Stream: Independent Submission
RFC: 9215
Category: Informational
Published: March 2022
ISSN: 2070-1721
Authors: D. Baryshkov, Ed. *Linaro Ltd.*
V. Nikolaev *CryptoPro*
A. Chelpanov *InfoTeCS JSC*

RFC 9215
Using GOST R 34.10-2012 and GOST R 34.11-2012 Algorithms with the Internet X.509 Public Key Infrastructure

Abstract

This document describes encoding formats, identifiers, and parameter formats for the GOST R 34.10-2012 and GOST R 34.11-2012 algorithms for use in the Internet X.509 Public Key Infrastructure (PKI).

This specification is developed to facilitate implementations that wish to support the GOST algorithms. This document does not imply IETF endorsement of the cryptographic algorithms used in this document.

Status of This Memo

This document is not an Internet Standards Track specification; it is published for informational purposes.

This is a contribution to the RFC Series, independently of any other RFC stream. The RFC Editor has chosen to publish this document at its discretion and makes no statement about its value for implementation or deployment. Documents approved for publication by the RFC Editor are not candidates for any level of Internet Standard; see Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9215.

Copyright Notice

Table of Contents

1. Introduction

This document describes the conventions for using the GOST R 34.10-2012 signature algorithm [GOSTR3410-2012] [RFC7091] and the GOST R 34.11-2012 hash function [GOSTR3411-2012] [RFC6986] in the Internet X.509 Public Key Infrastructure (PKI) [RFC5280].

This specification defines the contents of the signatureAlgorithm, signatureValue, signature, and subjectPublicKeyInfo fields within X.509 Certificates and Certificate Revocation Lists (CRLs). For each algorithm, the appropriate alternatives for the keyUsage certificate extension are provided.

This specification is developed to facilitate implementations that wish to support the GOST algorithms. This document does not imply IETF endorsement of the cryptographic algorithms used in this document.

1.1. Requirements Language

The key words "**MUST**", "**MUST NOT**", "**REQUIRED**", "**SHALL**", "**SHALL NOT**", "**SHOULD**", "**SHOULD NOT**", "**RECOMMENDED**", "**NOT RECOMMENDED**", "**MAY**", and "**OPTIONAL**" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Signature Algorithm Support

Conforming Certificate Authorities (CAs) **MAY** use the GOST R 34.10-2012 signature algorithm to sign certificates and CRLs. This signature algorithm **MUST** always be used with the GOST R 34.11-2012 hash function. It may use a key length of either 256 bits or 512 bits.

The ASN.1 object identifier (OID) used to identify the GOST R 34.10-2012 signature algorithm with a 256-bit key length and the GOST R 34.11-2012 hash function with a 256-bit hash code is:

```
id-tc26-signwithdigest-gost3410-12-256 OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        algorithms(1) signwithdigest(3) gost3410-12-256(2)}
```

The GOST R 34.10-2012 signature algorithm with a 256-bit key length generates a digital signature in the form of two 256-bit integers: 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.

The ASN.1 OID used to identify the GOST R 34.10-2012 signature algorithm with a 512-bit key length and the GOST R 34.11-2012 hash function with a 512-bit hash code is:

```
id-tc26-signwithdigest-gost3410-12-512 OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        algorithms(1) signwithdigest(3) gost3410-12-512(3)}
```

The GOST R 34.10-2012 signature algorithm with a 512-bit key length generates a digital signature in the form of two 512-bit integers: r and s. Its octet string representation consists of 128 octets, where the first 64 octets contain the big-endian representation of s and the second 64 octets contain the big-endian representation of r.

When either of these OIDs is used as the algorithm field in an AlgorithmIdentifier structure, the encoding **MUST** omit the parameters field.

The described definition of a signature value is directly usable in the Cryptographic Message Syntax (CMS) [RFC5652], where such values are represented as octet strings. However, signature values in certificates and CRLs [RFC5280] are represented as bit strings, and thus the octet string representation must be converted.

To convert an octet string signature value to a bit string, the most significant bit of the first octet of the signature value **SHALL** become the first bit of the bit string, and so on through the least significant bit of the last octet of the signature value, which **SHALL** become the last bit of the bit string.

3. Hash Function Support

The ASN.1 OID used to identify the GOST R 34.11-2012 hash function with a 256-bit hash code is:

```
id-tc26-gost3411-12-256 OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        algorithms(1) digest(2) gost3411-12-256(2)}
```

The ASN.1 OID used to identify the GOST R 34.11-2012 hash function with a 512-bit hash code is:

```
id-tc26-gost3411-12-512 OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        algorithms(1) digest(2) gost3411-12-512(3)}
```

When either of these OIDs is used as the algorithm field in an AlgorithmIdentifier structure, the encoding **MUST** omit the parameters field.

4. Subject Public Keys Information Fields

4.1. Public Key Identifiers

GOST R 34.10-2012 public keys with a 256-bit private key length are identified by the following OID:

```
id-tc26-gost3410-12-256 OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        algorithms(1) sign(1) gost3410-12-256(1)}
```

GOST R 34.10-2012 public keys with a 512-bit private key length are identified by the following OID:

```
id-tc26-gost3410-12-512 OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        algorithms(1) sign(1) gost3410-12-512(2)}
```

4.2. Public Key Parameters

When either of these identifiers appears as the algorithm field in the SubjectPublicKeyInfo.algorithm.algorithm field, the parameters field **MUST** have the following structure:

```
GostR3410-2012-PublicKeyParameters ::= SEQUENCE
    {
        publicKeyParamSet OBJECT IDENTIFIER,
        digestParamSet OBJECT IDENTIFIER OPTIONAL
    }
```

where:

- `publicKeyParamSet` is the public key parameters identifier for GOST R 34.10-2012 parameters (see Sections 5.1.1 and 5.2.1 of [RFC7836] or Appendix C) or GOST R 34.10-2001 parameters (see Section 8.4 of [RFC4357]).
- `digestParamSet` is the parameters identifier for the corresponding GOST R 34.11-2012 parameters (see Section 3).

The following values, when used as `publicKeyParamSet`, define test public key parameter sets and **MUST NOT** be used outside of testing scenarios:

- `id-GostR3410-2001-TestParamSet`
- `id-tc26-gost-3410-2012-512-paramSetTest`

The `digestParamSet` field:

- **SHOULD** be omitted if the GOST R 34.10-2012 signature algorithm is used with a 512-bit key length
- **MUST** be present and must be equal to `id-tc26-digest-gost3411-12-256` if one of the following values is used as `publicKeyParamSet`:
  - `id-GostR3410-2001-TestParamSet`
  - `id-GostR3410-2001-CryptoPro-A-ParamSet`
  - `id-GostR3410-2001-CryptoPro-B-ParamSet`
  - `id-GostR3410-2001-CryptoPro-C-ParamSet`
  - `id-GostR3410-2001-CryptoPro-XchA-ParamSet`
  - `id-GostR3410-2001-CryptoPro-XchB-ParamSet`
- **SHOULD** be omitted if publicKeyParamSet is equal to:
  - `id-tc26-gost-3410-2012-256-paramSetA`
- **MUST** be omitted if one of the following values is used as publicKeyParamSet:
  - `id-tc26-gost-3410-2012-256-paramSetB`

- `id-tc26-gost-3410-2012-256-paramSetC`
- `id-tc26-gost-3410-2012-256-paramSetD`

4.3. Public Key Encoding

The GOST R 34.10-2012 public key **MUST** be ASN.1 DER encoded as an OCTET STRING. This encoding **SHALL** be used as the content (i.e., the value) of the subjectPublicKey field (a BIT STRING) of the SubjectPublicKeyInfo structure.

```
GostR3410-2012-256-PublicKey ::= OCTET STRING (SIZE(64))
GostR3410-2012-512-PublicKey ::= OCTET STRING (SIZE (128))
```

`GostR3410-2012-256-PublicKey` **MUST** contain 64 octets, where the first 32 octets contain the little-endian representation of the `x` coordinate of the public key and the second 32 octets contain the little-endian representation of the `y` coordinate of the public key.

