $i = 1, 2, \ldots, q\text{-}1$, $P_q \in V_r$, $r \le s$. The ciphertext blocks are calculated as follows:

$$R_1 = IV,$$

$$\begin{cases} Y_i = e_K(\mathrm{MSB}_n(R_i)), \\ C_i = P_i \oplus \mathrm{T}_s(Y_i), \\ R_{i+1} = \mathrm{LSB}_{m-n}(R_i)||Y_i, \end{cases} \quad i = 1, 2, \ldots, q\text{-}1, \qquad (6)$$

$$Y_q = e_K(\mathrm{MSB}_n(R_q)),$$

$$C_q = P_q \oplus \mathrm{T}_r(Y_q).$$

The resulting ciphertext is:

$$C = C_1||C_2||...||C_q.$$

The encryption procedure in the OFB mode is shown in Figure 5.

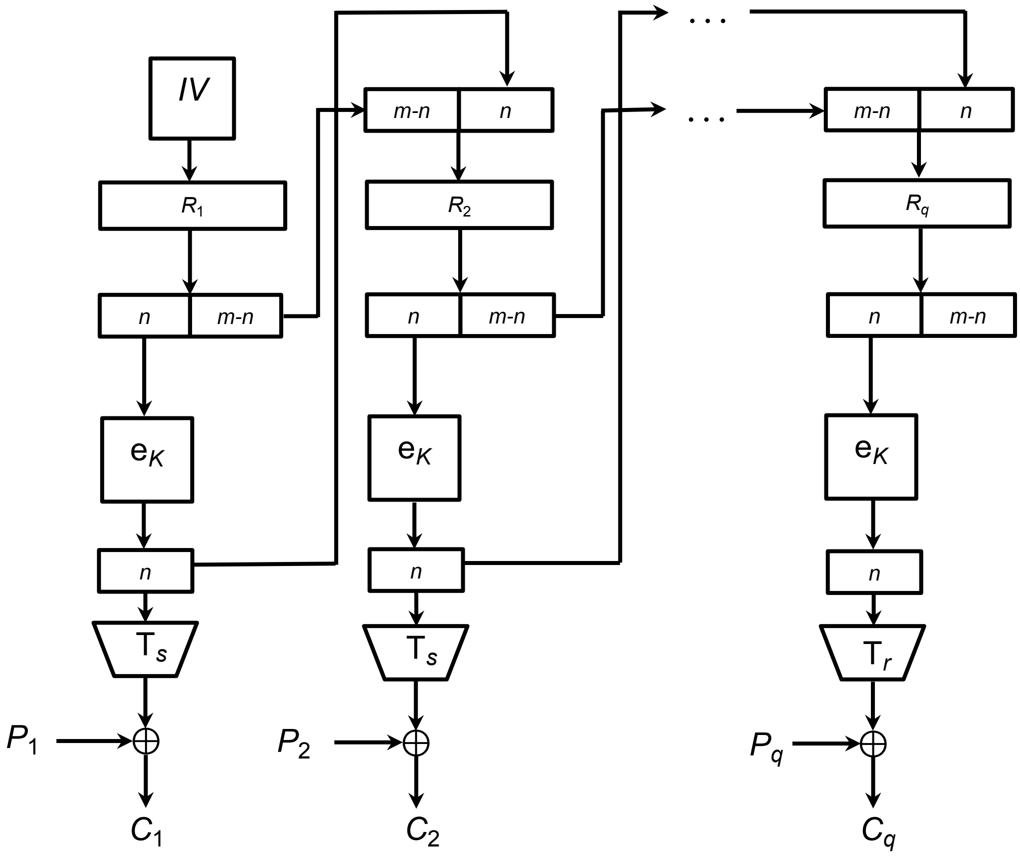

Fig. 5 – Encryption in OFB mode

5.3.2 Decryption

The ciphertext is split into blocks $C = C_1||C_2||...||C_q$, $C_i \in V_s$, $i = 1, 2, \ldots, q\text{-}1$, $C_q \in V_r$, $r \leq s$.

The plaintext blocks are calculated as follows:

$$R_1 = IV,$$

$$\begin{cases} Y_i = \mathrm{e}_K(\mathrm{MSB}_n(R_i)), \\ P_i = C_i \oplus \mathrm{T}_s(Y_i), \\ R_{i+1} = \mathrm{LSB}_{m-n}(R_i)||Y_i, \end{cases} \qquad i = 1, 2, \ldots, q\text{-}1, \qquad (7)$$

$$Y_q = \mathrm{e}_K(\mathrm{MSB}_n(R_q)),$$

$$P_q = C_q \oplus \mathrm{T}_r(Y_q)$$

The original plaintext is

$$P = P_1||P_2||...||P_q.$$

The decryption procedure in the OFB mode is shown in Figure 6.

