-
Internet-Draft
Intended status: Standards Track
Expires: March 12, 2007

G. Chudov, Ed.
S. Leontiev, Ed.
CRYPTO-PRO
September 8, 2006

GOST 28147-89 Cipher Suites for Transport Layer Security (TLS)
draft-chudov-cryptopro-cptls-03.txt

Status of this Memo

Copyright Notice

Abstract

This document is intended to register new cipher suites for the Transport Layer Security (TLS) protocol, according to the procedure specified in TLS Protocol standards. These cipher suites are based on Russian national cryptographic standards - GOST R 34.10-94 and GOST R 34.10-2001 public keys, GOST 28147-89 encryption algorithm and GOST R 34.11-94 digest algorithm.

Table of Contents

1. Introduction

This document proposes the addition of new cipher suites to the Transport Layer Security (TLS) protocol to support GOST R 34.11-94 digest, GOST 28147-89 encryption and VKO GOST R 34.10-94/2001 key exchange algorithms. The cipher suites defined here were proposed by CRYPTO-PRO Company for the "Russian Cryptographic Software Compatibility Agreement" community.

Algorithms GOST R 34.10-94, GOST R 34.10-2001, GOST 28147-89 and GOST R 34.11-94 have been developed by Russian Federal Agency of Governmental Communication and Information (FAGCI) and "All-Russian Scientific and Research Institute of Standardization". They are described in [GOSTR341094], [GOSTR341001], [GOSTR341194] and [GOST28147] ([GOST3431095], [GOST3431004], [GOST3431195]). Algorithms VKO GOST R 34.10-94/2001 and PRF_GOSTR3411 are described in [CPALGS].

This document defines two configurations:

- anonymous client - authenticated server (only server provides a certificate);
- authenticated client - authenticated server (client and server exchange certificates).

The presentation language used here is the same as in [TLS1.2]. Since this specification extends [TLS1.2], these descriptions should be merged with those in the TLS specification and any others that extend TLS. This means, that enum types may not specify all possible values and structures with multiple formats chosen with a select() clause may not indicate all possible cases.

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. CipherSuite Definitions

2.1. Key Exchange

The cipher suites defined here use the following key exchange algorithms:

CipherSuite	Key Exchange Algorithm
TLS_GOSTR341094_WITH_28147_CNT_IMIT	VKO GOST R 34.10-94
TLS_GOSTR341001_WITH_28147_CNT_IMIT	VKO GOST R 34.10-2001
TLS_GOSTR341094_WITH_NULL_GOSTR3411	VKO GOST R 34.10-94
TLS_GOSTR341001_WITH_NULL_GOSTR3411	VKO GOST R 34.10-2001

Key derivation algorithms based on GOST R 34.10-94 and GOST R 34.10-2001 public keys (VKO GOST R 34.10-94, VKO GOST R 34.10-2001) are described in [CPALGS].

2.2. PRF, Signature and Hash

For a PRF, described in section 5 of [TLS1.2], the cipher suites described here use PRF_GOSTR3411 (refer to Section 3.1). The same PRF MUST be used for all dependent protocols, such as [EAP-TLS].

GOST R 34.10-94/2001 signature is used for CertificateVerify message.

GOST R 34.11 digest algorithm ([GOSTR341194]) is used for CertificateVerify.signature.gostR3411_hash and Finished.verify_data (see sections 7.4.10 and 7.4.11 of [TLS1.2])

2.3. Cipher and MAC

The following cipher algorithm and MAC functions are used (for details refer to Section 3.1):

CipherSuite	Cipher	MAC
TLS_GOSTR341094_WITH_28147_CNT_IMIT	GOST28147	IMIT_GOST28147
TLS_GOSTR341001_WITH_28147_CNT_IMIT	GOST28147	IMIT_GOST28147
TLS_GOSTR341094_WITH_NULL_GOSTR3411	-	HMAC_GOSTR3411
TLS_GOSTR341001_WITH_NULL_GOSTR3411	-	HMAC_GOSTR3411