`GostR3410-2012-512-PublicKey` **MUST** contain 128 octets, where the first 64 octets contain the little-endian representation of the `x` coordinate of the public key and the second 64 octets contain the little-endian representation of the `y` coordinate of the public key.

4.4. Key Usage Extension

If the KeyUsage extension is present in a certificate with the GOST R 34.10-2012 public key, the following values **MAY** be present:

- `digitalSignature` (0)
- `contentCommitment` (1)
- `keyEncipherment` (2)
- `dataEncipherment` (3)
- `keyAgreement` (4)
- `keyCertSign` (5)
- `cRLSign` (6)
- `encipherOnly` (7)
- `decipherOnly` (8)

Note that `contentCommitment` was named `nonRepudiation` in previous versions of X.509.

If the key is going to be used for key agreement, the `keyAgreement` flag **MUST** be present in the `KeyUsage` extension, with the `encipherOnly` and `decipherOnly` flags being optional. However, the `encipherOnly` and `decipherOnly` flags **MUST NOT** be present simultaneously.

5. Qualified Certificate Extensions

This section defines additional OIDs for use in qualified certificates for checking digital signatures.

5.1. Distinguished Name Additions

OGRN is the main state registration number of juridical entities.

```
OGRN ::= NUMERIC STRING (SIZE(13))
```

The corresponding OID is `1.2.643.100.1`.

SNILS is the individual insurance account number.

```
SNILS ::= NUMERIC STRING (SIZE(11))
```

The corresponding OID is `1.2.643.100.3`.

INNLE is the individual taxpayer number (ITN) of the legal entity.

```
INNLE ::= NUMERIC STRING (SIZE(10))
```

The corresponding OID is `1.2.643.100.4`.

OGRNIP is the main state registration number of individual entrepreneurs (sole traders).

```
OGRNIP ::= NUMERIC STRING (SIZE(15))
```

The corresponding OID is `1.2.643.100.5`.

IdentificationKind represents the way the receiver of the certificate was identified by the CA.

```
IdentificationKind ::= INTEGER { personal(0), remote-cert(1),
            remote-passport(2), remote-system(3) }
```

The corresponding OID is `1.2.643.100.114`.

INN is the individual taxpayer number (ITN).

```
INN ::= NUMERIC STRING (SIZE(12))
```

The corresponding OID is `1.2.643.3.131.1.1`.

5.2. Certificate Policies

The Russian national regulation body for cryptography defines several security levels of cryptographic tools. Depending on the class of cryptographic token used by the certificate owner, the following OIDs must be included in certificate policies. Certificates should include OIDs, starting from the lowest (KC1) up to the strongest applicable.

- `1.2.643.100.113.1` - class KC1
- `1.2.643.100.113.2` - class KC2
- `1.2.643.100.113.3` - class KC3
- `1.2.643.100.113.4` - class KB1
- `1.2.643.100.113.5` - class KB2
- `1.2.643.100.113.6` - class KA1

5.3. Subject Sign Tool

To denote the token or software type used by the certificate owner, the following non-critical `SubjectSignTool` extension with OID `1.2.643.100.111` should be included. It is defined as

```
SubjectSignTool ::= UTF8String(SIZE(1..200))
```

5.4. Issuer Sign Tool

To denote the tools used to generate key pairs and tools used by the CA to sign certificates, the following non-critical `IssuerSignTool` extension with OID `1.2.643.100.112` should be included. It is defined as

```
IssuerSignTool ::= SEQUENCE {
  signTool     UTF8String(SIZE(1..200)),
  cATool       UTF8String(SIZE(1..200)),
  signToolCert UTF8String(SIZE(1..100)),
  cAToolCert   UTF8String(SIZE(1..100)) }
```

where:

- `signTool` identifies tools used to create key pairs.
- `cATool` identifies tools used by the CA.
- `signToolCert` and `cAToolCert` contain the notice of the conformance of respective tools to Russian federal law on digital signatures.

6. Historical Considerations

Note that, for a significant period of time, there were no documents describing `GostR3410-2012-PublicKeyParameters`. Several old implementations have used `GostR3410-2001-PublicKeyParameters` instead. These implementations will return an error if the `digestParamSet` field is not included in public key parameters. Thus, an implementation wishing to collaborate with old implementations might want to include `digestParamSet` equal to `id-tc26-digest-gost3411-12-512` if one of the following values is used as `publicKeyParamSet`:

- `id-tc26-gost-3410-12-512-paramSetA`
- `id-tc26-gost-3410-12-512-paramSetB`

Note that the usage of `keyEncipherment` and `dataEncipherment` values for the `KeyUsage` extension is not fully defined for the GOST R 34.10-2012 public keys, so they **SHOULD** be used with additional care.

7. IANA Considerations

This document has no IANA actions.

8. Security Considerations

It is **RECOMMENDED** that applications verify signature values and subject public keys to conform to the GOST R 34.10-2012 standard [GOSTR3410-2012] [RFC7091] prior to their use.

It is **RECOMMENDED** that CAs and applications make sure that the private key for creating signatures is not used for more than its allowed validity period (typically 15 months for the GOST R 34.10-2012 algorithm).

Test parameter sets (`id-GostR3410-2001-TestParamSet` and `id-tc26-gost-3410-2012-512-paramSetTest`) **MUST NOT** be used outside of testing scenarios. The use of parameter sets not described herein is **NOT RECOMMENDED**. When different parameters are used, it is **RECOMMENDED** that they be subjected to examination by an authorized agency with approved methods of cryptographic analysis.

For security discussions concerning the use of algorithm parameters, see [ANS17] and the Security Considerations sections in [RFC4357] and [RFC7836].

9. References

9.1. Normative References

[RFC2119]

Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC4357] 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, DOI 10.17487/RFC4357, January 2006, <https://www.rfc-editor.org/info/rfc4357>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC6986] Dolmatov, V., Ed. and A. Degtyarev, "GOST R 34.11-2012: Hash Function", RFC 6986, DOI 10.17487/RFC6986, August 2013, <https://www.rfc-editor.org/info/rfc6986>.

[RFC7091] Dolmatov, V., Ed. and A. Degtyarev, "GOST R 34.10-2012: Digital Signature Algorithm", RFC 7091, DOI 10.17487/RFC7091, December 2013, <https://www.rfc-editor.org/info/rfc7091>.

[RFC7836] Smyshlyaev, S., Ed., Alekseev, E., Oshkin, I., Popov, V., Leontiev, S., Podobaev, V., and D. Belyavsky, "Guidelines on the Cryptographic Algorithms to Accompany the Usage of Standards GOST R 34.10-2012 and GOST R 34.11-2012", RFC 7836, DOI 10.17487/RFC7836, March 2016, <https://www.rfc-editor.org/info/rfc7836>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[ANS17] Alekseev, E.K., Nikolaev, V.D., and S.V. Smyshlyaev, "On the security properties of Russian standardized elliptic curves", Mathematical Aspects of Cryptography, 9:3, P. 5-32, DOI 10.4213/mvk260, 2018, <https://doi.org/10.4213/mvk260>.

