Network Working Group
Request for Comments: 4490
Category: Standards Track

S. Leontiev, Ed.
G. Chudov, Ed.
CRYPTO-PRO

May 2006

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

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

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:

```
id-GostR3411-94 OBJECT IDENTIFIER ::=
      { iso(1) member-body(2) ru(643) rans(2) cryptopro(2)
        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:

```
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:

```
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.

```
Gost28147-89-EncryptedKey ::=   SEQUENCE {
  encryptedKey         Gost28147-89-Key,
  maskKey              [0] IMPLICIT Gost28147-89-Key
                           OPTIONAL,
  macKey               Gost28147-89-MAC
}
```

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:

```
   30 1D 06 06  2A 85 03 02  02 15 30 13  04 08 00 00
   00 00 00 00  00 00 06 07  2A 85 03 02  02 1F 01
```

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.

```
  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
```

```
        :        }
 222   10:       SEQUENCE {
 224    6:        OBJECT IDENTIFIER id-GostR3410-2001
 232    0:        NULL
        :        }
 234   64:       OCTET STRING
        :        C0 C3 42 D9 3F 8F FE 25 11 11 88 77 BF 89 C3 DB
        :        83 42 04 D6 20 F9 68 2A 99 F6 FE 30 3B E4 F4 C8
        :        F8 D5 B4 DA FB E1 C6 91 67 34 1F BC A6 7A 0D 12
        :        7B FD 10 25 C6 51 DB 8D B2 F4 8C 71 7E ED 72 A9
        :       }
        :      }
        :     }
        :    }
        :   }
```

```
|>GostR3410-2001-signed.bin
|MIIBKAYJKoZIhvcNAQcCoIIBGTCCARUCAQExDDAKBgYqhQMCAgkFADAbBgkqhkiG
|9w0BBwGgDgQMc2FtcGxlIHRleHQKMYHkMIHhAgEBMIGBMG0xHzAdBgNVBAMMFkdv
|c3RSMzQxMC0yMDAxIGV4YW1wbGUxEjAQBgNVBAoMCUNyeXB0b1BybzELMAkGA1UE
|BhMCUlUxKTAnBgkqhkiG9w0BCQEWGkdvc3RSMzQxMC0yMDAxQGV4YW1wbGUuY29t
|AhAr9cYewhG9F8fc1GJmtC4hMAoGBiqFAwICCQUAMAoGBiqFAwICEwUABEDAw0LZ
|P4/+JRERiHe/icPbg0IE1iD5aCqZ9v4wO+T0yPjVtNr74caRZzQfvKZ6DRJ7/RAl
|xlHbjbL0jHF+7XKp
|<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
```

```
          :            id-GostR3411-94-CryptoProParamSet
          :           }
          :          }
  71   67:         BIT STRING, encapsulates {
  74   64:          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
          :          }
          :         }
          :        }
 140   10:       [1] {
 142    8:        OCTET STRING 2F F0 F6 D1 86 4B 32 8A
          :        }
 152   30:       SEQUENCE {
 154    6:        OBJECT IDENTIFIER id-GostR3410-2001-CryptoPro-ESDH
 162   20:        SEQUENCE {
 164    7:         OBJECT IDENTIFIER id-Gost28147-89-None-KeyWrap
 173    9:         SEQUENCE {
 175    7:          OBJECT IDENTIFIER
          :           id-Gost28147-89-CryptoPro-A-ParamSet
          :          }
          :         }
          :        }
 184  179:       SEQUENCE {
 187  176:        SEQUENCE {
 190  129:         SEQUENCE {
 193  109:          SEQUENCE {
 195   31:           SET {
 197   29:            SEQUENCE {
 199    3:             OBJECT IDENTIFIER commonName
 204   22:             UTF8String 'GostR3410-2001 example'
          :            }
          :           }
 228   18:           SET {
 230   16:            SEQUENCE {
 232    3:             OBJECT IDENTIFIER organizationName
 237    9:             UTF8String 'CryptoPro'
          :            }
          :           }
 248   11:           SET {
 250    9:            SEQUENCE {
 252    3:             OBJECT IDENTIFIER countryName
 257    2:             PrintableString 'RU'
          :            }
          :           }
 261   41:           SET {
```

