GOST R 34.12-2015: Block Cipher "Magma"
draft-dolmatov-magma-05

Abstract

In addition to a new cipher with block length of n=128 bits (referred to as "Kyznyechik" and described in RFC 7801) Russian Federal standard GOST R 34.12-2015 includes an updated version of the block cipher with block length of n=64 bits and key length k=256 bits, which is also referred to as "Magma". The algorithm is an updated version of an older block cipher with block length of n=64 bits described in GOST 28147-89 (RFC 5830). This document is intended to be a source of information about the updated version of the 64-bit cipher. It may facilitate the use of the block cipher in Internet applications by providing information for developers and users of GOST 64-bit cipher with the revised version of the cipher for encryption and decryption.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on May 20, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.
1. Introduction

The Russian Federal standard [GOSTR3412-2015] specifies basic block ciphers used as cryptographic techniques for information processing and information protection including the provision of confidentiality, authenticity, and integrity of information during information transmission, processing and storage in computer-aided systems.
The cryptographic algorithms defined in this specification are designed both for hardware and software implementation. They comply with modern cryptographic requirements, and put no restrictions on the confidentiality level of the protected information.

This document is intended to be a source of information about the updated version of 64-bit cipher. It may facilitate the use of the block cipher in Internet applications by providing information for developers and users of GOST 64-bit cipher with the revised version of the cipher for encryption and decryption.

2. General Information

The Russian Federal standard [GOSTR3412-2015] was 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" (InfoTeCS JSC). GOST R 34.12-2015 was approved and introduced by Decree #749 of the Federal Agency on Technical Regulating and Metrology on 19.06.2015.

Terms and concepts in the specification comply with the following international standards:

- ISO/IEC 10116 [ISO-IEC10116],

3. Definitions and Notations

The following terms and their corresponding definitions are used in the specification.

3.1. Definitions

Definitions

encryption algorithm: process which transforms plaintext into ciphertext (Clause 2.19 of [ISO-IEC18033-1]),

decryption algorithm: process which transforms ciphertext into plaintext (Clause 2.14 of [ISO-IEC18033-1]),

basic block cipher: block cipher which for a given key provides a single invertible mapping of the set of fixed-length plaintext blocks into ciphertext blocks of the same length,
3.2. Notations

The following notations are used in the specification:
4. Parameter Values

4.1. Nonlinear Bijection

The bijective nonlinear mapping is a set of substitutions:

\[ \Pi_i = \text{Vec}_4 \Pi'_i \text{Int}_4: \text{V}_4 \rightarrow \text{V}_4, \]
where

\[ \Pi'_i: \mathbb{Z}_{(2^4)} \rightarrow \mathbb{Z}_{(2^4)}, \ i = 0, 1, \ldots, 7. \]

The values of the substitution \( \Pi' \) are specified below as arrays

\[ \Pi'_i = (\Pi'_i(0), \Pi'_i(1), \ldots, \Pi'_i(15)), \ i = 0, 1, \ldots, 7; \]

\[ \Pi'_0 = (12, 4, 6, 2, 10, 5, 11, 9, 14, 8, 13, 7, 0, 3, 15, 1); \]
\[ \Pi'_1 = (6, 8, 2, 3, 9, 10, 5, 12, 1, 14, 4, 7, 11, 13, 0, 15); \]
\[ \Pi'_2 = (11, 3, 5, 8, 2, 15, 10, 13, 14, 1, 7, 4, 12, 9, 6, 0); \]
\[ \Pi'_3 = (12, 8, 2, 1, 13, 4, 15, 6, 7, 0, 10, 5, 3, 14, 9, 11); \]
\[ \Pi'_4 = (7, 15, 5, 10, 8, 1, 6, 13, 0, 9, 3, 14, 11, 4, 2, 12); \]
\[ \Pi'_5 = (5, 13, 15, 6, 9, 2, 12, 10, 11, 7, 8, 1, 4, 3, 14, 0); \]
\[ \Pi'_6 = (8, 14, 2, 5, 6, 9, 1, 12, 15, 4, 11, 0, 13, 10, 3, 7); \]
\[ \Pi'_7 = (1, 7, 14, 13, 0, 5, 8, 3, 4, 15, 10, 6, 9, 12, 11, 2); \]