[GOSTR3410-2012] "Information technology. Cryptographic data security. Signature and verification processes of [electronic] digital signature", GOST R 34.10-2012, Federal Agency on Technical Regulating and Metrology, 2012.

[GOSTR3411-2012] "Information technology. Cryptographic Data Security. Hashing function", GOST R 34.11-2012, Federal Agency on Technical Regulating and Metrology, 2012.

Appendix A. GostR3410-2012-PKISyntax

```
GostR3410-2012-PKISyntax
    { iso(1) member-body(2) ru(643) rosstandart(7)
      tc26(1) modules(0) gostR3410-2012-PKISyntax(2) }

DEFINITIONS ::=
BEGIN
-- EXPORTS All --

    -- ASN.1 TC 26 root
    id-tc26 OBJECT IDENTIFIER ::=
        { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1) }

    -- Signature algorithm
    id-tc26-sign OBJECT IDENTIFIER ::=
    { id-tc26 algorithms(1) sign(1) }

    -- Hash algorithm
    id-tc26-digest OBJECT IDENTIFIER ::=
    { id-tc26 algorithms(1) digest(2) }

    -- Public key identifiers
    id-tc26-sign-constants OBJECT IDENTIFIER ::=
    { id-tc26 constants(2) sign(1) }

    -- Public key algorithm GOST R 34.10-2012 / 256-bit identifiers
    id-tc26-gost-3410-2012-256-constants OBJECT IDENTIFIER ::=
    { id-tc26-sign-constants gost-3410-2012-256(1) }

    -- Public key algorithm GOST R 34.10-2012 / 512-bit identifiers
    id-tc26-gost-3410-2012-512-constants OBJECT IDENTIFIER ::=
    { id-tc26-sign-constants gost-3410-2012-512(2) }

    -- GOST R 34.10-2012 / 256-bit signature algorithm
    id-tc26-gost3410-12-256 OBJECT IDENTIFIER ::=
    { id-tc26-sign gost3410-12-256(1) }

    -- GOST R 34.10-2012 / 512-bit signature algorithm
    id-tc26-gost3410-12-512 OBJECT IDENTIFIER ::=
    { id-tc26-sign gost3410-12-512(2) }

    -- GOST R 34.11-2012 / 256-bit hash algorithm
    id-tc26-gost3411-12-256 OBJECT IDENTIFIER ::=
    { id-tc26-digest gost3411-12-256(2)}

    -- GOST R 34.11-2012 / 512-bit hash algorithm
    id-tc26-gost3411-12-512 OBJECT IDENTIFIER ::=
    { id-tc26-digest gost3411-12-512(3)}

    -- GOST R 34.10-2012 / GOST R 34.11-2012 sign/hash algorithm
    id-tc26-signwithdigest OBJECT IDENTIFIER ::=
    { id-tc26 algorithms(1) signwithdigest(3) }

    -- Signature & hash algorithm GOST R 34.10-2012 / 256 bits
```

```
    -- with GOST R 34.11-2012
    id-tc26-signwithdigest-gost3410-12-256 OBJECT IDENTIFIER ::=
    { id-tc26-signwithdigest gost3410-12-256(2) }

    -- Signature & hash algorithm GOST R 34.10-2012 / 512 bits
    -- with GOST R 34.11-2012
    id-tc26-signwithdigest-gost3410-12-512 OBJECT IDENTIFIER ::=
    { id-tc26-signwithdigest gost3410-12-512(3) }

    -- GOST R 34.10-2012 / 256-bit signature algorithm
    -- parameters identifier: "Set A"
    id-tc26-gost-3410-2012-256-paramSetA OBJECT IDENTIFIER ::=
    { id-tc26-gost-3410-2012-256-constants paramSetA(1) }

    -- GOST R 34.10-2012 / 256-bit signature algorithm
    -- parameters identifier: "Set B"
    id-tc26-gost-3410-2012-256-paramSetB OBJECT IDENTIFIER ::=
    { id-tc26-gost-3410-2012-256-constants paramSetB(2) }

    -- GOST R 34.10-2012 / 256-bit signature algorithm
    -- parameters identifier: "Set C"
    id-tc26-gost-3410-2012-256-paramSetC OBJECT IDENTIFIER ::=
    { id-tc26-gost-3410-2012-256-constants paramSetC(3) }

    -- GOST R 34.10-2012 / 256-bit signature algorithm
    -- parameters identifier: "Set D"
    id-tc26-gost-3410-2012-256-paramSetD OBJECT IDENTIFIER ::=
    { id-tc26-gost-3410-2012-256-constants paramSetD(4) }

    -- GOST R 34.10-2012 / 512-bit signature algorithm
    -- parameters identifier: "Test set"
    id-tc26-gost-3410-2012-512-paramSetTest OBJECT IDENTIFIER ::=
    { id-tc26-gost-3410-2012-512-constants paramSetTest(0) }

    -- GOST R 34.10-2012 / 512-bit signature algorithm
    -- parameters identifier: "Set A"
    id-tc26-gost-3410-2012-512-paramSetA OBJECT IDENTIFIER ::=
    { id-tc26-gost-3410-2012-512-constants paramSetA(1) }

    -- GOST R 34.10-2012 / 512-bit signature algorithm
    -- parameters identifier: "Set B"
    id-tc26-gost-3410-2012-512-paramSetB OBJECT IDENTIFIER ::=
    { id-tc26-gost-3410-2012-512-constants paramSetB(2) }

    -- GOST R 34.10-2012 / 512-bit signature algorithm
    -- parameters identifier: "Set C"
    id-tc26-gost-3410-2012-512-paramSetC OBJECT IDENTIFIER ::=
    { id-tc26-gost-3410-2012-512-constants paramSetC(3) }

    -- Public key GOST R 34.10-2012 / 256 bits
    GostR3410-2012-256-PublicKey ::= OCTET STRING (SIZE (64))
    -- Public key GOST R 34.10-2012 / 512 bits
    GostR3410-2012-512-PublicKey ::= OCTET STRING (SIZE (128))
    -- Public key GOST R 34.10-2012
    GostR3410-2012-PublicKey ::= OCTET STRING (SIZE (64 | 128))

    -- Public key parameters GOST R 34.10-2012
    GostR3410-2012-PublicKeyParameters ::=
```

```
        SEQUENCE {
            publicKeyParamSet OBJECT IDENTIFIER,
            digestParamSet OBJECT IDENTIFIER OPTIONAL
            }

END -- GostR3410-2012-PKISyntax
```