```
 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
 401    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
|MIIBpAYJKoZIhvcNAQcDoIIBlTCCAZECAQIxggFQoYIBTAIBA6BloWMwHAYGKoUD
|AgITMBIGByqFAwICJAAGByqFAwICHgEDQwAEQLNVOfRngZcrpcTZhB8n+4HtCDLm
|mtTyAHi4/4Nk6tIdsHg8ff4DwfQG5DvMFrnF9vYZNxwXuKCqx9GhlLOlNiChCgQI
|L/D20YZLMoowHgYGKoUDAgJgMBQGByqFAwICDQAwCQYHKoUDAgIfATCBszCBsDCB
|gTBtMR8wHQYDVQQDDBZHb3N0UjM0MTAtMjAwMSBleGFtcGxlMRIwEAYDVQQKDAlD
|cnlwdG9Qcm8xCzAJBgNVBAYTAlJVMSkwJwYJKoZIhvcNAQkBFhpHb3N0UjM0MTAt
|MjAwMUBleGFtcGxlLmNvbQIQK/XGHsIRvRfH3NRiZrQuIQQqMCgEIBajHOfOTukN
|8ex0aQRoHsefOu24Ox8dSn75pdnLGdXoBAST/YZ+MDgGCSqGSIb3DQEHATAdBgYq
|hQMCAhUwEwQItzXhegc1oh0GByqFAwICHwGADDmxivS/qeJlJbZVyQ==
|<GostR3410-2001-keyagree.bin
```

9.3. Enveloped 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
        :        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
```

```
        :          id-GostR3410-2001-CryptoPro-XchA-ParamSet
 190    7:         OBJECT IDENTIFIER
        :          id-GostR3411-94-CryptoProParamSet
        :         }
        :        }
 199  167:       OCTET STRING, encapsulates {
 202  164:        SEQUENCE {
 205   40:         SEQUENCE {
 207   32:          OCTET STRING
        :           6A 2F A8 21 06 95 68 9F 9F E4 47 AA 9E CB 61 15
        :           2B 7E 41 60 BC 5D 8D FB F5 3D 28 1B 18 9A F9 75
 241    4:          OCTET STRING
        :           36 6D 98 B7
        :          }
 247  120:         [0] {
 249    7:          OBJECT IDENTIFIER
        :           id-Gost28147-89-CryptoPro-A-ParamSet
 258   99:          [0] {
 260   28:           SEQUENCE {
 262    6:            OBJECT IDENTIFIER id-GostR3410-2001
 270   18:            SEQUENCE {
 272    7:             OBJECT IDENTIFIER
        :              id-GostR3410-2001-CryptoPro-XchA-ParamSet
 281    7:             OBJECT IDENTIFIER
        :              id-GostR3411-94-CryptoProParamSet
        :             }
        :            }
 290   67:           BIT STRING encapsulates {
 293   64:            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 0E 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
        :            }
        :           }
 359    8:          OCTET STRING
        :           97 95 E3 2C 2B AD 2B 0C
        :          }
        :         }
        :        }
        :       }
        :      }
 369   56:     SEQUENCE {
 371    9:      OBJECT IDENTIFIER data
 382   29:      SEQUENCE {
 384    6:       OBJECT IDENTIFIER id-Gost28147-89
 392   19:       SEQUENCE {
 394    8:        OCTET STRING BC 10 8B 1F 0B FF 34 29
```

```
 404    7:        OBJECT IDENTIFIER id-Gost28147-89-CryptoPro-A-ParamSet
          :       }
          :     }
 413   12:     [0] AA 8E 72 1D EE 4F B3 2E E3 0F A1 37
          :     }
          :   }
          :  }
          : }
```

```
|>GostR3410-2001-keytrans.bin
|MIIBpwYJKoZIhvcNAQcDoIIBmDCCAZQCAQAxggFTMIIBTwIBADCBgTBtMR8wHQYD
|VQQDDBZHb3N0UjM0MTAtMjAwMSBleGFtcGxlMRIwEAYDVQQKDAlDcnlwdG9Qcm8x
|CzAJBgNVBAYTAlJVMSkwJwYJKoZIhvcNAQkBFhpHb3N0UjM0MTAtMjAwMUBleGFt
|cGxlLmNvbQIQK/XGHsIRvRfH3NRiZrQuITAcBgYqhQMCAhMwEgYHKoUDAgIkAAYH
|KoUDAgIeAQSBpzCBpDAoBCBqL6ghBpVon5/kR6qey2EVK35BYLxdjfv1PSgbGJr5
|dQQENm2Yt6B4BgcqhQMCAh8BoGMwHAYGKoUDAgITMBIGByqFAwICJAAGByqFAwIC
|HgEDQwAEQE0rLzOQ5tyj3VUqzd/g7/sx93N+Tv+/eImKK8PNMZQESw5gSJYf28dd
|Em/askCKd7W96vLsNMsjn5uL3Z4SwPYECJeV4ywrrSsMMDgGCSqGSIb3DQEHATAd
|BgYqhQMCAhUwEwQIvBCLHwv/NCkGByqFAwICHwGADKqOch3uT7Mu4w+hNw==
|<GostR3410-2001-keytrans.bin
```

