Using the GOST 28147-89, GOST R 34.11-94,
GOST R 34.10-94, and GOST R 34.10-2001 Algorithms with
Cryptographic Message Syntax (CMS)

Status of This Memo

This document specifies an Internet standards track protocol for the
Internet community, and requests discussion and suggestions for
improvements. Please refer to the current edition of the "Internet
Official Protocol Standards" (STD 1) for the standardization state
and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

This document describes the conventions for using the cryptographic
algorithms GOST 28147-89, GOST R 34.10-94, GOST R 34.10-2001, and
GOST R 34.11-94 with the Cryptographic Message Syntax (CMS). The CMS
is used for digital signature, digest, authentication, and encryption
of arbitrary message contents.
# Table of Contents

1. Introduction .................................................... 3  
   1.1. Terminology ................................................ 3  
2. Message Digest Algorithms ....................................... 3  
   2.1. Message Digest Algorithm GOST R 34.11-94 ................ 3  
3. Signature Algorithms ........................................... 4  
   3.1. Signature Algorithm GOST R 34.10-94 ...................... 4  
   3.2. Signature Algorithm GOST R 34.10-2001 .................... 5  
4. Key Management Algorithms ..................................... 5  
   4.1. Key Agreement Algorithms .................................. 6  
      4.1.1. Key Agreement Algorithms Based on GOST R  
            34.10-94/2001 Public ................................ 6  
   4.2. Key Transport Algorithms .................................. 8  
      4.2.1. Key Transport Algorithm Based on GOST R  
            34.10-94/2001 Public ................................ 8  
5. Content Encryption Algorithms ................................ 9  
   5.1. Content Encryption Algorithm GOST 28147-89 ............ 10  
6. MAC Algorithms ................................................ 10  
   6.1. HMAC with GOST R 34.11-94 ................................ 10  
7. Use with S/MIME ............................................... 11  
   7.1. Parameter micalg ........................................... 11  
   7.2. Attribute SMIMECapabilities ............................... 11  
8. Security Considerations ....................................... 12  
9. Examples .................................................... 12  
   9.1. Signed Message ............................................ 12  
   9.2. Enveloped Message Using Key Agreement .................... 14  
   9.3. Enveloped Message Using Key Transport .................... 17  
10. ASN.1 Modules ............................................... 19  
   10.1. GostR3410-EncryptionSyntax ............................ 19  
   10.2. GostR3410-94-SignatureSyntax .......................... 21  
   10.3. GostR3410-2001-SignatureSyntax ........................ 22  
11. Acknowledgements ........................................... 23  
12. References .................................................. 24  
   12.1. Normative References .................................... 24  
   12.2. Informative References ................................. 25
1. Introduction

The Cryptographic Message Syntax [CMS] is used for digital signature, digest, authentication, and encryption of arbitrary message contents. This companion specification describes the use of cryptographic algorithms GOST 28147-89 [GOST28147], GOST R 34.10-94 [GOST3431095, GOSTR341094], GOST R 34.10-2001 [GOST3431004, GOSTR341001], and GOST R 34.11-94 [GOST3431195, GOSTR341194] in CMS, as proposed by the CRYPTO-PRO Company for the "Russian Cryptographic Software Compatibility Agreement" community. This document does not describe these cryptographic algorithms; they are defined in corresponding national standards.

The CMS values are generated using ASN.1 [X.208-88], using BER encoding [X.209-88]. This document specifies the algorithm identifiers for each algorithm, including ASN.1 for object identifiers and any associated parameters.

The fields in the CMS employed by each algorithm are identified.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Message Digest Algorithms

This section specifies the conventions for using the digest algorithm GOST R 34.11-94 employed by CMS.

Digest values are located in the DigestedData digest field and the Message Digest authenticated attribute. In addition, digest values are input to signature algorithms.

2.1. Message Digest Algorithm GOST R 34.11-94

The hash function GOST R 34.11-94 has been developed by "GUBS of Federal Agency Government Communication and Information" and "All-Russian Scientific and Research Institute of Standardization". The algorithm GOST R 34.11-94 produces a 256-bit hash value of the arbitrary finite bit-length input. This document does not contain the full GOST R 34.11-94 specification, which can be found in [GOSTR341194] in Russian. [Schneier95], ch. 18.11, p. 454, contains a brief technical description in English.
The hash algorithm GOST R 34.11-94 has the following identifier:

\[
\text{id-GostR3411-94 OBJECT IDENTIFIER ::= } \\
\{ \text{ iso(1) member-body(2) ru(643) rans(2) cryptopro(2) } \\
\text{ gostr3411(9) } \}
\]

The AlgorithmIdentifier parameters field MUST be present, and the parameters field MUST contain NULL. Implementations MAY accept the GOST R 34.11-94 AlgorithmIdentifiers with absent parameters as well as NULL parameters.

This function is always used with default parameters id-GostR3411-94-CryptoProParamSet (see Section 8.2 of [CPALGS]).

When the Message Digest authenticated attribute is present, the DigestedData digest contains a 32-byte digest in little-endian representation:

\[
\text{GostR3411-94-Digest ::= OCTET STRING (SIZE (32))}
\]

3. Signature Algorithms

This section specifies the CMS procedures for the GOST R 34.10-94 and GOST R 34.10-2001 signature algorithms.

Signature algorithm identifiers are located in the SignerInfo signatureAlgorithm field of SignedData. Also, signature algorithm identifiers are located in the SignerInfo signatureAlgorithm field of countersignature attributes.