Appendix B. GostR3410-2012-RuStrongCertsSyntax

```
RuStrongCertsSyntax
    { iso(1) member-body(2) ru(643) rosstandart(7)
      tc26(1) modules(0) ruStrongCertsSyntax(6) }

DEFINITIONS ::=
BEGIN
-- EXPORTS All --

    id-ca OBJECT IDENTIFIER ::=
        { iso(1) member-body(2) ru(643) ca(3) }

    id-fss OBJECT IDENTIFIER ::=
        { iso(1) member-body(2) ru(643) fss(100) }

    id-fns OBJECT IDENTIFIER ::=
        { id-ca fns(131) }

    -- The main state registration number of juridical entities.
    OGRN ::= NumericString(SIZE (13))

    id-OGRN OBJECT IDENTIFIER ::=
         { id-fss ogrn(1) }

    --  The individual insurance account number.
    SNILS ::= NumericString(SIZE (11))

    id-SNILS OBJECT IDENTIFIER ::=
         { id-fss snils(3) }

    -- The main state registration number of
    -- individual entrepreneurs (sole traders).
    OGRNIP ::= NumericString(SIZE (15))

    id-OGRNIP OBJECT IDENTIFIER ::=
         { id-fss ogrnip(5) }

    id-class OBJECT IDENTIFIER ::=
         { id-fss class(113) }

    id-class-kc1 OBJECT IDENTIFIER ::=
         { id-class kc1(1) }

    id-class-kc2 OBJECT IDENTIFIER ::=
         { id-class kc2(2) }

    id-class-kc3 OBJECT IDENTIFIER ::=
```

```
          { id-class kc3(3) }

     id-class-kb1 OBJECT IDENTIFIER ::=
          { id-class kb1(4) }

     id-class-kb2 OBJECT IDENTIFIER ::=
          { id-class kb2(5) }

     id-class-ka OBJECT IDENTIFIER ::=
          { id-class ka(6) }

     -- The individual taxpayer number (ITN).
     INN ::= NumericString(SIZE (12))

     id-INN OBJECT IDENTIFIER ::=
          { id-fns ids(1) inn(1) }

     -- The organization taxpayer number (OTN).
     INNLE ::= NumericString(SIZE (10))

     id-INNLE OBJECT IDENTIFIER ::=
          { id-fss innle(4) }

     -- The token or software type used by the certificate owner.
     SubjectSignTool ::= UTF8String(SIZE(1..200))

     id-SubjectSignTool OBJECT IDENTIFIER ::=
          { id-fss subjectSignTool(111) }

     -- The tools used to generate key pairs and tools used by
     -- the CA to sign certificates.
     IssuerSignTool ::= SEQUENCE {
          signTool     UTF8String(SIZE(1..200)),
          cATool       UTF8String(SIZE(1..200)),
          signToolCert UTF8String(SIZE(1..100)),
          cAToolCert   UTF8String(SIZE(1..100)) }

     id-IssuerSignTool OBJECT IDENTIFIER ::=
          { id-fss issuerSignTool(112) }

     -- The method of identifying the owner, when it applies/receives
     -- the certificate in the CA.
     IdentificationKind ::= INTEGER { personal(0), remote-cert(1),
          remote-passport(2), remote-system(3) }

     id-IdentificationKind OBJECT IDENTIFIER ::=
          { id-fss identificationKind(114) }

END -- RuStrongCertsSyntax
```

Appendix C. Public Key Parameters

Here we define three new OIDs for three existing public key parameter sets defined in [RFC4357]. These OIDs **MUST** be used with GOST R 34.10-2012 public keys only.

```
id-tc26-gost-3410-2012-256-paramSetB OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        constants(2) sign-constants(1) gost-3410-12-256-constants(1)
          paramSetB(2)}
```

The elliptic curve of this parameter set is the same as that of `id-GostR3410-2001-CryptoPro-A-ParamSet` (and `id-GostR3410-2001-CryptoPro-XchA-ParamSet`), which can be found in [RFC4357].

```
id-tc26-gost-3410-2012-256-paramSetC OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        constants(2) sign-constants(1) gost-3410-12-256-constants(1)
          paramSetC(3)}
```

The elliptic curve of this parameter set is the same as that of `id-GostR3410-2001-CryptoPro-B-ParamSet`, which can be found in [RFC4357].

```
id-tc26-gost-3410-2012-256-paramSetD OBJECT IDENTIFIER ::=
    { iso(1) member-body(2) ru(643) rosstandart(7) tc26(1)
        constants(2) sign-constants(1) gost-3410-12-256-constants(1)
          paramSetD(4)}
```

The elliptic curve of this parameter set is the same as that of `id-GostR3410-2001-CryptoPro-C-ParamSet` (and `id-GostR3410-2001-CryptoPro-XchB-ParamSet`), which can be found in [RFC4357].

Appendix D. Test Examples

D.1. GOST R 34.10-2001 Test Parameters (256-Bit Private Key Length)

This example uses the curve defined in Section 7.1 of [RFC7091].

The private key is

```
d = 0x7A929ADE789BB9BE10ED359DD39A72C1\\
      1B60961F49397EEE1D19CE9891EC3B28
```

The public key is

```
x = 0x7F2B49E270DB6D90D8595BEC458B50C5\\
      8585BA1D4E9B788F6689DBD8E56FD80B

y = 0x26F1B489D6701DD185C8413A977B3CBB\\
      AF64D1C593D26627DFFB101A87FF77DA
```

D.1.1. Certificate Request

```
-----BEGIN CERTIFICATE REQUEST-----
MIHTMIGBAgEAMBIxEDAOBgNVBAMTB0V4YW1wbGUwZjAfBggqhQMHAQEBATATBgcq
hQMCAiMABggqhQMHAQECAgNDAARAC9hv5djbiWaPeJtOHbqFhcVQi0XsW1nYkG3b
cOJJK3/ad/+HGhD73ydm0pPF0WSvuzx7lzpByIXRHXDWibTxJqAAMAoGCCqFAwcB
AQMCA0EAaqqzjjXUqqUXlAMBeZEi2FVIT1efTLuW1jzf3zrMQypBqijS8asUgoDN
ntVv7aQZdAU1VKQnZ7g60EP9OdwEkw==
-----END CERTIFICATE REQUEST-----

  0 211: SEQUENCE {
  3 129:   SEQUENCE {
  6   1:     INTEGER 0
  9  18:     SEQUENCE {
 11  16:       SET {
 13  14:         SEQUENCE {
 15   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 20   7:           PrintableString 'Example'
       :           }
       :         }
       :       }
 29 102:     SEQUENCE {
 31  31:       SEQUENCE {
 33   8:         OBJECT IDENTIFIER '1 2 643 7 1 1 1 1'
 43  19:         SEQUENCE {
 45   7:           OBJECT IDENTIFIER '1 2 643 2 2 35 0'
 54   8:           OBJECT IDENTIFIER '1 2 643 7 1 1 2 2'
       :           }
       :         }
 64  67:       BIT STRING, encapsulates {
 67  64:         OCTET STRING
       :           0B D8 6F E5 D8 DB 89 66 8F 78 9B 4E 1D BA 85 85
       :           C5 50 8B 45 EC 5B 59 D8 90 6D DB 70 E2 49 2B 7F
       :           DA 77 FF 87 1A 10 FB DF 27 66 D2 93 C5 D1 64 AF
       :           BB 3C 7B 97 3A 41 C8 85 D1 1D 70 D6 89 B4 F1 26
       :         }
       :       }
133   0:     [0] {}
       :     }
135  10:   SEQUENCE {
137   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
       :     }
147  65:   BIT STRING
       :     6A AA B3 8E 35 D4 AA A5 17 94 03 01 79 91 22 D8
       :     55 48 4F 57 9F 4C BB 96 D6 3C DF DF 3A CC 43 2A
       :     41 AA 28 D2 F1 AB 14 82 80 CD 9E D5 6F ED A4 19
       :     74 05 35 54 A4 27 67 B8 3A D0 43 FD 39 DC 04 93
       :   }
```