For all four cipher suites, the use of MAC is slighttly different from the one, described in section 6.2.3.1 of [TLS1.2]. In [TLS1.2], MAC is calculated from the following data:

```
MACed_data[seq_num] = seq_num +
                      TLSCompressed.type +
                      TLSCompressed.version +
                      TLSCompressed.length +
                      TLSCompressed.fragment;
```


These cipher suites use the same input for first record, but for each next record the input from all previous records is concatenated:

```
MACed_data[0] + ... + MACed_data[n]
```

3. Data Structures and Computations

3.1. Algorithms

GOST 28147-89 [GOST28147] uses 256-bit key size and 8-byte IV. Cipher suites, defined here, use GOST 28147-89 as a stream cipher in counter mode with S-box from id-Gost28147-89-CryptoPro-A-ParamSet (see [CPALGS]) and CryptoPro key meshing algorithm.

IMIT_GOST28147 is GOST 28147-89 [GOST28147] in "IMITOVSTAVKA" mode (4 bytes)

HMAC_GOSTR3411(secret, data) is based on GOST R 34.11-94 digest and described in [CPALGS].

PRF_GOSTR3411(secret, label, seed) is based on HMAC_GOSTR3411 and

described in [CPALGS].

3.2. Keys Calculation

Key calculation is done according to section 6.3 of [TLS1.2], with PRF_GOSTR3411 function used instead of PRF. The parameters are as follows:

SecurityParameters.hash_size = 32
SecurityParameters.key_material_length = 32
SecurityParameters.IV_size = 8

Length of necessary key material is 144 bytes.

3.3. Server Certificate

For these cipher suites this message is required and it MUST contain a certificate, with a public key algorithm matching ServerHello.cipher_suite.

3.4. Server Key Exchange

This message MUST NOT be used in these cipher suites, because all the parameters necessary are present in server certificate (see [CPPK]).

3.5. Certificate Request

This message is used as described in section 7.4.5 of [TLS1.2], and extended as follows:

```
 enum {
     gostr341094(21), gostr34102001(22),(255)
 } ClientCertificateType;
```

gostr341094 and gostr34102001 certificate types identify that the server accepts GOST R 34.10-94 and GOST R 34.10-2001 public key certificates.

[IANA please remove] Note: The above numeric definitions for ClientCertificateType have not yet been registered.

```
enum{
  gostr3411(XX), (255)
} HashType;
```

gostr3411 hash type identifes that the server accepts GOST R 34.11-94 hash function. It is RECOMMENDED to populate CertificateRequest.certificate_hash only with gostr3411 value, when one of the cipher suites described in this document is chosen.

3.6. Client Key Exchange Message

This message is used as described in section 7.4.9 of [TLS1.2], it is required for these suites, and contains DER-encoded TLSGostKeyTransportBlob structure [X.660].

```
enum { vko_gost } KeyExchangeAlgorithm;

struct {
    select (KeyExchangeAlgorithm) {
        case vko_gost: TLSGostKeyTransportBlob;
    } exchange_keys;
} ClientKeyExchange;
```

ASN1-syntax for this structure is:


```
TLSGostKeyTransportBlob ::= SEQUENCE {
    keyBlob GostR3410-KeyTransport,
    proxyKeyBlobs SEQUENCE OF TLSProxyKeyTransportBlob OPTIONAL
}

TLSProxyKeyTransportBlob ::= SEQUENCE {
    keyBlob GostR3410-KeyTransport,
    cert    OCTET STRING
```

}

GostR3410-KeyTransport is defined in [CPCMS].

keyBlob.transportParameters MUST be present.

keyBlob.transportParameters.ephemeralPublicKey MUST be present if the server didn't request client certificate or client's public key algorithm and parameters do not match those of the recipient. Else it SHOULD be omited.

proxyKeyBlobs - (optional) contains key exchange for secondary recipients (for example, for the firewall, which audits connections).

cert - contains secondary recipient's certificate.

Actions of client:

First, the client generates a random 32-byte premaster_secret.