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,
        gostR3410-94-PKISyntax,
        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-GostR3410-94
        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, Infotecs 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

[CMS] Housley, R., "Cryptographic Message Syntax (CMS)", RFC 3852, July 2004.

[CPALGS] Popov, V., Kurepkin, I., and S. Leontiev, "Additional Cryptographic Algorithms for Use with GOST 28147-89, GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 Algorithms", RFC 4357, January 2006.

[CPPK] Leontiev, S., Ed. and D. Shefanovskij, Ed., "Using the GOST R 34.10-94, GOST R 34.10-2001, and GOST R 34.11-94 Algorithms with the Internet X.509 Public Key Infrastructure Certificate and CRL Profile", RFC 4491, May 2006.

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

[GOST3431195] "Information technology. Cryptographic Data Security. Cashing function.", GOST 34.311-95, Council for Standardization, Metrology and Certification of the Commonwealth of Independence States (EASC), Minsk, 1995. (In Russian)

[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)

[GOST3431004] "Information technology. Cryptographic Data Security. Formation and verification processes of (electronic) digital signature based on Asymmetric Cryptographic Algorithm.", GOST 34.310-2004, Council for Standardization, Metrology and Certification of the Commonwealth of Independence States (EASC), Minsk, 2004. (In Russian)

[GOSTR341094] "Information technology. Cryptographic Data Security. Produce and check procedures of Electronic Digital Signatures based on Asymmetric Cryptographic Algorithm.", GOST R 34.10-94, Gosudarstvennyi Standard of Russian Federation, Government Committee of the Russia for Standards, 1994. (In Russian)

[GOSTR341001] "Information technology. Cryptographic data security. Signature and verification processes of [electronic] digital signature.", GOST R 34.10-2001, Gosudarstvennyi Standard of Russian Federation, Government Committee of the Russia for Standards, 2001. (In Russian)

[GOSTR341194] "Information technology. Cryptographic Data Security. Hashing function.", GOST R 34.10-94, Gosudarstvennyi Standard of Russian Federation, Government Committee of the Russia for Standards, 1994. (In Russian)

[PROFILE] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3280, April 2002.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC3851] Ramsdell, B., "Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification", RFC 3851, July 2004.

[X.208-88] CCITT. Recommendation X.208: Specification of Abstract Syntax Notation One (ASN.1). 1988.

[X.209-88] CCITT. Recommendation X.209: Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1). 1988.

12.2. Informative References

[CRYPTOLIC] "Russian Federal Government Regulation on Licensing of Selected Activity Categories in Cryptography Area", 23 Sep 2002 N 691.

[RFC4134] Hoffman, P., "Examples of S/MIME Messages", RFC 4134, July 2005.

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

[RFLLIC] "Russian Federal Law on Licensing of Selected Activity Categories", 08 Aug 2001 N 128-FZ.

[Schneier95] B. Schneier, Applied Cryptography, Second Edition, John Wiley & Sons, Inc., 1995.

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: afa1@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

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

EMail: msm@top-cross.ru

Igor Ustinov
Cryptocom
office 239, 51, Leninskij 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

Intellectual Property

Acknowledgement

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