D.1.2. Certificate

```
-----BEGIN CERTIFICATE-----
MIIBLTCB26ADAgECAgEKMAoGCCqFAwcBAQMCMBIxEDAOBgNVBAMTB0V4YW1wbGUw
IBcNMDEwMTAxMDAwMDAwWhgPMjA1MDEyMzEwMDAwMDBaMBIxEDAOBgNVBAMTB0V4
```

```
YW1wbGUwZjAfBggqhQMHAQEBATATBgcqhQMCAiMABggqhQMHAQECAgNDAARAC9hv
5djbiWaPeJtOHbqFhcVQi0XsW1nYkG3bcOJJK3/ad/+HGhD73ydm0pPF0WSvuzx7
lzpByIXRHXDWibTxJqMTMBEwDwYDVR0TAQH/BAUwAwEB/zAKBggqhQMHAQEDAgNB
AE1T8BL+CBd2UH1Nm7gfAO/bTu/Uq4O6xLrPc1Fzz6gcQaoo0vGrFIKAzZ7Vb+2k
GXQFNVSkJ2e4OtBD/TncBJM=
-----END CERTIFICATE-----

  0 301: SEQUENCE {
  4 219:   SEQUENCE {
  7   3:     [0] {
  9   1:       INTEGER 2
        :       }
 12   1:     INTEGER 10
 15  10:     SEQUENCE {
 17   8:       OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
        :       }
 27  18:     SEQUENCE {
 29  16:       SET {
 31  14:         SEQUENCE {
 33   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 38   7:           PrintableString 'Example'
        :           }
        :         }
        :       }
 47  32:     SEQUENCE {
 49  13:       UTCTime 01/01/2001 00:00:00 GMT
 64  15:       GeneralizedTime 31/12/2050 00:00:00 GMT
        :       }
 81  18:     SEQUENCE {
 83  16:       SET {
 85  14:         SEQUENCE {
 87   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 92   7:           PrintableString 'Example'
        :           }
        :         }
        :       }
101 102:     SEQUENCE {
103  31:       SEQUENCE {
105   8:         OBJECT IDENTIFIER '1 2 643 7 1 1 1 1'
115  19:         SEQUENCE {
117   7:           OBJECT IDENTIFIER '1 2 643 2 2 35 0'
126   8:           OBJECT IDENTIFIER '1 2 643 7 1 1 2 2'
        :           }
        :         }
136  67:       BIT STRING, encapsulates {
139  64:         OCTET STRING
        :           0B D8 6F E5 D8 DB 89 66 8F 78 9B 4E 1D BA 85 85
        :           C5 50 8B 45 EC 5B 59 D8 90 6D DB 70 E2 49 2B 7F
        :           DA 77 FF 87 1A 10 FB DF 27 66 D2 93 C5 D1 64 AF
        :           BB 3C 7B 97 3A 41 C8 85 D1 1D 70 D6 89 B4 F1 26
        :         }
        :       }
205  19:     [3] {
207  17:       SEQUENCE {
209  15:         SEQUENCE {
211   3:           OBJECT IDENTIFIER basicConstraints (2 5 29 19)
216   1:           BOOLEAN TRUE
219   5:           OCTET STRING, encapsulates {
```

```
221   3:             SEQUENCE {
223   1:               BOOLEAN TRUE
        :               }
        :             }
        :           }
        :         }
        :       }
        :     }
226  10:   SEQUENCE {
228   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
        :     }
238  65:   BIT STRING
        :     4D 53 F0 12 FE 08 17 76 50 7D 4D 9B B8 1F 00 EF
        :     DB 4E EF D4 AB 83 BA C4 BA CF 73 51 73 CF A8 1C
        :     41 AA 28 D2 F1 AB 14 82 80 CD 9E D5 6F ED A4 19
        :     74 05 35 54 A4 27 67 B8 3A D0 43 FD 39 DC 04 93
        :   }
```

D.1.3. Certificate Revocation List

```
-----BEGIN X509 CRL-----
MIGSMEECAQEwCgYIKoUDBwEBAwIwEjEQMA4GA1UEAxMHRXhhbXBsZRcNMTQwMTAx
MDAwMDAwWhcNMTQwMTAyMDAwMDAwWjAKBggqhQMHAQEDAgNBAEK/OSoU0+vpV68+
RstQv19CIaADrT0XJ1PJSpw3ox0gQaoo0vGrFIKAzZ7Vb+2kGXQFNVSkJ2e4OtBD
/TncBJM=
-----END X509 CRL-----

  0 146: SEQUENCE {
  3  65:   SEQUENCE {
  5   1:     INTEGER 1
  8  10:     SEQUENCE {
 10   8:       OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
        :       }
 20  18:     SEQUENCE {
 22  16:       SET {
 24  14:         SEQUENCE {
 26   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 31   7:           PrintableString 'Example'
        :           }
        :         }
        :       }
 40  13:     UTCTime 01/01/2014 00:00:00 GMT
 55  13:     UTCTime 02/01/2014 00:00:00 GMT
        :     }
 70  10:   SEQUENCE {
 72   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
        :     }
 82  65:   BIT STRING
        :     42 BF 39 2A 14 D3 EB E9 57 AF 3E 46 CB 50 BF 5F
        :     42 21 A0 03 AD 3D 17 27 53 C9 4A 9C 37 A3 1D 20
        :     41 AA 28 D2 F1 AB 14 82 80 CD 9E D5 6F ED A4 19
        :     74 05 35 54 A4 27 67 B8 3A D0 43 FD 39 DC 04 93
        :   }
```

D.2. GOST R 34.10-2012 TC26-256-A Parameters (256-Bit Private Key Length)

This example uses the curve defined in Appendix A.2 of [RFC7836].

The private key is

```
d = 0x3A929ADE789BB9BE10ED359DD39A72C1\\
      0B87C83F80BE18B85C041F4325B62EC1
```

The public key is

```
x = 0x99C3DF265EA59350640BA69D1DE04418\\
      AF3FEA03EC0F85F2DD84E8BED4952774

y = 0xE218631A69C47C122E2D516DA1C09E6B\\
      D19344D94389D1F16C0C4D4DCF96F578
```

D.2.1. Certificate Request

```
-----BEGIN CERTIFICATE REQUEST-----
MIHKMHkCAQAwEjEQMA4GA1UEAxMHRXhhbXBsZTBeMBcGCCqFAwcBAQEBMAsGCSqF
AwcBAgEBAQNDAARAdCeV1L7ohN3yhQ/sA+o/rxhE4B2dpgtkUJOlXibfw5l49ZbP
TU0MbPHRiUPZRJPRa57AoW1RLS4SfMRpGmMY4qAAMAoGCCqFAwcBAQMCA0EAG9wq
Exdnm2YjL2PqFv98ZMyqua2FX8bhgJFmHbedSBIdDh2lvjR8bxtSVseurCAK1krH
em9bOg4Jcxjnrm7naQ==
-----END CERTIFICATE REQUEST-----

  0 202: SEQUENCE {
  3 121:   SEQUENCE {
  5   1:     INTEGER 0
  8  18:     SEQUENCE {
 10  16:       SET {
 12  14:         SEQUENCE {
 14   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 19   7:           PrintableString 'Example'
       :           }
       :         }
       :       }
 28  94:     SEQUENCE {
 30  23:       SEQUENCE {
 32   8:         OBJECT IDENTIFIER '1 2 643 7 1 1 1 1'
 42  11:         SEQUENCE {
 44   9:           OBJECT IDENTIFIER '1 2 643 7 1 2 1 1 1'
       :           }
       :         }
 55  67:       BIT STRING, encapsulates {
 58  64:         OCTET STRING
       :           74 27 95 D4 BE E8 84 DD F2 85 0F EC 03 EA 3F AF
       :           18 44 E0 1D 9D A6 0B 64 50 93 A5 5E 26 DF C3 99
       :           78 F5 96 CF 4D 4D 0C 6C F1 D1 89 43 D9 44 93 D1
       :           6B 9E C0 A1 6D 51 2D 2E 12 7C C4 69 1A 63 18 E2
       :         }
       :       }
124   0:     [0] {}
       :     }
126  10:   SEQUENCE {
128   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
       :     }
138  65:   BIT STRING
       :     1B DC 2A 13 17 67 9B 66 23 2F 63 EA 16 FF 7C 64
       :     CC AA B9 AD 85 5F C6 E1 80 91 66 1D B7 9D 48 12
       :     1D 0E 1D A5 BE 34 7C 6F 1B 52 56 C7 AE AC 20 0A
       :     D6 4A C7 7A 6F 5B 3A 0E 09 73 18 E7 AE 6E E7 69
       :   }
```