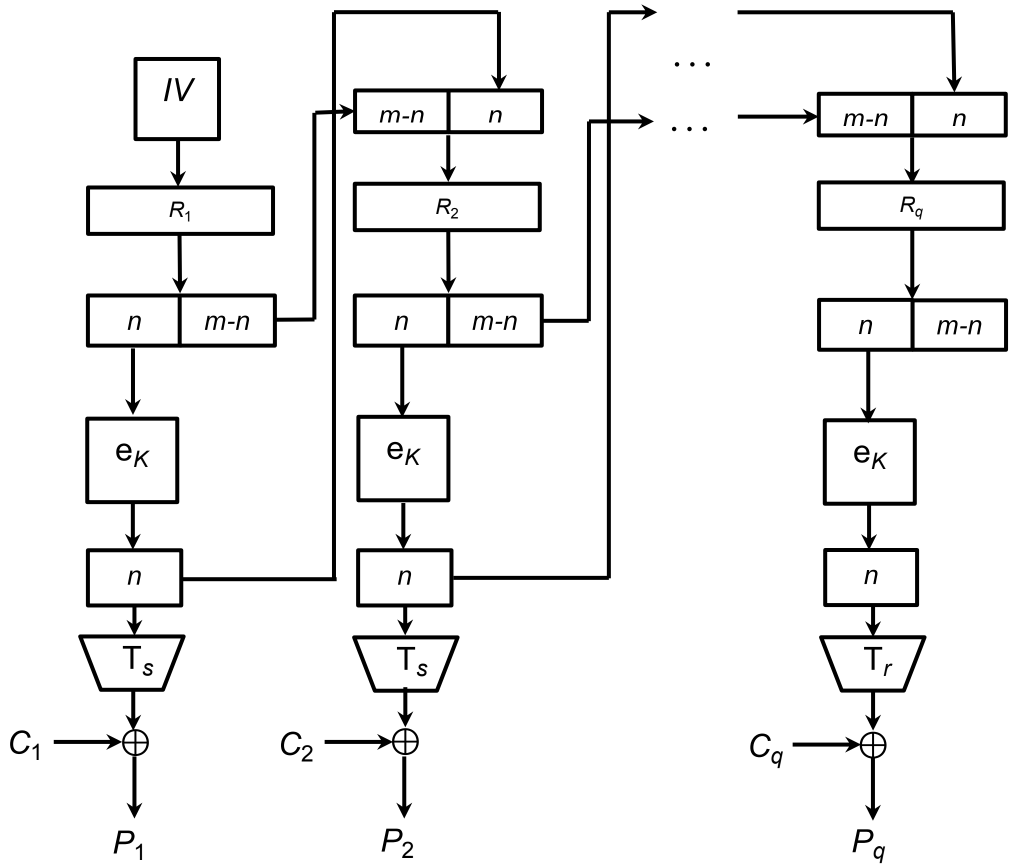

Fig. 6 – Decryption in OFB mode

5.4 Cipher Block Chaining (CBC) mode

The CBC mode of operation is defined by an integer parameter m, $m = n \cdot z$, where $z \geq 1$ is also an integer.

The bit length of the messages encrypted in CBC mode shall be a multiple of the block length n of the basic block cipher. Therefore, padding shall be applied to the original message prior to encryption if necessary.

For each plaintext to be encrypted under a particular key an unpredictable (random or pseudorandom) initializing value $IV \in V_m$ shall be used.

Encryption in CBC mode is implemented using a binary shift register R of m bits. The starting variable loaded in the register is the initializing value IV.

In the CBC mode the ciphertext block is obtained by encrypting the bitwise sum modulo 2 of the plaintext block and n higher order bits of the shift register. Then the register is shifted by n positions in the direction of higher order bits, and n bits of the ciphertext are inserted in the lower order positions of the register.

5.4.1 Encryption

The (padded if needed) plaintext $P \in V^*$ is spit into blocks: $P = P_1||P_2|| \dots ||P_q$. The ciphertext blocks are calculated as follows:

$$R_1 = IV,$$

$$\begin{cases} C_i = \mathrm{e}_K(P_i \oplus \mathrm{MSB}_n(R_i)), \\ R_{i+1} = \mathrm{LSB}_{m-n}(R_i) \| C_i, \end{cases} \quad i = 1, 2, \ldots, q-1, \qquad (8)$$

$$C_q = \mathrm{e}_K(P_q \oplus \mathrm{MSB}_n(R_q)).$$

The resulting ciphertext is

$$C = C_1 \| C_2 \| \ldots \| C_q.$$

The encryption procedure in the CBC mode is shown in Figure 7.

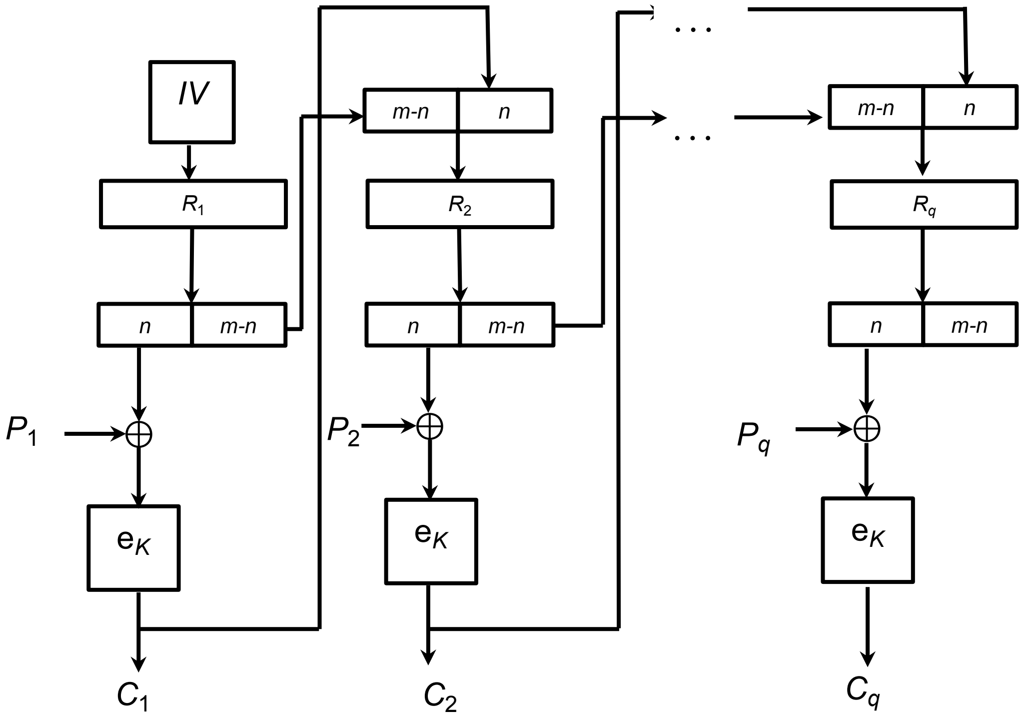

Fig. 7 – Encryption in CBC mode

5.4.2 Decryption

The ciphertext is split into blocks: $C = C_1 \| C_2 \| \ldots \| C_q$, $C_i \in V_n$, $i = 1, 2, \ldots, q$. The plaintext blocks are calculated as follows:

$$R_1 = IV,$$

$$\begin{cases} P_i = \mathrm{d}_K(C_i) \oplus \mathrm{MSB}_n(R_i), \\ R_{i+1} = \mathrm{LSB}_{m-n}(R_i) \| C_i, \end{cases} \quad i = 1, 2, \ldots, q-1, \qquad (9)$$

$$P_q = \mathrm{d}_K(C_q) \oplus \mathrm{MSB}_n(R_q).$$

The padded plaintext is:

$$P = P_1 \| P_2 \| \ldots \| P_q.$$

N o t e. If the original plaintext was padded prior to encryption, the inverse procedure shall be applied after decryption. For unambiguous message recovery the length of the original message may be required.