Signature values are located in the SignerInfo signature field of SignedData. Also, signature values are located in the SignerInfo signature field of countersignature attributes.

3.1. Signature Algorithm GOST R 34.10-94

GOST R 34.10-94 has been developed by "GUBS of Federal Agency Government Communication and Information" and "All-Russian Scientific and Research Institute of Standardization". This signature algorithm MUST be used conjointly with the GOST R 34.11-94 message digest algorithm. This document does not contain the full GOST R 34.10-94 specification, which is fully described in [GOSTR341094] in Russian; and a brief description in English can be found in [Schneier95], ch. 20.3, p. 495.

The GOST R 34.10-94 signature algorithm has the following public key algorithm identifier:
id-GostR3410-94-signature OBJECT IDENTIFIER ::= id-GostR3410-94

id-GostR3410-94 is defined in Section 2.3.1 of [CPPK].

The signature algorithm GOST R 34.10-94 generates a digital signature in the form of two 256-bit numbers, \( r' \) and \( s \). Its octet string representation consists of 64 octets, where the first 32 octets contain the big-endian representation of \( s \) and the second 32 octets contain the big-endian representation of \( r' \).

GostR3410-94-Signature ::= OCTET STRING (SIZE (64))

3.2. Signature Algorithm GOST R 34.10-2001

GOST R 34.10-2001 has been developed by "GUBS of Federal Agency Government Communication and Information" and "All-Russian Scientific and Research Institute of Standardization". This signature algorithm MUST be used conjointly with GOST R 34.11-94. This document does not contain the full GOST R 34.10-2001 specification, which is fully described in [GOSTR341001].

The signature algorithm GOST R 34.10-2001 has the following public key algorithm identifier:

id-GostR3410-2001-signature OBJECT IDENTIFIER ::= id-GostR3410-2001

id-GostR3410-2001 is defined in Section 2.3.2 of [CPPK].

The signature algorithm GOST R 34.10-2001 generates a digital signature in the form of two 256-bit numbers, \( r \) and \( s \). Its octet string representation consists of 64 octets, where the first 32 octets contain the big-endian representation of \( s \) and the second 32 octets contain the big-endian representation of \( r \).

GostR3410-2001-Signature ::= OCTET STRING (SIZE (64))

4. Key Management Algorithms

This chapter describes the key agreement and key transport algorithms, based on the VKO GOST R 34.10-94 and VKO GOST R 34.10-2001 key derivation algorithms, and the CryptoPro and GOST 28147-89 key wrap algorithms, described in [CPALGS]. They MUST be used only with the content encryption algorithm GOST 28147-89, defined in Section 5 of this document.
4.1. Key Agreement Algorithms

This section specifies the conventions employed by CMS implementations that support key agreement using both the VKO GOST R 34.10-94 and VKO GOST R 34.10-2001 algorithms, described in [CPALGS].

Key agreement algorithm identifiers are located in the EnvelopedData RecipientInfos KeyAgreeRecipientInfo keyEncryptionAlgorithm and AuthenticatedData RecipientInfos KeyAgreeRecipientInfo keyEncryptionAlgorithm fields.

Wrapped content-encryption keys are located in the EnvelopedData RecipientInfos KeyAgreeRecipientInfo RecipientEncryptedKeys encryptedKey field. Wrapped message-authentication keys are located in the AuthenticatedData RecipientInfos KeyAgreeRecipientInfo RecipientEncryptedKeys encryptedKey field.

4.1.1. Key Agreement Algorithms Based on GOST R 34.10-94/2001 Public Keys

The EnvelopedData RecipientInfos KeyAgreeRecipientInfo field is used as follows:

The version MUST be 3.

The originator MUST be the originatorKey alternative. The originatorKey algorithm field MUST contain the object identifier id-GostR3410-94 or id-GostR3410-2001 and corresponding parameters (defined in Sections 2.3.1, 2.3.2 of [CPPK]).

The originatorKey publicKey field MUST contain the sender’s public key.

keyEncryptionAlgorithm MUST be the id-GostR3410-94-CryptoPro-ESDH or the id-GostR3410-2001-CryptoPro-ESDH algorithm identifier, depending on the recipient public key algorithm. The algorithm identifier parameter field for these algorithms is KeyWrapAlgorithm, and this parameter MUST be present. The KeyWrapAlgorithm denotes the algorithm and parameters used to encrypt the content-encryption key with the pairwise key-encryption key generated using the VKO GOST R 34.10-94 or the VKO GOST R 34.10-2001 key agreement algorithms.

The algorithm identifiers and parameter syntax is:

\[
\text{id-GostR3410-94-CryptoPro-ESDH OBJECT IDENTIFIER ::=}
\{ iso(1) member-body(2) ru(643) rans(2) cryptopro(2) gostR3410-94-CryptoPro-ESDH(97) \}
\]
id-GostR3410-2001-CryptoPro-ESDH OBJECT IDENTIFIER ::= 
    { iso(1) member-body(2) ru(643) rans(2) cryptopro(2)
      gostR3410-2001-CryptoPro-ESDH(96) }

KeyWrapAlgorithm ::= AlgorithmIdentifier

When keyEncryptionAlgorithm is id-GostR3410-94-CryptoPro-ESDH,
KeyWrapAlgorithm algorithm MUST be the id-Gost28147-89-CryptoPro-
KeyWrap algorithm identifier.

id-Gost28147-89-CryptoPro-KeyWrap OBJECT IDENTIFIER ::= 
    { iso(1) member-body(2) ru(643) rans(2) cryptopro(2)
      keyWrap(13) cryptoPro(1) }

The CryptoPro Key Wrap algorithm is described in Sections 6.3 and
6.4 of [CPALGS].