4.2. Transformations

The following transformations are applicable for encryption and decryption algorithms:

\[ t: V_{32} \rightarrow V_{32} \]
\[ t(a) = t(a_7||\ldots||a_0) = \Pi_7(a_7)||\ldots||\Pi_0(a_0), \]
where \( a=a_7||\ldots||a_0 \) belongs to \( V_{32} \), \( a_i \) belongs to \( V_4 \), \( i=0, 1, \ldots, 7; \)

\[ g[k]: V_{32} \rightarrow V_{32} \]
\[ g[k](a) = (t(Vec_{32}(Int_{32}(a) [+ Int_{32}(k)))) \lll_{11}, \] where \( k, a \) belong to \( V_{32}; \)

\[ G[k]: V_{32^*}V_{32} \rightarrow V_{32^*}V_{32} \]
\[ G[k](a_1, a_0) = (a_0, g[k](a_0) \ (xor) \ a_1), \] where \( k, a_0, a_1 \) belong to \( V_{32}; \)

\[ G^*[k]: V_{32^*}V_{32} \rightarrow V_{64} \]
\[ G^*[k](a_1, a_0) = (g[k](a_0) \ (xor) \ a_1) || a_0, \] where \( k, a_0, a_1 \) belong to \( V_{32}. \)

4.3. Key Schedule

Round keys \( K_i \) belonging to \( V_{32}, i=1, 2, \ldots, 32 \) are derived from key \( K=K_{255}\ldots||k_0 \) belonging to \( V_{256}, k_i \) belongs to \( V_1, i=0, 1, \ldots, 255, \) as follows:
K_1=k_{255}||...||k_{224};
K_2=k_{223}||...||k_{192};
K_3=k_{191}||...||k_{160};
K_4=k_{159}||...||k_{128};
K_5=k_{127}||...||k_{96};
K_6=k_{95}||...||k_{64};
K_7=k_{63}||...||k_{32};
K_8=k_{31}||...||k_{0};
K_{(i+8)}=K_i, i = 1, 2, ..., 8;
K_{(i+16)}=K_i, i = 1, 2, ..., 8;
K_{(i+24)}=K_{(9-i)}, i = 1, 2, ..., 8.

5. Basic Encryption Algorithm

5.1. Encryption

Depending on the values of round keys K_1,...,K_{32}, the encryption algorithm is a substitution E_{(K_1,...,K_{32})} defined as follows:

E_{(K_1,...,K_{32})}(a)=G^*[K_{32}]G[K_{31}]...G[K_2]G[K_1](a_1, a_0),

where a=(a_1, a_0) belongs to V_{64}, and a_0, a_1 belong to V_{32}.

5.2. Decryption

Depending on the values of round keys K_1,...,K_{32}, the decryption algorithm is a substitution D_{(K_1,...,K_{32})} defined as follows:

D_{(K_1,...,K_{32})}(a)=G^*[K_1]G[K_2]...G[K_{31}]G[K_{32}](a_1, a_0),

where a=(a_1, a_0) belongs to V_{64}, and a_0, a_1 belong to V_{32}.

6. IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

This entire document is about security considerations.

Unlike [RFC5830] (GOST 28147-89), but like [RFC7801] this specification does not define exact block modes which should be used together with updated Magma cipher. One is free to select block modes depending on the protocol and necessity.
8. References

8.1. Normative References


8.2. Informative References


Appendix A. Test Examples

This section is for information only and is not a normative part of the specification.

A.1. Transformation t

\[ t(fdb97531) = 2a196f34, \]
\[ t(2a196f34) = ebd9f03a, \]
\[ t(ebd9f03a) = b039bb3d, \]
\[ t(b039bb3d) = 68695433. \]

A.2. Transformation g

\[ g[87654321](fedcba98) = fdcbc20c, \]
\[ g[fdcbc20c](87654321) = 7e791a4b, \]
\[ g[7e791a4b](fdcbc20c) = c76549ec, \]
\[ g[c76549ec](7e791a4b) = 9791c849. \]

A.3. Key schedule

With key set to

\[ K = ffeeddccbba99887766554433221100f0f1f2f3f4f5f6f7f8f9fafbfcfdeff, \]