The decryption procedure in the CBC mode is shown in Figure 8.

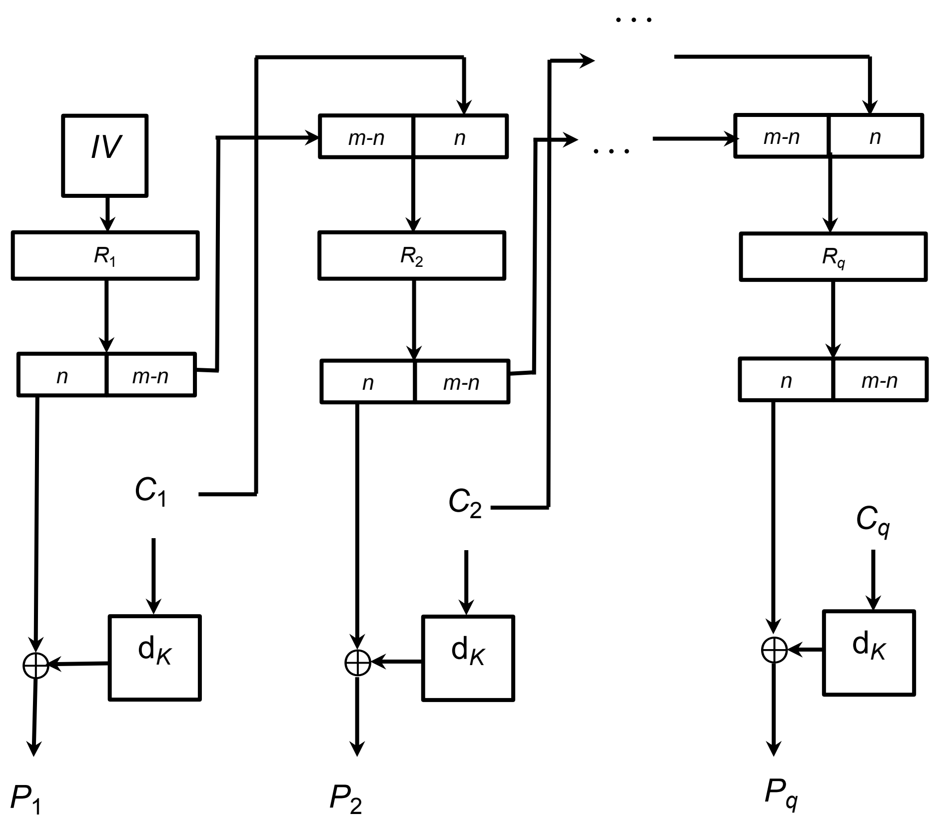

Fig. 8 – Decryption in CBC mode

5.5 Cipher Feedback (CFB) mode

The CFB mode of operation is defined by integer parameters s and m, $0 < s \le n$, $n \le m$.

Depending on operation conditions, the length of the message P may be either constrained by $|P|=s \cdot q$ or unconstrained. If it is constrained, then padding shall be applied to the original message prior to encryption.

For each plaintext to be encrypted under a particular key an unpredictable (random or pseudorandom) initializing value $IV \in V_m$ shall be used.

Encryption in the CFB mode is implemented using a binary shift register R of m bits. The starting variable loaded in the register is the initializing value IV.

Encryption in the CFB mode is performed by bitwise addition modulo 2 of the plaintext with a keystream which is generated block-wise by s bits. While computing a block of keystream n higher order bits of the shift register are encrypted by the basic block cipher followed by truncation. Then the register is shifted by *s* positions in the direction of higher order bits. The lower order positions of the register are then filled with a ciphertext block which is the bitwise sum modulo 2 of the block of keystream and the block of plaintext.

5.5.1 Encryption

The plaintext $P \in V^*$ is split into blocks: $P = P_1||P_2||...||P_q$, $P_i \in V_s$, $i = 1, 2, \ldots, q\text{-}1$, $P_q \in V_r$, $r \leq s$. The ciphertext blocks are calculated as follows:

$$R_1 = IV,$$

$$\begin{cases} C_i = P_i \oplus \mathrm{T}_s(\mathrm{e}_K(\mathrm{MSB}_n(R_i))), \\ R_{i+1} = \mathrm{LSB}_{m-s}(R_i)||C_i, \end{cases} \quad i = 1, 2, \ldots, q\text{-}1, \qquad (10)$$

$$C_q = P_q \oplus \mathrm{T}_r(\mathrm{e}_K(\mathrm{MSB}_n(R_q))).$$

The resulting ciphertext is:

$$C = C_1||C_2||...||C_q.$$

The encryption procedure in the CFB mode is shown in Figure 9.