When keyEncryptionAlgorithm is id-GostR3410-2001-CryptoPro-ESDH,
KeyWrapAlgorithm algorithm MUST be either the id-Gost28147-89-
CryptoPro-KeyWrap or id-Gost28147-89-None-KeyWrap algorithm
identifier.

id-Gost28147-89-None-KeyWrap OBJECT IDENTIFIER ::= 
    { iso(1) member-body(2) ru(643) rans(2) cryptopro(2)
      keyWrap(13) none(0) }

The GOST 28147-89 Key Wrap algorithm is described in Sections 6.1
and 6.2 of [CPALGS].

KeyWrapAlgorithm algorithm parameters MUST be present. The syntax
for KeyWrapAlgorithm algorithm parameters is

Gost28147-89-KeyWrapParameters ::= 
    SEQUENCE { 
      encryptionParamSet Gost28147-89-ParamSet, 
      ukm OCTET STRING (SIZE (8)) OPTIONAL 
    } 

Gost28147-89-ParamSet ::= OBJECT IDENTIFIER

Gost28147-89-KeyWrapParameters ukm MUST be absent.

KeyAgreeRecipientInfo ukm MUST be present and contain eight
octets.

encryptedKey MUST encapsulate Gost28147-89-EncryptedKey, where
maskKey MUST be absent.
Using the secret key corresponding to the originatorKey publicKey and the recipient’s public key, the algorithm VKO GOST R 34.10-94 or VKO GOST R 34.10-2001 (described in [CPALGS]) is applied to produce the KEK.

Then the key wrap algorithm, specified by KeyWrapAlgorithm, is applied to produce CEK_ENC, CEK_MAC, and UKM. Gost28147-89-KeyWrapParameters encryptionParamSet is used for all encryption operations.

The resulting encrypted key (CEK_ENC) is placed in the Gost28147-89-EncryptedKey encryptedKey field, its mac (CEK_MAC) is placed in the Gost28147-89-EncryptedKey macKey field, and UKM is placed in the KeyAgreeRecipientInfo ukm field.

4.2. Key Transport Algorithms

This section specifies the conventions employed by CMS implementations that support key transport using both the VKO GOST R 34.10-94 and VKO GOST R 34.10-2001 algorithms, described in [CPALGS].

Key transport algorithm identifiers are located in the EnvelopedData RecipientInfos KeyTransRecipientInfo keyEncryptionAlgorithm field.

Key transport encrypted content-encryption keys are located in the EnvelopedData RecipientInfos KeyTransRecipientInfo encryptedKey field.

4.2.1. Key Transport Algorithm Based on GOST R 34.10-94/2001 Public Keys

The EnvelopedData RecipientInfos KeyTransRecipientInfo field is used as follows:

The version MUST be 0 or 3.

keyEncryptionAlgorithm and parameters MUST be identical to the recipient public key algorithm and parameters.
encryptedKey encapsulates GostR3410-KeyTransport, which consists of encrypted content-encryption key, its MAC, GOST 28147-89 algorithm parameters used for key encryption, the sender’s ephemeral public key, and UKM (UserKeyingMaterial; see [CMS], Section 10.2.6).

transportParameters MUST be present.

ephemeralPublicKey MUST be present and its parameters, if present, MUST be equal to the recipient public key parameters;

GostR3410-KeyTransport ::= SEQUENCE {
  sessionEncryptedKey   Gost28147-89-EncryptedKey,
  transportParameters   [0] IMPLICIT GostR3410-TransportParameters OPTIONAL
}

GostR3410-TransportParameters ::= SEQUENCE {
  encryptionParamSet   OBJECT IDENTIFIER,
  ephemeralPublicKey   [0] IMPLICIT SubjectPublicKeyInfo OPTIONAL,
  ukm                  OCTET STRING
}

Using the secret key corresponding to the GostR3410-TransportParameters ephemeralPublicKey and the recipient’s public key, the algorithm VKO GOST R 34.10-94 or VKO GOST R 34.10-2001 (described in [CPALGS]) is applied to produce the KEK.

Then the CryptoPro key wrap algorithm is applied to produce CEK_ENC, CEK_MAC, and UKM. GostR3410-TransportParameters encryptionParamSet is used for all encryption operations.

The resulting encrypted key (CEK_ENC) is placed in the Gost28147-89-EncryptedKey encryptedKey field, its mac (CEK_MAC) is placed in the Gost28147-89-EncryptedKey macKey field, and UKM is placed in the GostR3410-TransportParameters ukm field.

5. Content Encryption Algorithms

This section specifies the conventions employed by CMS implementations that support content encryption using GOST 28147-89.

Content encryption algorithm identifiers are located in the EnvelopedData EncryptedContentInfo contentEncryptionAlgorithm and the EncryptedData EncryptedContentInfo contentEncryptionAlgorithm fields.
Content encryption algorithms are used to encipher the content located in the EnvelopedData EncryptedContentInfo encryptedContent field and the EncryptedData EncryptedContentInfo encryptedContent field.

5.1. Content Encryption Algorithm GOST 28147-89

This section specifies the use of GOST 28147-89 algorithm for data encipherment.

GOST 28147-89 is fully described in [GOST28147] (in Russian).