following round keys are generated:
\[ K_1 = \text{ffeeeddcc}, \]
\[ K_2 = \text{bbaa9988}, \]
\[ K_3 = 77665544, \]
\[ K_4 = 33221100, \]
\[ K_5 = \text{f0f1f2f3}, \]
\[ K_6 = \text{f4f5f6f7}, \]
\[ K_7 = \text{f8f9fafb}, \]
\[ K_8 = \text{fcfdfeff}, \]
\[ K_9 = \text{ffeeeddcc}, \]
\[ K_{10} = \text{bbaa9988}, \]
\[ K_{11} = 77665544, \]
\[ K_{12} = 33221100, \]
\[ K_{13} = \text{f0f1f2f3}, \]
\[ K_{14} = \text{f4f5f6f7}, \]
\[ K_{15} = \text{f8f9fafb}, \]
\[ K_{16} = \text{fcfdfeff}, \]
\[ K_{17} = \text{ffeeeddcc}, \]
\[ K_{18} = \text{bbaa9988}, \]
\[ K_{19} = 77665544, \]
\[ K_{20} = 33221100, \]
\[ K_{21} = \text{f0f1f2f3}, \]
\[ K_{22} = \text{f4f5f6f7}, \]
\[ K_{23} = \text{f8f9fafb}, \]
\[ K_{24} = \text{fcfdfeff}, \]
\[ K_{25} = \text{fcfdfeff}, \]
\[ K_{26} = \text{f8f9fafb}, \]
\[ K_{27} = \text{f4f5f6f7}, \]
\[ K_{28} = \text{f0f1f2f3}, \]
\[ K_{29} = 33221100, \]
\[ K_{30} = 77665544, \]
\[ K_{31} = \text{bbaa9988}, \]
\[ K_{32} = \text{ffeeeddcc}. \]

**A.4. Test Encryption**

In this test example, encryption is performed on the round keys specified in clause A.3. Let the plaintext be

\[ a = \text{fedcba9876543210}, \]

then
(a_1, a_0) = (fedcba98, 76543210), 
G[K_1](a_1, a_0) = (76543210, 28da3b14), 
G[K_2]G[K_1](a_1, a_0) = (28da3b14, b14337a5),  
G[K_3]...G[K_1](a_1, a_0) = (b14337a5, 633a7c68), 
G[K_4]...G[K_1](a_1, a_0) = (633a7c68, ea89c02c), 
G[K_5]...G[K_1](a_1, a_0) = (ea89c02c, 11fe726d), 
G[K_6]...G[K_1](a_1, a_0) = (11fe726d, ad0310a4),  
G[K_7]...G[K_1](a_1, a_0) = (ad0310a4, 37d97f25), 
G[K_8]...G[K_1](a_1, a_0) = (37d97f25, 46324615),  
G[K_9]...G[K_1](a_1, a_0) = (46324615, ce995f2a), 
G[K_10]...G[K_1](a_1, a_0) = (ce995f2a, 93c1f449), 
G[K_11]...G[K_1](a_1, a_0) = (93c1f449, 4811c7ad), 
G[K_12]...G[K_1](a_1, a_0) = (4811c7ad, c4b3edca), 
G[K_13]...G[K_1](a_1, a_0) = (c4b3edca, 44ca5ce1), 
G[K_14]...G[K_1](a_1, a_0) = (44ca5ce1, fef51b68), 
G[K_15]...G[K_1](a_1, a_0) = (fef51b68, 2098cd86), 
G[K_16]...G[K_1](a_1, a_0) = (2098cd86, 4f15b0bb), 
G[K_17]...G[K_1](a_1, a_0) = (4f15b0bb, e32805bc), 
G[K_18]...G[K_1](a_1, a_0) = (e32805bc, e7116722), 
G[K_19]...G[K_1](a_1, a_0) = (e7116722, 89cadf21), 
G[K_20]...G[K_1](a_1, a_0) = (89cadf21, bac8444d), 
G[K_21]...G[K_1](a_1, a_0) = (bac8444d, 11263a21),  
G[K_22]...G[K_1](a_1, a_0) = (11263a21, 625434c3), 
G[K_23]...G[K_1](a_1, a_0) = (625434c3, 8025c0a5), 
G[K_24]...G[K_1](a_1, a_0) = (8025c0a5, b0d66514),  
G[K_25]...G[K_1](a_1, a_0) = (b0d66514, 47b1d5f4), 
G[K_26]...G[K_1](a_1, a_0) = (47b1d5f4, c78e6d50), 
G[K_27]...G[K_1](a_1, a_0) = (c78e6d50, 80251e99), 
G[K_28]...G[K_1](a_1, a_0) = (80251e99, 2b96eca6), 
G[K_29]...G[K_1](a_1, a_0) = (2b96eca6, a5ef4401),  
G[K_30]...G[K_1](a_1, a_0) = (a5ef4401, 239a4577), 
G[K_31]...G[K_1](a_1, a_0) = (239a4577, c2d8ca3d).