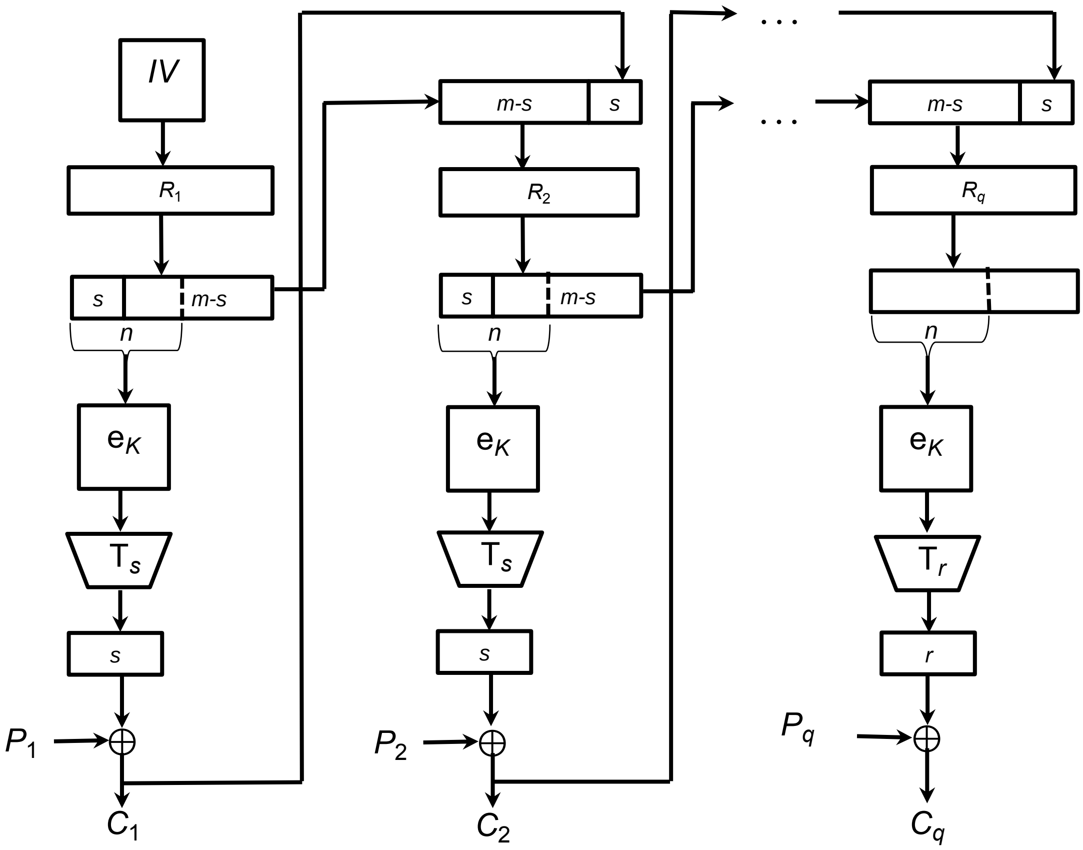

Fig. 9 – Encryption in CFB mode

5.5.2 Decryption

The ciphertext is split into blocks $C = C_1||C_2||...||C_q$, $C_i \in V_s$, $i = 1, 2, \dots, q$-1, $C_q \in V_r$, $r \le s$. The plaintext blocks are calculated as follows:

$$R_1 = IV,$$

$$\begin{cases} P_i = C_i \oplus \mathrm{T}_s(\mathrm{e}_K(\mathrm{MSB}_n(R_i))), \\ R_{i+1} = \mathrm{LSB}_{m-s}(R_i)||C_i, \end{cases} \quad i = 1, 2, \dots, q\text{-}1, \quad (11)$$

$$P_q = C_q \oplus \mathrm{T}_r(\mathrm{e}_K(\mathrm{MSB}_n(R_q))).$$

The original plaintext is:

$$P = P_1||P_2||...||P_q .$$

N o t e. If the original plaintext was padded prior to encryption, the inverse procedure shall be applied after decryption. For unambiguous message recovery the length of the original message may be required.

The decryption procedure in the CFB mode is shown in Figure 10.

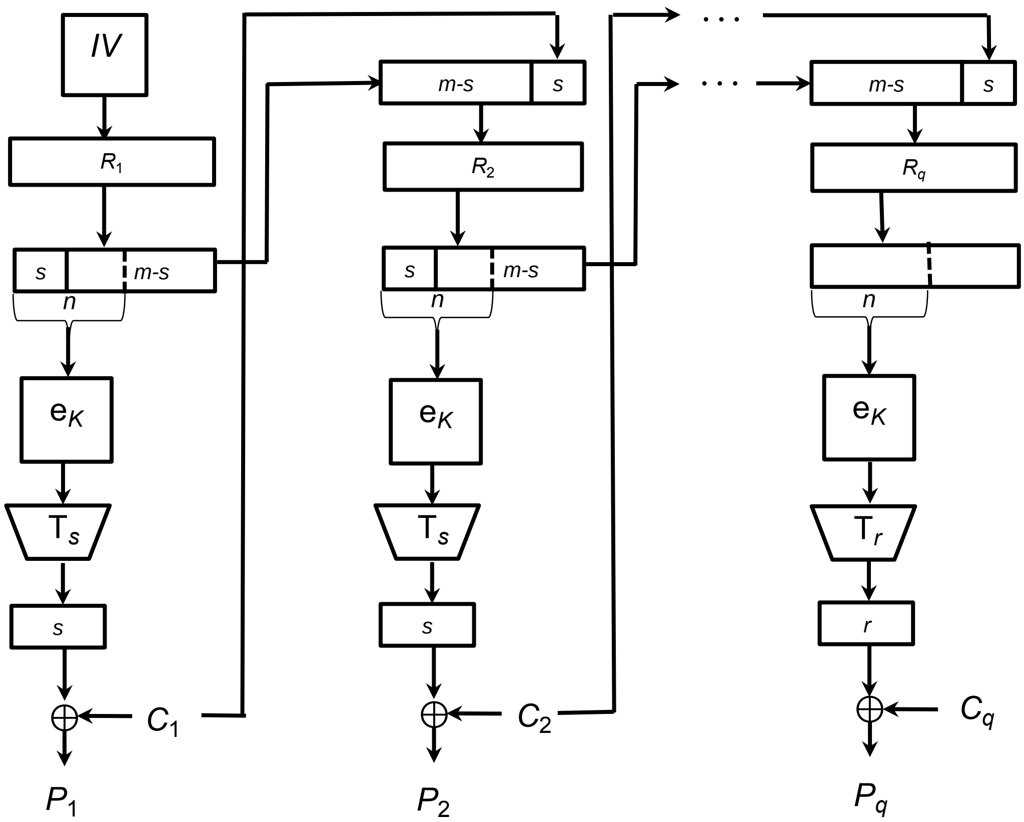

Fig. 10 – Decryption in CFB mode

5.6 Message Authentication Code algorithm

The Message Authentication Code algorithm specified further is commonly known as OMAC1 or CMAC [1].