This document specifies the following object identifier (OID) for this algorithm:

```
id-Gost28147-89 OBJECT IDENTIFIER ::= 
  { iso(1) member-body(2) ru(643) rans(2) cryptopro(2) 
    gost28147-89(21) }
```

Algorithm parameters MUST be present and have the following structure:

```
Gost28147-89-Parameters ::= 
  SEQUENCE {
    iv                    Gost28147-89-IV,
    encryptionParamSet   OBJECT IDENTIFIER
  }
```

```
Gost28147-89-IV ::= OCTET STRING (SIZE (8))
```

encryptionParamSet specifies the set of corresponding Gost28147-89-ParamSetParameters (see Section 8.1 of [CPALGS])

6. MAC Algorithms

This section specifies the conventions employed by CMS implementations that support the message authentication code (MAC) based on GOST R 34.11-94.

MAC algorithm identifiers are located in the AuthenticatedData macAlgorithm field.

MAC values are located in the AuthenticatedData mac field.

6.1. HMAC with GOST R 34.11-94

HMAC_GOSTR3411 (K,text) function is based on hash function GOST R 34.11-94, as defined in Section 3 of [CPALGS].
This document specifies the following OID for this algorithm:

```
id-HMACGostR3411-94 OBJECT IDENTIFIER ::= 
    { iso(1) member-body(2) ru(643) rans(2) cryptopro(2)
        hmacgostr3411(10) }
```

This algorithm has the same parameters as the GOST R 34.11-94 digest algorithm and uses the same OIDs for their identification (see [CPPK]).

7. Use with S/MIME

This section defines the use of the algorithms defined in this document with S/MIME [RFC3851].

7.1. Parameter micalg

When using the algorithms defined in this document, micalg parameter SHOULD be set to "gostr3411-94"; otherwise, it MUST be set to "unknown".

7.2. Attribute SMIMECapabilities

The SMIMECapability value that indicates support for the GOST R 34.11-94 digest algorithm is the SEQUENCE with the capabilityID field containing the object identifier id-GostR3411-94 and no parameters. The DER encoding is:

```
30 08 06 06 2A 85 03 02 02 09
```

The SMIMECapability value that indicates support for the GOST 28147-89 encryption algorithm is the SEQUENCE with the capabilityID field containing the object identifier id-Gost28147-89 and no parameters. The DER encoding is:

```
30 08 06 06 2A 85 03 02 02 15
```

If the sender wishes to indicate support for a specific parameter set, SMIMECapability parameters MUST contain the Gost28147-89-Parameters structure. Recipients MUST ignore the Gost28147-89-Parameters iv field and assume that the sender supports the parameters specified in the Gost28147-89-Parameters encryptionParamSet field.

The DER encoding for the SMIMECapability, indicating support for GOST 28147-89 with id-Gost28147-89-CryptoPro-A-ParamSet (see [CPALGS]), is:
8. Security Considerations

Conforming applications MUST use unique values for ukm and iv. Recipients MAY verify that ukm and iv, specified by the sender, are unique.

It is RECOMMENDED that software applications verify that signature values, subject public keys, and algorithm parameters conform to [GOSTR341001] and [GOSTR341094] standards prior to their use.

Cryptographic algorithm parameters affect algorithm strength. The use of parameters not listed in [CPALGS] is NOT RECOMMENDED (see the Security Considerations section of [CPALGS]).

Use of the same key for signature and key derivation is NOT RECOMMENDED. When signed CMS documents are used as an analogue to a manual signing, in the context of Russian Federal Electronic Digital Signature Law [RFEDSL], signer certificate MUST contain the keyUsage extension, it MUST be critical, and keyUsage MUST NOT include keyEncipherment or keyAgreement (see [PROFILE], Section 4.2.1.3). Application SHOULD be submitted for examination by an authorized agency in appropriate levels of target_of_evaluation (TOE), according to [RFEDSL], [RFLLIC], and [CRYPTOLIC].

9. Examples

Examples here are stored in the same format as the examples in [RFC4134] and can be extracted using the same program.

If you want to extract without the program, copy all the lines between the "|>" and "<|" markers, remove any page breaks, and remove the "|" in the first column of each line. The result is a valid Base64 blob that can be processed by any Base64 decoder.

9.1. Signed Message

This message is signed using the sample certificate from Section 4.2 of [CPPK]. The public key (x,y) from the same section can be used to verify the message signature.

```plaintext
0 296: SEQUENCE {
4    9:  OBJECT IDENTIFIER signedData
15 281:  [0] {
19 277:   SEQUENCE {
23    1:    INTEGER 1
```
26 12:    SET {
28 10:     SEQUENCE {
30 6:      OBJECT IDENTIFIER id-GostR3411-94
38 0:      NULL
:   }
:   }
40 27:    SEQUENCE {
42 9:     OBJECT IDENTIFIER data
53 14:     [0] {
55 12:      OCTET STRING 73 61 6D 70 6C 65 20 74 65 78 74 0A
:   }
:   }
69 228:    SET {
72 225:     SEQUENCE {
75 1:      INTEGER 1
78 129:     SEQUENCE {
81 109:      SEQUENCE {
83 31:       SET {
85 29:        SEQUENCE {
87 3:         OBJECT IDENTIFIER commonName
92 22:          UTF8String 'GostR3410-2001 example'
:     }
:     }
116 18:     SET {
118 16:      SEQUENCE {
120 3:       OBJECT IDENTIFIER organizationName
125 9:        UTF8String 'CryptoPro'
:      }
:      }
136 11:     SET {
138 9:      SEQUENCE {
140 3:       OBJECT IDENTIFIER countryName
145 2:        PrintableString 'RU'
:     }
:     }
149 41:     SET {
151 39:      SEQUENCE {
153 9:       OBJECT IDENTIFIER emailAddress
164 26:        IA5String 'GostR3410-2001@example.com'
:     }
:     }
192 16:      INTEGER
:        2B F5 C6 1E C2 11 BD 17 C7 DC D4 62 66 B4 2E 21
:   }
210 10:     SEQUENCE {
212 6:      OBJECT IDENTIFIER id-GostR3411-94
220 0:      NULL
Using GOST with CMS