D.2.2. Certificate

```
-----BEGIN CERTIFICATE-----
MIIBJTCB06ADAgECAgEKMAoGCCqFAwcBAQMCMBIxEDAOBgNVBAMTB0V4YW1wbGUw
IBcNMDEwMTAxMDAwMDAwWhgPMjA1MDEyMzEwMDAwMDBaMBIxEDAOBgNVBAMTB0V4
YW1wbGUwXjAXBggqhQMHAQEBATALBgkqhQMHAQIBAQEDQwAEQHQnldS+6ITd8oUP
```

```
7APqP68YROAdnaYLZFCTpV4m38OZePWWz01NDGzx0YlD2UST0WuewKFtUS0uEnzE
aRpjGOKjEzARMA8GA1UdEwEB/wQFMAMBAf8wCgYIKoUDBwEBAwIDQQAUC02pEksJ
yw1c6Sjuh0JzoxASlJLsDik2njt5EkhXjB0OHaW+NHxvG1JWx66sIArWSsd6b1s6
DglzGOeubudp
-----END CERTIFICATE-----

  0 293: SEQUENCE {
  4 211:   SEQUENCE {
  7   3:     [0] {
  9   1:       INTEGER 2
      :       }
 12   1:     INTEGER 10
 15  10:     SEQUENCE {
 17   8:       OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
      :       }
 27  18:     SEQUENCE {
 29  16:       SET {
 31  14:         SEQUENCE {
 33   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 38   7:           PrintableString 'Example'
      :           }
      :         }
      :       }
 47  32:     SEQUENCE {
 49  13:       UTCTime 01/01/2001 00:00:00 GMT
 64  15:       GeneralizedTime 31/12/2050 00:00:00 GMT
      :       }
 81  18:     SEQUENCE {
 83  16:       SET {
 85  14:         SEQUENCE {
 87   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 92   7:           PrintableString 'Example'
      :           }
      :         }
      :       }
101  94:     SEQUENCE {
103  23:       SEQUENCE {
105   8:         OBJECT IDENTIFIER '1 2 643 7 1 1 1 1'
115  11:         SEQUENCE {
117   9:           OBJECT IDENTIFIER '1 2 643 7 1 2 1 1 1'
      :           }
      :         }
128  67:       BIT STRING, encapsulates {
131  64:         OCTET STRING
      :           74 27 95 D4 BE E8 84 DD F2 85 0F EC 03 EA 3F AF
      :           18 44 E0 1D 9D A6 0B 64 50 93 A5 5E 26 DF C3 99
      :           78 F5 96 CF 4D 4D 0C 6C F1 D1 89 43 D9 44 93 D1
      :           6B 9E C0 A1 6D 51 2D 2E 12 7C C4 69 1A 63 18 E2
      :         }
      :       }
197  19:     [3] {
199  17:       SEQUENCE {
201  15:         SEQUENCE {
203   3:           OBJECT IDENTIFIER basicConstraints (2 5 29 19)
208   1:           BOOLEAN TRUE
211   5:           OCTET STRING, encapsulates {
213   3:             SEQUENCE {
215   1:               BOOLEAN TRUE
```

```
       :               }
       :             }
       :           }
       :         }
       :       }
       :     }
218  10:   SEQUENCE {
220   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
       :     }
230  65:   BIT STRING
       :     14 0B 4D A9 12 4B 09 CB 0D 5C E9 28 EE 87 42 73
       :     A3 10 12 94 92 EC 0E 29 36 9E 3B 79 12 48 57 8C
       :     1D 0E 1D A5 BE 34 7C 6F 1B 52 56 C7 AE AC 20 0A
       :     D6 4A C7 7A 6F 5B 3A 0E 09 73 18 E7 AE 6E E7 69
       :   }
```

D.2.3. Certificate Revocation List

```
-----BEGIN X509 CRL-----
MIGSMEECAQEwCgYIKoUDBwEBAwIwEjEQMA4GA1UEAxMHRXhhbXBsZRcNMTQwMTAx
MDAwMDAwWhcNMTQwMTAyMDAwMDAwWjAKBggqhQMHAQEDAgNBABS9aAh8O5A8eqKL
B/6y571v4JY/VjJnNZ9c2Oq0UFmtHQ4dpb40fG8bUlbHrqwgCtZKx3pvWzoOCXMY
565u52k=
-----END X509 CRL-----

  0 146: SEQUENCE {
  3  65:   SEQUENCE {
  5   1:     INTEGER 1
  8  10:     SEQUENCE {
 10   8:       OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
       :       }
 20  18:     SEQUENCE {
 22  16:       SET {
 24  14:         SEQUENCE {
 26   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 31   7:           PrintableString 'Example'
       :           }
       :         }
       :       }
 40  13:     UTCTime 01/01/2014 00:00:00 GMT
 55  13:     UTCTime 02/01/2014 00:00:00 GMT
       :     }
 70  10:   SEQUENCE {
 72   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 2'
       :     }
 82  65:   BIT STRING
       :     14 BD 68 08 7C 3B 90 3C 7A A2 8B 07 FE B2 E7 BD
       :     6F E0 96 3F 56 32 67 35 9F 5C D8 EA B4 50 59 AD
       :     1D 0E 1D A5 BE 34 7C 6F 1B 52 56 C7 AE AC 20 0A
       :     D6 4A C7 7A 6F 5B 3A 0E 09 73 18 E7 AE 6E E7 69
       :   }
```

D.3. GOST R 34.10-2012 Test Parameters (512-Bit Private Key Length)

This example uses the curve defined in Appendix E.

The private key is

```
d = 0x0BA6048AADAE241BA40936D47756D7C9\\
      3091A0E8514669700EE7508E508B1020\\
      72E8123B2200A0563322DAD2827E2714\\
      A2636B7BFD18AADFC62967821FA18DD4
```

The public key is

```
x = 0x115DC5BC96760C7B48598D8AB9E740D4\\
      C4A85A65BE33C1815B5C320C854621DD\\
      5A515856D13314AF69BC5B924C8B4DDF\\
      F75C45415C1D9DD9DD33612CD530EFE1

y = 0x37C7C90CD40B0F5621DC3AC1B751CFA0\\
      E2634FA0503B3D52639F5D7FB72AFD61\\
      EA199441D943FFE7F0C70A2759A3CDB8\\
      4C114E1F9339FDF27F35ECA93677BEEC
```