The MAC algorithm parameter is the length s (in bits) of the MAC, $0 < s \le n$.

5.6.1 Key derivation

The MAC algorithm uses two whitening keys, which are computed using the key K. The length of the whitening keys shall be equal to the block length n of the basic block cipher.

The key derivation procedure may be represented in the following way:

$$R = e_K(0^n);$$

$$K_1 = \begin{cases} R \ll 1, & \text{if } \mathrm{MSB}_1(R) = 0, \\ (R \ll 1) \oplus B_n, & \text{otherwise;} \end{cases}$$

$$K_2 = \begin{cases} K_1 \ll 1, & \text{if } \mathrm{MSB}_1(K_1) = 0, \\ (K_1 \ll 1) \oplus B_n, & \text{otherwise;} \end{cases}$$

where $B_{64} = 0^{59} \,||\, 11011$, $B_{128} = 0^{120} \,||\, 10000111$

If n is different from 64 and 128, the following procedure shall be applied to calculate the value of the B_n constant. Consider a set of primitive polynomials of degree n over GF(2) with minimum number of nonzero coefficients. Arrange this set in lexicographical order with the values of the vectors of coefficients increasing, and denote by $f_n(x)$ the first polynomial in the arranged set.

Consider the field $GF(2^n)[x] / (f_n(x))$, and fix a polynomial basis in it. Denote by the symbol $\otimes$ the multiplication operation in this field. The keys K_1 and K_2 are derived as follows:

$$\begin{cases} R = e_K(0^n), \\ K_1 = \mathrm{Poly}_n^{-1}(\mathrm{Poly}_n(R) \otimes x), \\ K_2 = \mathrm{Poly}_n^{-1}(\mathrm{Poly}_n(R) \otimes x^2). \end{cases} \quad (12)$$

N o t e. The derived keys K_1 and K_2, the intermediate value R along with the key K are secret parameters. Compromising any of these values leads to effective methods of analysis of the entire algorithm.

5.6.2 Message authentication code calculation

The MAC calculation is similar to the cipher block chaining mode for $m = n$ with the shift register initialized to 0^n. The bitwise sum modulo 2 of the plaintext block and the ciphertext block from the previous step is fed to the basic block cipher. The main difference is in the processing of the last plaintext block. The bitwise sum modulo 2 of the last plaintext block, the ciphertext block from the previous step and one of the whitening keys is fed to the basic block cipher. A particular whitening key is selected depending on whether the last block of the original message is complete or not. The MAC value is the result of applying the truncation procedure to the output of the basic block cipher after processing the last block.

The original message $P \in V^*$, for which a MAC must be calculated, is split into blocks:

$$P = P_1||P_2||...||P_q,\ P_i \in V_n\ ,$$

where $i = 1, 2, \ldots, q$-1, $P_q \in V_r$, $r \leq n$.

The message authentication code is calculated as follows:

$$\begin{cases} C_0 = 0^n, \\ C_i = e_K(P_i \oplus C_{i-1}),\ i = 1, 2, \ldots, q-1, \\ MAC = \mathrm{T}_s(\mathrm{e}_K(P_q^* \oplus C_{q-1} \oplus K^*)), \end{cases} \tag{13}$$

where

$$K^* = \begin{cases} K_1, & \text{if } |P_q| = n; \\ K_2, & \text{otherwise.} \end{cases}$$

P_q^* is the last block of the padded message obtained from the original message by padding Method 3.

The MAC calculation is shown in Figures 11 to 13.

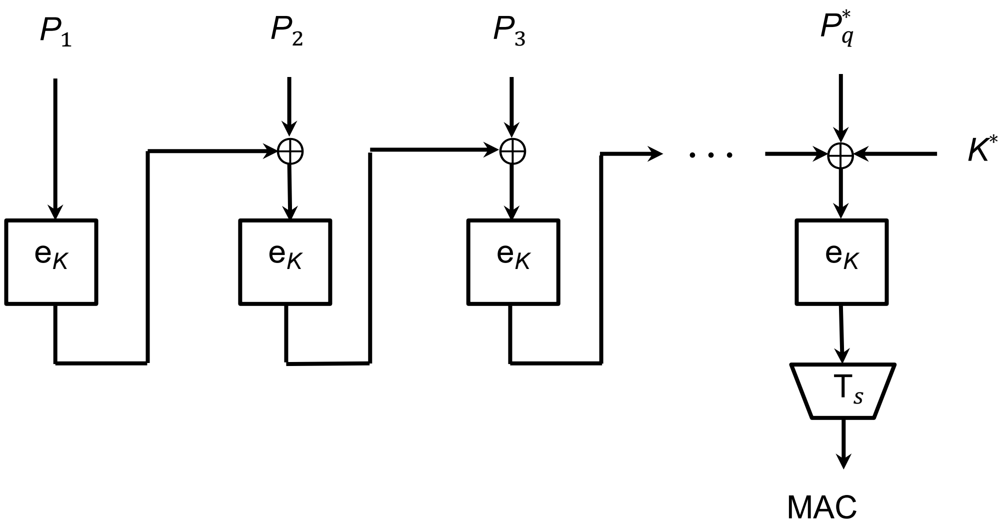

Fig. 11 – MAC algorithm

N o t e. It is highly recommended not to use the same key used in the MAC algorithm for other cryptographic techniques including the modes of operation providing confidentiality specified in Clauses 5.1-5.5.