|>GostR3410-2001-signed.bin

|MIIBKAYJKoZIhvcNAQcCoIBGTMCCARUCAQEExDDAKBgYqhQMCAgkFADAAbBgkgkhiG
9wOBwGgQMc2FtcGx1IHRleHQKMYHMKIHAgEBMIGBMG0xHzAdBgvNVBMFMFkdv
c3RMzQxMC0yMDAxIGV4YW1wbGUxJCAQBoGvNBAMeMCUNyeXB0b1BybzELMAkGA1UE
BwMUUlBwKGA1UwIwEwUwIBAgIwMwYwYwYwIwMwYwMwYwMwYwMwYwMwYwMwYwMwYwMwYw
AhAr9cYewhG9F8fc1GJmtC4hMAoGBigFAwICCUAMQGigFAwICEwIAREDAw0Lz
|<GostR3410-2001-signed.bin

9.2. Enveloped Message Using Key Agreement

This message is encrypted using the sample certificate from Section 4.2 of [CPPK] as a recipient certificate. The private key ‘d’ from the same section can be used to decrypt this message.

0 420: SEQUENCE {
4  9: OBJECT IDENTIFIER envelopedData
15 405: [0] {
19 401: SEQUENCE {
23  1: INTEGER 2
26 336: SET {
30 332: [1] {
34  1: INTEGER 3
37 101: [0] {
39  99: [1] {
41 28: SEQUENCE {
43  6: OBJECT IDENTIFIER id-GostR3410-2001
51 18: SEQUENCE {
53  7: OBJECT IDENTIFIER
      : id-GostR3410-2001-CryptoPro-XchA-ParamSet
62  7: OBJECT IDENTIFIER

Leontiev & Chudov Standards Track
id-GostR3411-94-CryptoProParamSet

BIT STRING, encapsulates {

OCTET STRING

B3 55 39 F4 67 81 97 2B A5 C4 D9 84 1F 27 FB 81
ED 08 32 E6 9A D4 F2 00 78 B8 FF 83 64 EA D2 1D
B0 78 3C 7D FE 03 C1 F4 06 E4 3B CC 16 B9 C5 F6
F6 19 37 1C 17 B8 A0 AA C7 D1 A1 94 B3 A5 36 20

}

]

{}{1}

OCTET STRING 2F F0 F6 D1 86 4B 32 8A

SEQUENCE {

OBJECT IDENTIFIER id-GostR3410-2001-CryptoPro-ESDH

OBJECT IDENTIFIER id-Gost28147-89-None-KeyWrap

OBJECT IDENTIFIER id-Gost28147-89-CryptoPro-A-ParamSet

}

}

SEQUENCE {

SEQUENCE {

SEQUENCE {

OBJECT IDENTIFIER commonName

UTF8String ‘GostR3410-2001 example’

}

}

SET {

OBJECT IDENTIFIER organizationName

UTF8String ‘CryptoPro’

}

}

SET {

OBJECT IDENTIFIER countryName

PrintableString ‘RU’

}

}

SET {

OBJECT IDENTIFIER countryName

PrintableString ‘RU’

}
263 39:     SEQUENCE {
265  9:     OBJECT IDENTIFIER emailAddress
276 26:     IA5String 'GostR3410-2001@example.com'
:     }
:     }

304 16:     INTEGER
:     2B F5 C6 1E C2 11 BD 17 C7 DC D4 62 66 B4 2E 21
:     }

322 42:     OCTET STRING, encapsulates {
324 40:     SEQUENCE {
326  32:     OCTET STRING
:     16 A3 1C E7 CE 4E E9 0D F1 EC 74 69 04 68 1E C7
:     9F 3A ED B8 3B 1F 1D 4A 7E F9 A5 D9 CB 19 D5 E8
360  4:     OCTET STRING
:     93 FD 86 7E
:     }
:     }

366 56:     SEQUENCE {
368  9:     OBJECT IDENTIFIER data
379 29:     SEQUENCE {
381  6:     OBJECT IDENTIFIER id-Gost28147-89
389 19:     SEQUENCE {
391  8:     OCTET STRING
:     B7 35 E1 7A 07 35 A2 1D
399  7:     OBJECT IDENTIFIER id-Gost28147-89-CryptoPro-A-ParamSet
:     }
:     }