D.3.1. Certificate Request

```
-----BEGIN CERTIFICATE REQUEST-----
MIIBTzCBvAIBADASMRAwDgYDVQQDEwdFeGFtcGxlMIGgMBcGCCqFAwcBAQECMAsG
CSqFAwcBAgECAAOBhAAEgYDh7zDVLGEz3dmdHVxBRVz3302LTJJbvGmvFDPRVlhR
Wt0hRoUMMlxbgcEzvmVaqMTUQOe5io1ZSHsMdpa8xV0R7L53NqnsNX/y/TmTH04R
TLjNo1knCsfw5/9D2UGUGeph/Sq3f12fY1I9O1CgT2PioM9Rt8E63CFWDwvUDMnH
N6AAMAoGCCqFAwcBAQMDA4GBAEM7HWzkClHx5XN+sWqixoOCmkBbnZEn4hJg/J1q
wF2HvyTibEUnilwhkqdbqUmTq9YHTn/xvwP9L1OXr6HZRVgvhvpgoIEJGiPdeV4e
PGie5RKjyC7g3MJkPHjuqPys01SSVYSGsg8cnsGXyQaZhQJgyTvLzZxcMxfhk0Th
c642
-----END CERTIFICATE REQUEST-----

  0 335: SEQUENCE {
  4 188:   SEQUENCE {
  7   1:     INTEGER 0
 10  18:     SEQUENCE {
 12  16:       SET {
 14  14:         SEQUENCE {
 16   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 21   7:           PrintableString 'Example'
       :           }
       :         }
       :       }
 30 160:     SEQUENCE {
 33  23:       SEQUENCE {
 35   8:         OBJECT IDENTIFIER '1 2 643 7 1 1 1 2'
 45  11:         SEQUENCE {
 47   9:           OBJECT IDENTIFIER '1 2 643 7 1 2 1 2 0'
       :           }
       :         }
 58 132:       BIT STRING, encapsulates {
 62 128:         OCTET STRING
       :           E1 EF 30 D5 2C 61 33 DD D9 9D 1D 5C 41 45 5C F7
```

```
       :           DF 4D 8B 4C 92 5B BC 69 AF 14 33 D1 56 58 51 5A
       :           DD 21 46 85 0C 32 5C 5B 81 C1 33 BE 65 5A A8 C4
       :           D4 40 E7 B9 8A 8D 59 48 7B 0C 76 96 BC C5 5D 11
       :           EC BE 77 36 A9 EC 35 7F F2 FD 39 93 1F 4E 11 4C
       :           B8 CD A3 59 27 0A C7 F0 E7 FF 43 D9 41 94 19 EA
       :           61 FD 2A B7 7F 5D 9F 63 52 3D 3B 50 A0 4F 63 E2
       :           A0 CF 51 B7 C1 3A DC 21 56 0F 0B D4 0C C9 C7 37
       :         }
       :       }
193   0:     [0] {}
       :     }
195  10:   SEQUENCE {
197   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 3'
       :     }
207 129:   BIT STRING
       :     43 3B 1D 6C E4 0A 51 F1 E5 73 7E B1 6A A2 C6 83
       :     82 9A 40 5B 9D 91 27 E2 12 60 FC 9D 6A C0 5D 87
       :     BF 24 E2 6C 45 27 8A 5C 21 92 A7 5B A9 49 93 AB
       :     D6 07 4E 7F F1 BF 03 FD 2F 53 97 AF A1 D9 45 58
       :     2F 86 FA 60 A0 81 09 1A 23 DD 79 5E 1E 3C 68 9E
       :     E5 12 A3 C8 2E E0 DC C2 64 3C 78 EE A8 FC AC D3
       :     54 92 55 84 86 B2 0F 1C 9E C1 97 C9 06 99 85 02
       :     60 C9 3B CB CD 9C 5C 33 17 E1 93 44 E1 73 AE 36
       :   }
```

D.3.2. Certificate

```
-----BEGIN CERTIFICATE-----
MIIBqjCCARagAwIBAgIBCzAKBggqhQMHAQEDAzASMRAwDgYDVQQDEwdFeGFtcGxl
MCAXDTAxMDEwMTAwMDAwMFoYDzIwNTAxMjMxMDAwMDAwWjASMRAwDgYDVQQDEwdF
eGFtcGxlMIGgMBcGCCqFAwcBAQECMAsGCSqFAwcBAgECAAOBhAAEgYDh7zDVLGEz
3dmdHVxBRVz3302LTJJbvGmvFDPRVlhRWt0hRoUMMlxbgcEzvmVaqMTUQOe5io1Z
SHsMdpa8xV0R7L53NqnsNX/y/TmTH04RTLjNo1knCsfw5/9D2UGUGeph/Sq3f12f
Y1I9O1CgT2PioM9Rt8E63CFWDwvUDMnHN6MTMBEwDwYDVR0TAQH/BAUwAwEB/zAK
BggqhQMHAQEDAwOBgQBBVwPYkvGl8/aMQ1MYmn7iB7gLVjHvnUlSmk1rVCws+hWq
LqzxH0cP3n2VSFaQPDX9j5Ve8wDZXHdTSnJKDu5wL4b6YKCBCRoj3XleHjxonuUS
o8gu4NzCZDx47qj8rNNUklWEhrIPHJ7Bl8kGmYUCYMk7y82cXDMX4ZNE4XOuNg==
-----END CERTIFICATE-----

  0 426: SEQUENCE {
  4 278:   SEQUENCE {
  8   3:     [0] {
 10   1:       INTEGER 2
       :       }
 13   1:     INTEGER 11
 16  10:     SEQUENCE {
 18   8:       OBJECT IDENTIFIER '1 2 643 7 1 1 3 3'
       :       }
 28  18:     SEQUENCE {
 30  16:       SET {
 32  14:         SEQUENCE {
 34   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 39   7:           PrintableString 'Example'
       :           }
       :         }
       :       }
 48  32:     SEQUENCE {
```

```
 50  13:       UTCTime 01/01/2001 00:00:00 GMT
 65  15:       GeneralizedTime 31/12/2050 00:00:00 GMT
       :       }
 82  18:     SEQUENCE {
 84  16:       SET {
 86  14:         SEQUENCE {
 88   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 93   7:           PrintableString 'Example'
       :           }
       :         }
       :       }
102 160:     SEQUENCE {
105  23:       SEQUENCE {
107   8:         OBJECT IDENTIFIER '1 2 643 7 1 1 1 2'
117  11:         SEQUENCE {
119   9:           OBJECT IDENTIFIER '1 2 643 7 1 2 1 2 0'
       :           }
       :         }
130 132:       BIT STRING, encapsulates {
134 128:         OCTET STRING
       :           E1 EF 30 D5 2C 61 33 DD D9 9D 1D 5C 41 45 5C F7
       :           DF 4D 8B 4C 92 5B BC 69 AF 14 33 D1 56 58 51 5A
       :           DD 21 46 85 0C 32 5C 5B 81 C1 33 BE 65 5A A8 C4
       :           D4 40 E7 B9 8A 8D 59 48 7B 0C 76 96 BC C5 5D 11
       :           EC BE 77 36 A9 EC 35 7F F2 FD 39 93 1F 4E 11 4C
       :           B8 CD A3 59 27 0A C7 F0 E7 FF 43 D9 41 94 19 EA
       :           61 FD 2A B7 7F 5D 9F 63 52 3D 3B 50 A0 4F 63 E2
       :           A0 CF 51 B7 C1 3A DC 21 56 0F 0B D4 0C C9 C7 37
       :         }
       :       }
265  19:     [3] {
267  17:       SEQUENCE {
269  15:         SEQUENCE {
271   3:           OBJECT IDENTIFIER basicConstraints (2 5 29 19)
276   1:           BOOLEAN TRUE
279   5:           OCTET STRING, encapsulates {
281   3:             SEQUENCE {
283   1:               BOOLEAN TRUE
       :               }
       :             }
       :           }
       :         }
       :       }
       :     }
286  10:   SEQUENCE {
288   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 3'
       :     }
298 129:   BIT STRING
       :     41 57 03 D8 92 F1 A5 F3 F6 8C 43 53 18 9A 7E E2
       :     07 B8 0B 56 31 EF 9D 49 52 9A 4D 6B 54 2C 2C FA
       :     15 AA 2E AC F1 1F 47 0F DE 7D 95 48 56 90 3C 35
       :     FD 8F 95 5E F3 00 D9 5C 77 53 4A 72 4A 0E EE 70
       :     2F 86 FA 60 A0 81 09 1A 23 DD 79 5E 1E 3C 68 9E
       :     E5 12 A3 C8 2E E0 DC C2 64 3C 78 EE A8 FC AC D3
       :     54 92 55 84 86 B2 0F 1C 9E C1 97 C9 06 99 85 02
       :     60 C9 3B CB CD 9C 5C 33 17 E1 93 44 E1 73 AE 36
       :   }
```