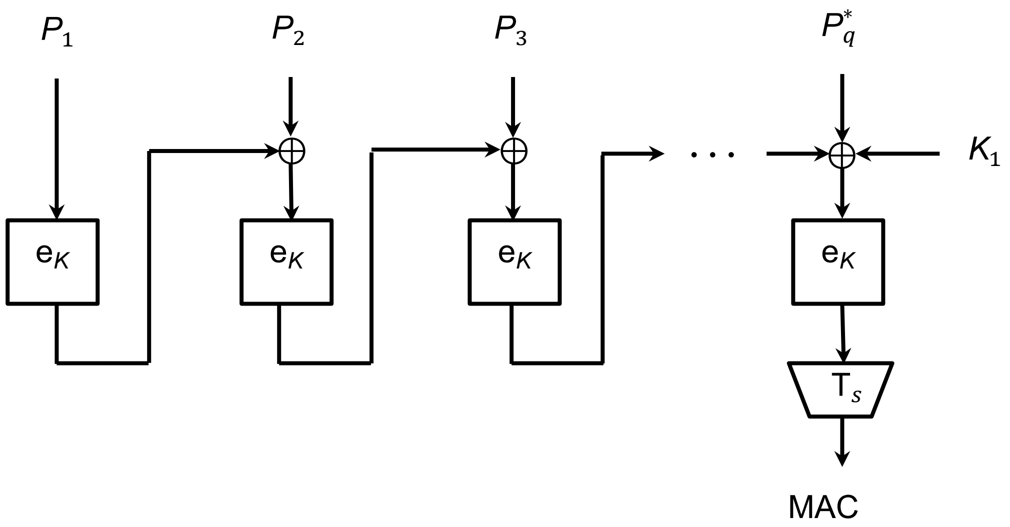

Fig. 12 – MAC algorithm: Last block is complete

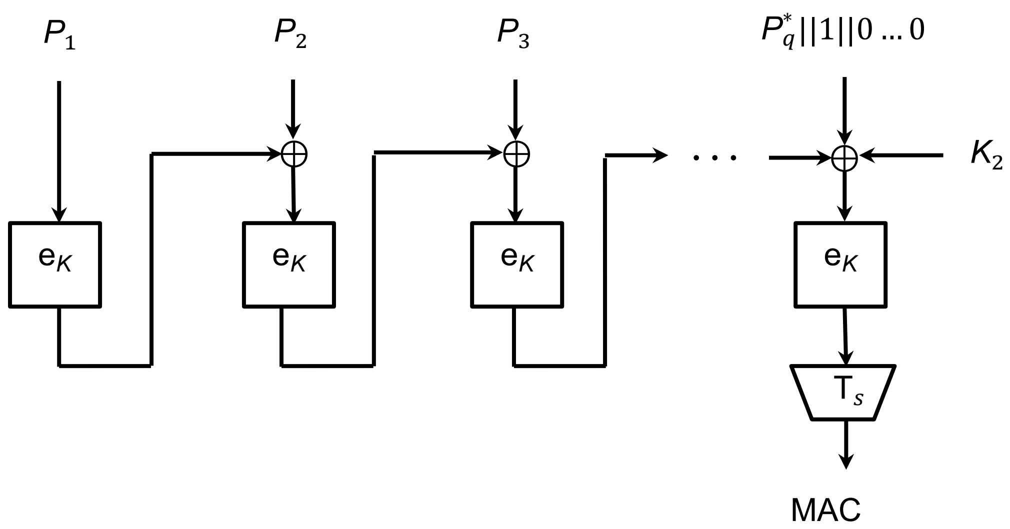

Fig. 13 – MAC algorithm: Last block is padded

Annex A

(informative)

Test examples

This Annex is for information only and is not a normative part of this Standard.

This Annex contains the examples of message encryption and decryption as well as of calculation of a message authentication code by using the modes of operation specified in this Standard.

In this Annex, the parameter *s* is chosen to be equal to *n* in order to simplify the calculations. The parameter *m* depends on a particular mode of operation to demonstrate its features.

In this Annex, binary strings from V^*, whose length is multiple of 4, are expressed in hexadecimal form, while the concatenation symbol ("||") is omitted. That is, a string $a \in V_{4r}$ shall be represented as $a_{r-1}a_{r-2}\ldots a_0$, where

$a_i \in \{0,1,2,\ldots,9, \text{a, b, c, d, e, f}\}$, $i=0,\ldots,r-1$.

Annex A.1 contains examples for the block cipher with block length of n = 128 bits (Kuznyechik). Annex A.2 contains examples for the block cipher with block length of n = 64 bits (Magma).

A.1 128-bit Block cipher

In the examples, the following parameters are used:

Key

K = 8899aabbccddeeff0011223344556677fedcba98765432100123456789abcdef.

Plaintext consists of four 64-bit blocks:

P_1 = 1122334455667700ffeeddccbbaa9988,
P_2 = 00112233445566778899aabbcceeff0a,
P_3 = 112233445566778899aabbcceeff0a00,
P_4 = 2233445566778899aabbcceeff0a0011.

A.1.1 ECB mode

Table A.1 – Encryption in ECB mode

Plaintext	Ciphetext
1122334455667700ffeeddccbbaa9988	7f679d90bebc24305a468d42b9d4edcd
00112233445566778899aabbcceeff0a	b429912c6e0032f9285452d76718d08b
112233445566778899aabbcceeff0a00	f0ca33549d247ceef3f5a5313bd4b157
2233445566778899aabbcceeff0a0011	d0b09ccde830b9eb3a02c4c5aa8ada98

A.1.2 CTR mode

A.1.2.1 Encryption

$s = n = 128$,

IV = 1234567890abcef0.

Table A.2 – Encryption in CTR mode

i	1	2
P_i	1122334455667700ffeeddccbbaa9988	00112233445566778899aabbcceeff0a
Input block	1234567890abcef00000000000000000	1234567890abcef00000000000000001
Output block	e0b7ebfa9468a6db2a95826efb173830	85ffc500b2f4582a7ba54e08f0ab21ee
C_i	f195d8bec10ed1dbd57b5fa240bda1b8	85eee733f6a13e5df33ce4b33c45dee4

i	3	4
P_i	112233445566778899aabbcceeff0a00	2233445566778899aabbcceeff0a0011
Input block	1234567890abcef00000000000000002	1234567890abcef00000000000000003
Output block	b4c8dbcfb353195b4c42cc3ddb9ba9a5	e9a2bee4947b322f7b7d1db6dfb7ba62
C_i	a5eae88be6356ed3d5e877f13564a3a5	cb91fab1f20cbab6d1c6d15820bdba73

A.1.2.2 Decryption

The initial values of P_1, P_2, P_3, P_4 are obtained by using the specified values of K, IV, and C and the decryption operation.