410 12:     [0] 39 B1 8A F4 BF A9 E2 65 25 B6 55 C9
:     }
:     }

> GostR3410-2001-keyagree.bin
  MIIBpAYJKoZIhvcNAQcDoIB1TCCA2ZECASIgTgQoYIBTAIBA6B1oWMwHAYGKoUD
  AgITMIB1GBygFAwICJAAGBygFAwICHqEDQwAEQLNVOfRng2zrPcTzhB8n+4HtcDLm
  ntTyAH14/4Nk6tIdsHg8ff4DwfQQ5DvMFrfnF9vYXwXuKCqx9GhiL01NiChCgQI
  L/D20YlMoowHgYKoUDAgJgMBQGBygFAwICDQAwCQYHkoUDAgIfATCBszCBsDCB
  qTBlMR8wHQYDVQDBZBHb3NUjM0MTATmJawMSB1eGFtcGxlMR1wEAYDVQKDAlD
  cnlwdG9Ccmx8CzAJBgNVBAYTA1JVMswJwYJKoZIhvcNvBkBpHBh3N0UjM0MTAT
  MjAwMUB1eGFtcGxlMvbQIQ/XGHSIRvRfH3NRiZrQuIQMgEIbajHOfOTukN
  8ex0aQRoHsefuOu240x8dSn75pddGdXoBAST/YZ+MDgCSqGSib3DQEHATAdBgYq
  hQMCAhUwEQtzXhecg1oh0GByqFAwICHwGAD0mxivS/qeJ1JbEZyQ==
  <GostR3410-2001-keyagree.bin
9.3. Envelope Message Using Key Transport

This message is encrypted using the sample certificate from Section 4.2 of [CPPK] as a recipient certificate. The private key ‘d’ from the same section can be used to decrypt this message.

0  423: SEQUENCE {
4    9:  OBJECT IDENTIFIER envelopedData
15   408:  [0] {
19  404:  SEQUENCE {
23    1:    INTEGER 0
26   339:  SET {
30  335:  SEQUENCE {
34    1:    INTEGER 0
37   129:  SEQUENCE {
40  109:  SEQUENCE {
42   31:  SET {
44   29:  SEQUENCE {
46    3:  OBJECT IDENTIFIER commonName
51   22:  UTF8String ‘GostR3410-2001 example’

75   18:  SET {
77   16:  SEQUENCE {
79    3:  OBJECT IDENTIFIER organizationName
84    9:  UTF8String ‘CryptoPro’

95   11:  SET {
97    9:  SEQUENCE {
99    3:  OBJECT IDENTIFIER countryName
104    2:  PrintableString ‘RU’

108   41:  SET {
110   39:  SEQUENCE {
112    9:  OBJECT IDENTIFIER emailAddress
123   26:  IA5String ‘GostR3410-2001@example.com’

151   16:  INTEGER
155:    2B F5 C6 1E C2 11 BD 17 C7 DC D4 62 66 B4 2E 21

169   28:  SEQUENCE {
171    6:  OBJECT IDENTIFIER id-GostR3410-2001
179   18:  SEQUENCE {
181   7:  OBJECT IDENTIFIER

Leontiev & Chudov Standards Track [Page 17]
id-GostR3410-2001-CryptoPro-XchA-ParamSet

OBJECT IDENTIFIER
id-GostR3411-94-CryptoProParamSet

OCTET STRING, encapsulates {

SEQUENCE {

OCTET STRING
6A 2F A8 21 06 95 68 9F E4 47 AA 9E CB 61 15
2B 7E 41 60 BC 5D 8D FB F5 3D 28 1B 18 9A F9 75

OCTET STRING
36 6D 98 B7

}

[0] {

OBJECT IDENTIFIER
id-Gost28147-89-CryptoPro-A-ParamSet

[0] {

SEQUENCE {

OBJECT IDENTIFIER id-GostR3410-2001

SEQUENCE {

OBJECT IDENTIFIER
id-GostR3410-2001-CryptoPro-XchA-ParamSet

OBJECT IDENTIFIER
id-GostR3411-94-CryptoProParamSet

}

BIT STRING encapsulates {

OCTET STRING
4D 2B 2F 33 90 E6 DC A3 DD 55 2A CD DF E0 EF FB
31 F7 73 7E 4E FF BF 78 89 8A 2B C3 CD 31 94 04
4B OE 60 48 96 1F DB C7 5D 12 6F DA B2 40 8A 77
B5 BD EA F2 EC 34 CB 23 9F 9B 8B DD 9E 12 C0 F6
}

}

OCTET STRING
97 95 E3 2C 2B AD 2B 0C
}

}

}

}

}

}

SEQUENCE {

OBJECT IDENTIFIER data

SEQUENCE {

OBJECT IDENTIFIER id-Gost28147-89

SEQUENCE {

OCTET STRING BC 10 8B 1F 0B FF 34 29

Leontiev & Chudov Standards Track [Page 18]
10.  ASN.1 Modules

Additional ASN.1 modules, referenced here, can be found in [CPALGS].

10.1.  GostR3410-EncryptionSyntax

GostR3410-EncryptionSyntax

{ iso(1) member-body(2) ru(643) rans(2) cryptopro(2)
  other(1) modules(1) gostR3410-EncryptionSyntax(5) 2 }