Then the ciphertext is 

\[ b = G^*[K_32]G[K_31]...G[K_1](a_1, a_0) = 4ee901e5c2d8ca3d. \]

### A.5. Test Decryption

In this test example, decryption is performed on the round keys specified in clause A.3. Let the ciphertext be 

\[ b = 4ee901e5c2d8ca3d, \]

then

\[ b = 4ee901e5c2d8ca3d. \]
(b_1, b_0) = (4ee901e5, c2d8ca3d),
G[K_32](b_1, b_0) = (c2d8ca3d, 239a4577),
G[K_31]G[K_32](b_1, b_0) = (239a4577, 05ef4401),
G[K_30]...G[K_32](b_1, b_0) = (05ef4401, 2b96eca6),
G[K_29]...G[K_32](b_1, b_0) = (2b96eca6, 80251e99),
G[K_28]...G[K_32](b_1, b_0) = (80251e99, c78e6d50),
G[K_27]...G[K_32](b_1, b_0) = (c78e6d50, 47b1d5f4),
G[K_26]...G[K_32](b_1, b_0) = (47b1d5f4, b0d66514),
G[K_25]...G[K_32](b_1, b_0) = (b0d66514, 8025c0a5),
G[K_24]...G[K_32](b_1, b_0) = (8025c0a5, 625434c3),
G[K_23]...G[K_32](b_1, b_0) = (625434c3, 11263a21),
G[K_22]...G[K_32](b_1, b_0) = (11263a21, bac8444d),
G[K_21]...G[K_32](b_1, b_0) = (bac8444d, 89cadf21),
G[K_20]...G[K_32](b_1, b_0) = (89cadf21, e7116722),
G[K_19]...G[K_32](b_1, b_0) = (e7116722, e32805bc),
G[K_18]...G[K_32](b_1, b_0) = (e32805bc, 4f15b0bb),
G[K_17]...G[K_32](b_1, b_0) = (4f15b0bb, 2098cd86),
G[K_16]...G[K_32](b_1, b_0) = (2098cd86, fef51b68),
G[K_15]...G[K_32](b_1, b_0) = (fef51b68, 44ca5ce1),
G[K_14]...G[K_32](b_1, b_0) = (44ca5ce1, c4b3edca),
G[K_13]...G[K_32](b_1, b_0) = (c4b3edca, 4811c7ad),
G[K_12]...G[K_32](b_1, b_0) = (4811c7ad, 93c1f449),
G[K_11]...G[K_32](b_1, b_0) = (93c1f449, ce995f2a),
G[K_10]...G[K_32](b_1, b_0) = (ce995f2a, 46324615),
G[K_9]...G[K_32](b_1, b_0) = (46324615, 37d97f25),
G[K_8]...G[K_32](b_1, b_0) = (37d97f25, ad0310a4),
G[K_7]...G[K_32](b_1, b_0) = (ad0310a4, 11fe726d),
G[K_6]...G[K_32](b_1, b_0) = (11fe726d, ea89c02c),
G[K_5]...G[K_32](b_1, b_0) = (ea89c02c, 633a7c68),
G[K_4]...G[K_32](b_1, b_0) = (633a7c68, b14337a5),
G[K_3]...G[K_32](b_1, b_0) = (b14337a5, 28da3b14),
G[K_2]...G[K_32](b_1, b_0) = (28da3b14, 76543210).

Then the plaintext is

a = G^[K_1]G[K_2]...G[K_32](b_1, b_0) = fedcba9876543210.

Appendix B.  Background

This specification is a translation of relevant parts of [GOSTR3412-2015] standard. The order of terms in both parts of Section 3 comes from original text. If one combines [RFC7801] with this document, he will have complete translation of [GOSTR3412-2015] into English.

Algoritmically Magma is a variation of block cipher defined in [RFC5830] ([GOST28147-89]) with the following clarifications and minor modifications:
1. S-BOX set is fixed at id-tc26-gost-28147-param-Z (See Appendix C of [RFC7836]);

2. key is parsed as a single big-endian integer (compared to little-endian approach used in [GOST28147-89]), which results in different subkey values being used;

3. data bytes are also parsed as single big-endian integer (instead of being parsed as little-endian integer).

Authors' Addresses

Vasily Dolmatov (editor)
JSC "NPK Kryptonite"
Spartakovskaya sq., 14, bld 2, JSC "NPK Kryptonite"
Moscow  105082
Russian Federation

Email: vdolmatov@gmail.com

Dmitry Eremin-Solenikov
Auriga, Inc
Torfyanaya Doroga, 7F, office 1410
Saint-Petersburg  197374
Russian Federation

Email: dbaryshkov@gmail.com