A.1.3 OFB mode

A.1.3.1 Encryption

$s = n = 128$, $m = 2n = 256$,

IV = 1234567890abcef0a1b2c3d4e5f0011223344556677889901213141516171819.

Table A.3 – Encryption in OFB mode

i	1	2
P_i	1122334455667700ffeeddccbbaa9988	00112233445566778899aabbcceeff0a
Input block	1234567890abcef0a1b2c3d4e5f00112	23344556677889901213141516171819
Output block	90a2391de4e25c2400f1a49232d0241d	ed4a659440d99cc3072c8b8d517dd9b5
C_i	81800a59b1842b24ff1f795e897abd95	ed5b47a7048cfab48fb521369d9326bf

i	3	4
P_i	112233445566778899aabbcceeff0a00	2233445566778899aabbcceeff0a0011
Input block	90a2391de4e25c2400f1a49232d0241d	ed4a659440d99cc3072c8b8d517dd9b5
Output block	778064e869c6cf3951a55c30fed78013	020dff9500640ef90a92eead099a3141
C_i	66a257ac3ca0b8b1c80fe7fc10288a13	203ebbc066138660a0292243f6903150

A.1.3.2 Decryption

The initial values of P_1, P_2, P_3, P_4 are obtained by using the specified values of K, IV, and C and the decryption operation.

A.1.4 CBC mode

A.1.4.1 Encryption

$m = 2n = 256$,

IV = 1234567890abcef0a1b2c3d4e5f0011223344556677889901213141516171819.

Table A.4 – Encryption in CBC mode

i	1	2
P_i	1122334455667700ffeeddccbbaa9988	00112233445566778899aabbcceeff0a
Input block	316653cc5cdb9f05e5c1e185e5a989a	23256765232defe79a8abeaedaf9e713
Output block	689972d4a085fa4d90e52e3d6d7dcc27	2826e661b478eca6af1e8e448d5ea5ac
C_i	689972d4a085fa4d90e52e3d6d7dcc27	2826e661b478eca6af1e8e448d5ea5ac

i	3	4
P_i	112233445566778899aabbcceeff0a00	2233445566778899aabbcceeff0a0011
Input block	79bb4190f5e38dc5094f95f18382c627	0a15a234d20f643f05a542aa7254a5bd
Output block	fe7babf1e91999e85640e8b0f49d90d0	167688065a895c631a2d9a1560b63970
C_i	fe7babf1e91999e85640e8b0f49d90d0	167688065a895c631a2d9a1560b63970

A.1.4.2 Decryption

The initial values of P_1, P_2, P_3, P_4 are obtained by using the specified values of K, IV, and C and the decryption operation.

A.1.5 CFB mode

A.1.5.1 Encryption

$s = n = 128$, $m = 2n = 256$,

IV = 1234567890abcef0a1b2c3d4e5f0011223344556677889901213141516171819.

Table A.5 – Encryption in CFB mode

i	1	2
P_i	1122334455667700ffeeddccbbaa9988	00112233445566778899aabbcceeff0a
Input block	1234567890abcef0a1b2c3d4e5f00112	23344556677889901213141516171819
Output block	90a2391de4e25c2400f1a49232d0241d	ed4a659440d99cc3072c8b8d517dd9b5
C_i	81800a59b1842b24ff1f795e897abd95	ed5b47a7048cfab48fb521369d9326bf

i	3	4
P_i	112233445566778899aabbcceeff0a00	2233445566778899aabbcceeff0a0011
Input block	81800a59b1842b24ff1f795e897abd95	ed5b47a7048cfab48fb521369d9326bf
Output block	68d09baf09a0fab01d879d82795d32b5	6dcdfa9828e5a57f6de01533bbf1f4c0
C_i	79f2a8eb5cc68d38842d264e97a238b5	4ffebecd4e922de6c75bd9dd44fbf4d1

A.1.5.2 Decryption

The initial values of P_1, P_2, P_3, P_4 are obtained by using the specified values of K, IV, and C and the decryption operation.

.

A.1.6 MAC algorithm

A.1.6.1 Whitening keys derivation

R = 94bec15e269cf1e506f02b994c0a8ea0

$\mathrm{MSB}_1(R) = 1,$

$K_1 = R \ll 1 \oplus B_n$ = 297d82bc4d39e3ca0de0573298151d40 $\oplus$ 87 =

= 297d82bc4d39e3ca0de0573298151dc7

$\mathrm{MSB}_1(K_1) = 0,$

$K_2 = K_1 \ll 1$ = 297d82bc4d39e3ca0de0573298151dc7 << 1 =

= 52fb05789a73c7941bc0ae65302a3b8e,

$|P_4| = n,\ K^* = K_1.$

A.1.6.2 MAC calculation

$s = 64.$

Table A.6 – MAC calculation

i	1	2
P_i	1122334455667700ffeeddccbbaa9988	00112233445566778899aabbcceeff0a
Input block	1122334455667700ffeeddccbbaa9988	7f76bfa3fae94247d2df27f9753a12c7
Output block	7f679d90bebc24305a468d42b9d4edcd	1ac9d976f83636f55ae9ef305e7c90d2

i	3	4
P_i	112233445566778899aabbcceeff0a00	2233445566778899aabbcceeff0a0011
Input block	0bebea32ad50417dc34354fcb0839ad2	1e2a9c1d8cc03bfa0cb340971252fe24
Output block	15645af4a78e50a9abe8db4b754de3f2	336f4d296059fbe34ddeb35b37749c67

MAC = 336f4d296059fbe3.

A.2 64-bit Block cipher

In the examples, the following parameters are used.

Key

K = ffeeddccbbaa99887766554433221100f0f1f2f3f4f5f6f7f8f9fafbfcfdfeff.