Then shared_ukm is calculated as first 8 bytes of digest of concatenated client random and server random:

shared_ukm = GOSTR3411(client_random|server_random)[0..7]

Then client chooses a sender key. If keyBlob.transportParameters.ephemeralPublicKey is present, the corresponding secret key MUST be used as a sender key. If it is missing, the secret key, corresponding to the client certificate MUST be used.

Using the sender key and recipient's public key, algorithm VKO GOST R 34.10-94 or VKO GOST R 34.10-2001 (described in [CPALGS]) is applied to produce KEK. VKO GOST R 34.10-2001 is used with shared_ukm as UKM.

Then CryptoPro Key Wrap algorithm is applied to encrypt premaster_secret and produce CEK_ENC and CEK_MAC. Again, shared_ukm is used as UKM. keyBlob.transportParameters.encryptionParamSet is used for all encryption operations.

The resulting encrypted key (CEK_ENC) is placed in

keyBlob.sessionEncryptedKey.encryptedKey field, it's mac (CEK_MAC) is placed in keyBlob.sessionEncryptedKey.macKey field, and shared_ukm (UKM) is placed in keyBlob.transportParameters.ukm field.

Actions of server:

Server MUST verify, that keyBlob.transportParameters.ukm is equal to GOSTR3411(client_random|server_random)[0..7], before decrypting the premaster_secret.

Server applies VKO GOST R 34.10-94 or VKO GOST R 34.10-2001, (depending on the client public key type), and CryptoPro Key Unwrap algorithm in the simillar manner to decrypt the premaster_secret.

Server MUST verify keyBlob.sessionEncryptedKey.macKey after decrypting the premaster_secret.

3.7. Certificate Verify

This message is used as described in section 7.4.10 of [TLS1.2]. If the client have sent both a client certificate and an ephemeral public key, it MUST send a certificate verify message, as a proof of possession of the private key for provided certificate.

The TLS structures are extended as follows:

```
 enum { gostr341094, gostr34102001 }
     SignatureAlgorithm;

 select (SignatureAlgorithm) {
     case gostr341094:
         digitally-signed struct {
             opaque gostr341194_hash[32];
         };
     case gostr34102001:
         digitally-signed struct {
             opaque gostr341194_hash[32];
         };
 } Signature;

 CertificateVerify.signature.gostR3411_hash =
     GOSTR3411(handshake_messages)
```

3.8. Finished

This message is used as described in section 7.4.11 of [TLS1.2].

```
Finished.verify_data = PRF_GOSTR3411(master_secret, finished_label,
                            GOSTR3411(handshake_messages)) [0..11]
```

4. Compatibility

Some applications use the cipher suites specified herein with [TLS1.0], using features of [TLS1.2], including cipher-suite dependent PRF, Finished and Certificate Verify computations.

5. Security Considerations

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

Use of the same key for signature and key derivation is NOT RECOMMENDED.

It is RECOMMENDED for both client and server to verify the private key usage period, if this extension is present in the certificate.

The cipher suites TLS_GOSTR341094_WITH_28147_CNT_IMIT and TLS_GOSTR341001_WITH_28147_CNT_IMIT proposed hereby, have been analyzed by special certification laboratory of Scientific and Technical Centre "ATLAS" in appropriate levels of target_of_evaluation (TOE).

It is RECOMMENDED to subject the implementations of these cipher suites to examination by an authorized agency with approved methods of cryptographic analysis.

6. IANA Considerations

IANA has assigned the following values for GOST 28147-89 mode ciphers definitions:

```
 enum {
     gostr341094(21), gostr34102001(22)
 } ClientCertificateType;
```

```
    enum{
      gostr3411(XX)
    } HashType;
```

```
   CipherSuite TLS_GOSTR341094_WITH_28147_CNT_IMIT = {0x00,0x80}
   CipherSuite TLS_GOSTR341001_WITH_28147_CNT_IMIT = {0x00,0x81}
   CipherSuite TLS_GOSTR341094_WITH_NULL_GOSTR3411 = {0x00,0x82}
   CipherSuite TLS_GOSTR341001_WITH_NULL_GOSTR3411 = {0x00,0x83}
```

[IANA please remove] Note: The above numeric definitions for CipherSuites and ClientCertificateType have not yet been registered.