DEFINITIONS ::= BEGIN
  -- EXPORTS All --
  -- The types and values defined in this module are exported for 
  -- use in the other ASN.1 modules contained within the Russian 
  -- Cryptography "GOST" & "GOST R" Specifications, and for the use 
  -- of other applications which will use them to access Russian 
  -- Cryptography services. Other applications may use them for 
  -- their own purposes, but this will not constrain extensions and 
  -- modifications needed to maintain or improve the Russian 
  -- Cryptography service.
  IMPORTS
    id-CryptoPro-algorithms,
    gost28147-89-EncryptionSyntax,
    gostR3410-1994-EncryptionSyntax,
    gostR3410-2001-EncryptionSyntax,
    ALGORITHM-IDENTIFIER,
    cryptographic-Gost-Useful-Definitions
FROM Cryptographic-Gost-Useful-Definitions -- in [CPALGS]
  { iso(1) member-body(2) ru(643) rans(2)
cryptopro(2) other(1) modules(1)
cryptographic-Gost-Useful-Definitions(0) 1 } id-GostR3410-94
FROM GostR3410-94-PKISyntax -- in [CPALGS]
gostR3410-94-PKISyntax
id-GostR3410-2001
FROM GostR3410-2001-PKISyntax -- in [CPALGS]
gostR3410-2001-PKISyntax
Gost28147-89-ParamSet,
Gost28147-89-EncryptedKey
FROM Gost28147-89-EncryptionSyntax -- in [CPALGS]
gost28147-89-EncryptionSyntax
SubjectPublicKeyInfo
FROM PKIX1Explicit88 (iso(1) identified-organization(3)
dod(6) internet(1) security(5) mechanisms(5) pkix(7)
id-mod(0) id-pkix1-explicit-88(1))

; -- CMS/PKCS#7 key agreement algorithms & parameters
Gost28147-89-KeyWrapParameters ::= SEQUENCE {
  encryptionParamSet Gost28147-89-ParamSet,
  ukm OCTET STRING (SIZE (8)) OPTIONAL
}

id-Gost28147-89-CryptoPro-KeyWrap OBJECT IDENTIFIER ::= {
  id-CryptoPro-algorithms keyWrap(13) cryptoPro(1) }
id-Gost28147-89-None-KeyWrap OBJECT IDENTIFIER ::= {
  id-CryptoPro-algorithms keyWrap(13) none(0) }
Gost28147-89-KeyWrapAlgorithms ALGORITHM-IDENTIFIER ::= {
  Gost28147-89-KeyWrapParameters IDENTIFIED BY
  id-Gost28147-89-CryptoPro-KeyWrap } |
  Gost28147-89-KeyWrapParameters IDENTIFIED BY
  id-Gost28147-89-None-KeyWrap
}

id-GostR3410-2001-CryptoPro-ESDH OBJECT IDENTIFIER ::= {
  id-CryptoPro-algorithms
  gostR3410-2001-CryptoPro-ESDH(96) }
id-GostR3410-94-CryptoPro-ESDH OBJECT IDENTIFIER ::= {
  id-CryptoPro-algorithms
  gostR3410-94-CryptoPro-ESDH(97) }

-- CMS/PKCS#7 key transport algorithms & parameters
-- OID for CMS/PKCS#7 Key transport is id-GostR3410-94 from
-- GostR3410-94-PKISyntax or id-GostR3410-2001 from
-- GostR3410-2001-PKISyntax
-- Algorithms for CMS/PKCS#7 Key transport are
-- GostR3410-94-PublicKeyAlgorithms from
-- GostR3410-94-PKISyntax or
GostR3410-2001-PublicKeyAlgorithms from
GostR3410-2001-PKISyntax
-- SMIMECapability for CMS/PKCS#7 Key transport are
--   id-GostR3410-94 from GostR3410-94-PKISyntax or
--   id-GostR3410-2001 from GostR3410-2001-PKISyntax
id-GostR3410-94-KeyTransportSMIMECapability
OBJECT IDENTIFIER ::= id-GostR3410-94
id-GostR3410-2001-KeyTransportSMIMECapability
OBJECT IDENTIFIER ::= id-GostR3410-2001

GostR3410-KeyTransport ::=  
   SEQUENCE { 
     sessionEncryptedKey Gost28147-89-EncryptedKey, 
     transportParameters [0] 
     IMPLICIT GostR3410-TransportParameters OPTIONAL 
   } 

GostR3410-TransportParameters ::=  
   SEQUENCE { 
     encryptionParamSet Gost28147-89-ParamSet, 
     ephemeralPublicKey [0] 
     IMPLICIT SubjectPublicKeyInfo OPTIONAL, 
     ukm OCTET STRING ( SIZE(8) ) 
   } 

END -- GostR3410-EncryptionSyntax

10.2. GostR3410-94-SignatureSyntax

GostR3410-94-SignatureSyntax
   { iso(1) member-body(2) ru(643) rans(2) cryptopro(2) 
     other(1) modules(1) gostR3410-94-SignatureSyntax(3) 1 } 
DEFINITIONS ::= BEGIN 
-- EXPORTS All --
-- The types and values defined in this module are exported for 
-- use in the other ASN.1 modules contained within the Russian 
-- Cryptography "GOST" & "GOST R" Specifications, and for the use 
-- of other applications which will use them to access Russian 
-- Cryptography services. Other applications may use them for 
-- their own purposes, but this will not constrain extensions and 
-- modifications needed to maintain or improve the Russian 
-- Cryptography service.
IMPORTS 
   gostR3410-94-PKISyntax, ALGORITHM-IDENTIFIER, 
   cryptographic-Gost-Useful-Definitions 
FROM Cryptographic-Gost-Useful-Definitions -- in [CPALGS] 
   { iso(1) member-body(2) ru(643) rans(2) 
     cryptopro(2) other(1) modules(1) 
     cryptographic-Gost-Useful-Definitions(0) 1 } 
   id-GostR3410-94,
GostR3410-94-PublicKeyParameters
FROM GostR3410-94-PKISyntax -- in [CPALGS]
gostR3410-94-PKISyntax

; -- GOST R 34.10-94 signature data type
GostR3410-94-Signature ::= OCTET STRING (SIZE (64))