Plaintext consists of four 64-bit blocks:

P_1 = 92def06b3c130a59,
P_2 = db54c704f8189d20,
P_3 = 4a98fb2e67a8024c,
P_4 = 8912409b17b57e41.

A.2.1 ECB mode

Table A.7 – Encryption in ECB mode

Plaintext	Ciphertext
92def06b3c130a59	2b073f0494f372a0
db54c704f8189d20	de70e715d3556e48
4a98fb2e67a8024c	11d8d9e9eacfbc1e
8912409b17b57e41	7c68260996c67efb

A.2.2 CTR mode

A.2.2.1 Encryption

$s = n = 64$,

$IV = 12345678$.

Table A.8 – Encryption in CTR mode

i	1	2
P_i	92def06b3c130a59	db54c704f8189d20
Input block	1234567800000000	1234567800000001
Output block	dc46e167aba4b365	e571ca972ef0c049
C_i	4e98110c 97b7b93c	3e250d93d6e85d69

i	3	4
P_i	4a98fb2e67a8024c	8912409b17b57e41
Input block	1234567800000002	1234567800000003
Output block	59f57da6601ad9a3	df9cf61bbce7df6c
C_i	136d868807b2dbef	568eb680ab52a12d

A.2.2.2 Decryption

The initial values of P_1, P_2, P_3, P_4 are obtained by using the specified values of *K, IV,* and *C* and the decryption operation.

A.2.3 OFB mode

A.2.3.1 Encryption

$s = n = 64$, $m = 2n = 128$

IV = 1234567890abcdef234567890abcdef1.

Table A.9 – Encryption in OFB mode

i	1	2
P_i	92def06b3c130a59	db54c704f8189d20
Input block	1234567890abcdef	234567890abcdef1
Output block	49e910895a8336da	d612a348e78295bc
C_i	db37e0e266903c83	0d46644c1f9a089c

i	3	4
P_i	4a98fb2e67a8024c	8912409b17b57e41
Input block	49e910895a8336da	d612a348e78295bc
Output block	ea60cb4c24a63032	4136af23aafaa544
C_i	a0f83062430e327e	c824efb8bd4fdb05

A.2.3.2 Decryption

The initial values of P_1, P_2, P_3, P_4 are obtained by using the specified values of *K*, *IV*, and *C* and the decryption operation.

A.2.4 CBC mode

A.2.4.1 Encryption

$m = 3n = 192$,

IV = 1234567890abcdef234567890abcdef134567890abcdef12.

Table A.10 – Encryption in CBC mode

i	1	2	3	4
P_i	92def06b3c130a59	db54c704f8189d20	4a98fb2e67a8024c	8912409b17b57e41
Input block	80eaa613acb8c7b6	f811a08df2a443d1	7ece83becc65ed5e	1fc3f0c5fddd4758
Output block	96d1b05eea683919	aff76129abb937b9	5058b4a1c4bc0019	20b78b1a7cd7e667
C_i	96d1b05eea683919	aff76129abb937b9	5058b4a1c4bc0019	20b78b1a7cd7e667

A.2.4.2 Decryption

The initial values of P_1, P_2, P_3, P_4 are obtained by using the specified values of K, IV, and C and the decryption operation.

A.2.5 CFB mode

A.2.5.1 Encryption

$s = n = 64$, $m = 2n = 128$,

IV = 1234567890abcdef234567890abcdef1.

Table A.11 – Encryption in CFB mode

i	1	2
P_i	92def06b3c130a59	db54c704f8189d20
Input block	1234567890abcdef	234567890abcdef1
Output block	49e910895a8336da	d612a348e78295bc
C_i	db37e0e266903c83	0d46644c1f9a089c

i	3	4
P_i	4a98fb2e67a8024c	8912409b17b57e41
Input block	db37e0e266903c83	0d46644c1f9a089c
Output block	6e25292d34bdd1c7	35d2728f36b22b44
C_i	24bdd2035315d38b	bcc0321421075505

A.2.5.1 Decryption

The initial values of P_1, P_2, P_3, P_4 are obtained by using the specified values of K, IV, and C and the decryption operation.

A.2.6 MAC algorithm

A.2.6.1 Whitening keys derivation

R = 2fa2cd99a1290a12,

$MSB_1(R) = 0$, $K_1 = R \ll 1$ = 5f459b3342521424,

$MSB_1(K_1) = 0$, hence $K_2 = K_1 \ll 1$= be8b366684a42848

$|P_4|=n, \; K^* = K_1$

A.2.6.2 MAC calculation

$s = 32$.

Table A.12 – MAC calculation

i	1	2
P_i	92def06b3c130a59	db54c704f8189d20
Input block	92def06b3c130a59	f053f8006cebef80
Output block	2b073f0494f372a0	c89ed814fd5e18e9

i	3	4
P_i	4a98fb2e67a8024c	8912409b17b57e41
Input block	8206233a9af61aa5	216e6a2561cff165
Output block	f739b18d34289b00	154e72102030c5bb

MAC = 154e7210.

Bibliography

[1] ISO/IEC 9797-1:2011 Information technology — Security techniques — Message Authentication Codes (MACs) — Part 1: Mechanisms using a block cipher

[2] ISO/IEC 10116:2006 Information technology — Security techniques — Modes of operation for an *n*-bit block cipher

[3] ISO/IEC 10118-1:2000 Information technology — Security techniques — Hash-functions — Part 1: General

[4] ISO/IEC 18033-1:2005 Information technology — Security techniques — Encryption algorithms — Part 1: General

[5] ISO/IEC 14888-1:2008 Information technology — Security techniques — Digital signatures with appendix — Part 1: General

* These International ISO/IEC standards are available at the FSUE *“Standartinform*” of the Federal Agency on Technical Regulation and Metrology.

UDC 681.3.06:006.354 OKS 35. 040 OKSTU 5002 P 85

Keywords: information technology, cryptographic data security, block cipher, modes of operation for block cipher, confidentiality, integrity, message authentication code, keystream, block chaining