D.3.3. Certificate Revocation List

```
-----BEGIN X509 CRL-----
MIHTMEECAQEwCgYIKoUDBwEBAwMwEjEQMA4GA1UEAxMHRXhhbXBsZRcNMTQwMTAx
MDAwMDAwWhcNMTQwMTAyMDAwMDAwWjAKBggqhQMHAQEDAwOBgQA6E/t67NtVYO72
E3z8XdZGkXMuv7NpCh/Ax+ik7uoIMH1kjU3AmGxGqHs/vkx69C6jQ1nHlZVMo5/z
q77ZBR9NL4b6YKCBCRoj3XleHjxonuUSo8gu4NzCZDx47qj8rNNUklWEhrIPHJ7B
l8kGmYUCYMk7y82cXDMX4ZNE4XOuNg==
-----END X509 CRL-----

  0 211: SEQUENCE {
  3  65:   SEQUENCE {
  5   1:     INTEGER 1
  8  10:     SEQUENCE {
 10   8:       OBJECT IDENTIFIER '1 2 643 7 1 1 3 3'
       :       }
 20  18:     SEQUENCE {
 22  16:       SET {
 24  14:         SEQUENCE {
 26   3:           OBJECT IDENTIFIER commonName (2 5 4 3)
 31   7:           PrintableString 'Example'
       :           }
       :         }
       :       }
 40  13:     UTCTime 01/01/2014 00:00:00 GMT
 55  13:     UTCTime 02/01/2014 00:00:00 GMT
       :     }
 70  10:   SEQUENCE {
 72   8:     OBJECT IDENTIFIER '1 2 643 7 1 1 3 3'
       :     }
 82 129:   BIT STRING
       :     3A 13 FB 7A EC DB 55 60 EE F6 13 7C FC 5D D6 46
       :     91 73 2E BF B3 69 0A 1F C0 C7 E8 A4 EE EA 08 30
       :     7D 64 8D 4D C0 98 6C 46 A8 7B 3F BE 4C 7A F4 2E
       :     A3 43 59 C7 95 95 4C A3 9F F3 AB BE D9 05 1F 4D
       :     2F 86 FA 60 A0 81 09 1A 23 DD 79 5E 1E 3C 68 9E
       :     E5 12 A3 C8 2E E0 DC C2 64 3C 78 EE A8 FC AC D3
       :     54 92 55 84 86 B2 0F 1C 9E C1 97 C9 06 99 85 02
       :     60 C9 3B CB CD 9C 5C 33 17 E1 93 44 E1 73 AE 36
       :   }
```

Appendix E. GOST R 34.10-2012 Test Parameters (Curve Definition)

The following parameters must be used for digital signature generation and verification.

E.1. Elliptic Curve Modulus

The following value is assigned to parameter p in this example:

```
p = 36239861022290036359077887536838743060213209255346786050\\
    86546150450856166624002482588482022271496854025090823603\\
    05873516373426382237196498722858290737240 3
```

```
p = 0x4531ACD1FE0023C7550D267B6B2FEE80922B14B2FFB90F04D4EB7C\\
      09B5D2D15DF1D852741AF4704A0458047E80E4546D35B8336FAC22\\
      4DD81664BBF528BE6373
```

E.2. Elliptic Curve Coefficients

Parameters a and b take the following values in this example:

```
a = 7
```

```
a = 0x7
```

```
b = 15186550692108285345089500347140431549287475277402064361\\
    94018823352809982443793732829756914785974674866041605397\\
    88367759662632641399013695904743581 1826396
```

```
b = 0x1CFF0806A31116DA29D8CFA54E57EB748BC5F377E49400FDD788B6\\
      49ECA1AC4361834013B2AD7322480A89CA58E0CF74BC9E540C2ADD\\
      6897FAD0A3084F302ADC
```

E.3. Elliptic Curve Points Group Order

Parameter m takes the following value in this example:

```
m = 36239861022290036359077887536838743060213209255346786050\\
    86546150450856166623969164898305032863068499961404079437\\
    93658545586519221297073480881261812 0619743
```

```
m = 0x4531ACD1FE0023C7550D267B6B2FEE80922B14B2FFB90F04D4EB7C\\
      09B5D2D15DA82F2D7ECB1DBAC719905C5EECC423F1D86E25EDBE23\\
      C595D644AAF187E6E6DF
```

E.4. Order of Cyclic Subgroup of Elliptic Curve Points Group

Parameter q takes the following value in this example:

```
q = 36239861022290036359077887536838743060213209255346786050\\
    86546150450856166623969164898305032863068499961404079437\\
    93658545586519221297073480881261812061974 3
```

```
q = 0x4531ACD1FE0023C7550D267B6B2FEE80922B14B2FFB90F04D4EB7C\\
      09B5D2D15DA82F2D7ECB1DBAC719905C5EECC423F1D86E25EDBE23\\
      C595D644AAF187E6E6DF
```

E.5. Elliptic Curve Point Coordinates

Point P coordinates take the following values in this example:

```
x = 19283569440670228493993094012431375989977866354595079743\\
    57075491307766592685835441065557681003184874819658004903\\
    21233288425233583025072952763238349357 3274
```

```
x = 0x24D19CC64572EE30F396BF6EBBFD7A6C5213B3B3D7057CC825F910\\
      93A68CD762FD60611262CD838DC6B60AA7EEE804E28BC849977FAC\\
      33B4B530F1B120248A9A
```

```
y = 22887286933719728599700121555294784163535623273295061803\\
    14497425931102860301572814141997072271708807066593850650\\
    33415238185734779888586480760509872401 3854
```

```
y = 0x2BB312A43BD2CE6E0D020613C857ACDDCFBF061E91E5F2C3F32447\\
      C259F39B2C83AB156D77F1496BF7EB3351E1EE4E43DC1A18B91B24\\
      640B6DBB92CB1ADD371E
```

Contributors

Semen Pianov
InfoTeCS JSC
Email: Semen.Pianov@infotecs.ru

Ekaterina Karelina
InfoTeCS JSC
Email: Ekaterina.Karelina@infotecs.ru

Dmitry Belyavsky
Cryptocom
Email: beldmit@gmail.com

Authors' Addresses

Dmitry Baryshkov (EDITOR)
Linaro Ltd.
Harston Mill Royston Rd
Harston, Cambridge
CB22 7GG
United Kingdom
Email: dbaryshkov@gmail.com

Vasily Nikolaev
CryptoPro
18, Suschevsky val
Moscow
127018
Russian Federation
Phone: +7 (495) 995-48-20
Email: nikolaev@cryptopro.ru

Alexander Chelpanov
InfoTeCS JSC
Email: Aleksandr.Chelpanov@infotecs.ru