-- GOST R 34.10-94 signature algorithm & parameters
GostR3410-94-CMSSignatureAlgorithms ALGORITHM-IDENTIFIER ::= {
  { GostR3410-94-PublicKeyParameters IDENTIFIED BY id-GostR3410-94 }
}

END -- GostR3410-94-SignatureSyntax

10.3. GostR3410-2001-SignatureSyntax

GostR3410-2001-SignatureSyntax
{ iso(1) member-body(2) ru(643) rans(2) cryptopro(2)
  other(1) modules(1) gostR3410-2001-SignatureSyntax(10) 1 }
DEFINITIONS ::= BEGIN

-- EXPORTS All --
-- The types and values defined in this module are exported for
-- use in the other ASN.1 modules contained within the Russian
-- Cryptography "GOST" & "GOST R" Specifications, and for the use
-- of other applications which will use them to access Russian
-- Cryptography services. Other applications may use them for
-- their own purposes, but this will not constrain extensions and
-- modifications needed to maintain or improve the Russian
-- Cryptography service.
IMPORTS
  gostR3410-2001-PKISyntax, ALGORITHM-IDENTIFIER,
cryptographic-Gost-Useful-Definitions
FROM Cryptographic-Gost-Useful-Definitions -- in [CPALGS]
{ iso(1) member-body(2) ru(643) rans(2)
cryptopro(2) other(1) modules(1)
cryptographic-Gost-Useful-Definitions(0) 1 }

id-GostR3410-2001,
GostR3410-2001-PublicKeyParameters -- in [CPALGS]
FROM GostR3410-2001-PKISyntax
gostR3410-2001-PKISyntax

; -- GOST R 34.10-2001 signature data type
GostR3410-2001-Signature ::= OCTET STRING (SIZE (64))

-- GOST R 34.10-2001 signature algorithms and parameters
GostR3410-2001-CMSSignatureAlgorithms
ALGORITHM-IDENTIFIER ::= {
  { GostR3410-2001-PublicKeyParameters IDENTIFIED BY
    id-GostR3410-2001 }
}
END -- GostR3410-2001-SignatureSyntax

11. Acknowledgements

This document was created in accordance with "Russian Cryptographic Software Compatibility Agreement", signed by FGUE STC "Atlas", CRYPTO-PRO, Factor-TS, MD PREI, Infotechs GmbH, SPRCIS (SPbRCZI), Cryptocom, R-Alpha. The aim of this agreement is to achieve mutual compatibility of the products and solutions.

The authors wish to thank:

Microsoft Corporation Russia for providing information about company products and solutions, and also for technical consulting in PKI.

RSA Security Russia and Demos Co Ltd for active collaboration and critical help in creation of this document.

Russ Housley (Vigil Security, LLC, housley@vigilsec.com) and Vasilij Sakharov (DEMOS Co Ltd, svp@dol.ru) for encouraging the authors to create this document.

Prikhodko Dmitriy (VSTU, PrikhodkoDV@volgablob.ru) for invaluable assistance in proofreading this document and verifying the form and the contents of the ASN.1 structures mentioned or used in this document.
12. References

12.1. Normative References


[GOST3431095] "Information technology. Cryptographic Data Security. Produce and check procedures of Electronic Digital Signature based on Asymmetric Cryptographic Algorithm.", GOST 34.310-95, Council for Standardization, Metrology and Certification of the Commonwealth of Independence States (EASC), Minsk, 1995. (In Russian)

Informative References


[RFEDSL] "Russian Federal Electronic Digital Signature Law", 10 Jan 2002 N 1-FZ.

Authors’ Addresses

Serguei Leontiev, Ed.
CRYPTO-PRO
38, Obraztsova,
Moscow, 127018, Russian Federation
EMail: lse@cryptopro.ru

Grigorij Chudov, Ed.
CRYPTO-PRO
38, Obraztsova,
Moscow, 127018, Russian Federation
EMail: chudov@cryptopro.ru

Vladimir Popov
CRYPTO-PRO
38, Obraztsova,
Moscow, 127018, Russian Federation
EMail: vpopov@cryptopro.ru

Alexandr Afanasiev
Factor-TS
office 711, 14, Presnenskij val,
Moscow, 123557, Russian Federation
EMail: afal@factor-ts.ru

Nikolaj Nikishin
Infotecs GmbH
p/b 35, 80-5, Leningradskij prospekt,
Moscow, 125315, Russian Federation
EMail: nikishin@infotecs.ru

Boleslav Izotov
FGUE STC "Atlas"
38, Obraztsova,
Moscow, 127018, Russian Federation
EMail: izotov@nii.voskhod.ru
Elena Minaeva
MD PREI
build 3, 6A, Vtoroj Troitskij per.,
Moscow, Russian Federation

EMail: evminaeva@mail.ru

Igor Ovcharenko
MD PREI
Office 600, 14, B.Novodmitrovskaya,
Moscow, Russian Federation

EMail: igori@mo.msk.ru

Sergei Murugov
R-Alpha
4/1, Raspletina,
Moscow, 123060, Russian Federation

EMail: msm@top-cross.ru

Igor Ustinov
Cryptocom
office 239, 51, Leninskiy prospekt,
Moscow, 119991, Russian Federation

EMail: igus@cryptocom.ru

Anatolij Erkin
SPRCIS (SPbRCZI)
1, Obrucheva,
St.Petersburg, 195220, Russian Federation

EMail: erkin@nevsky.net
Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).