7. References

7.1. Normative references

[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.

[CPCMS] Leontiev, S. and G. Chudov, "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)", RFC 4490, May 2006.

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

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

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

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

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

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

[GOSTR341194]
Government Committee of the Russia for Standards,

"Information technology. Cryptographic Data Security. Hashing function, Gosudarstvennyi Standard of Russian Federation (In Russian)", GOST R 34.11-94, 1994.

[TLS1.2] Dierks, T. and E. Rescorla, "The TLS Protocol", draft-ietf-tls-rfc4346-bis-01 (work in progress), June 2006.

7.2. Informative references

[EAP-TLS] Aboba, B. and D. Simon, "PPP EAP TLS Authentication Protocol", RFC 2716, October 1999.

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

[TLS1.0] Dierks, T. and C. Allen, "The TLS Protocol Version 1.0", RFC 2246, January 1999.

[X.660] ISO/IEC, "ITU-T Recommendation X.660 Information Technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T X.660, 1997.

Appendix A. ASN.1 Modules

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

A.1. Gost-CryptoPro-TLS

```
Gost-CryptoPro-TLS
    { iso(1) member-body(2) ru(643) rans(2)
      cryptopro(2) other(1) modules(1) gost-CryptoPro-TLS(16) 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
        Certificate,
        AlgorithmIdentifier
        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)}
        id-CryptoPro-algorithms, gostR3410-EncryptionSyntax
        FROM Cryptographic-Gost-Useful-Definitions
            { iso(1) member-body(2) ru(643) rans(2)
              cryptopro(2) other(1) modules(1)
              cryptographic-Gost-Useful-Definitions(0) 1 }
        GostR3410-KeyTransport
        FROM GostR3410-EncryptionSyntax
             gostR3410-EncryptionSyntax
    ;
    id-PRF-GostR3411-94 OBJECT IDENTIFIER ::=
        { id-CryptoPro-algorithms prf-gostr3411-94(23) }
    TLSProxyKeyTransportBlob ::=
```



```
        SEQUENCE {
            keyBlob GostR3410-KeyTransport,
            cert    OCTET STRING
        }
    TLSGostKeyTransportBlob ::=
        SEQUENCE {
            keyBlob GostR3410-KeyTransport,
            proxyKeyBlobs SEQUENCE OF
                TLSProxyKeyTransportBlob OPTIONAL
        }
    TLSGostSrvKeyExchange ::=
        SEQUENCE OF
            OCTET STRING (CONSTRAINED BY {Certificate})
    TLSGostExtensionHashHMACSelect ::=
        SEQUENCE {
            hashAlgorithm AlgorithmIdentifier,
            hmacAlgorithm AlgorithmIdentifier,
            prfAlgorithm AlgorithmIdentifier
```

```
        }
    TLSGostExtensionHashHMACSelectClient ::=
        SEQUENCE OF
            TLSGostExtensionHashHMACSelect
    TLSGostExtensionHashHMACSelectServer ::=
        TLSGostExtensionHashHMACSelect

END -- Gost-CryptoPro-TLS
```

Appendix B. Acknowledgments

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 provided information about company products and solutions, and also for technical consulting in PKI.
RSA Security Russia and Demos Co Ltd for active colaboration and critical help in creation of this document.
NIP Informzachita for compatibility testing of the proposed data formats while incorporating them into company products.
Citrix Inc for help in compatibility testing Citrix products for Microsoft Windows.

Russ Hously (Vigil Security, LLC, housley@vigilsec.com) and Vasilij Sakharov (DEMOS Co Ltd, svp@dol.ru) for initiative, creating this document.

Author's Addresses

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

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

Authors' Addresses

Grigorij S. Chudov (editor)
CRYPTO-PRO, Ltd.

38, Obraztsova
Moscow 127018
Russia

Phone: +7 (495) 933 11 68
Fax: +7 (495) 933 11 68
Email: chudov@CryptoPro.ru
URI: http://www.CryptoPro.ru

Serguei E. Leontiev (editor)
CRYPTO-PRO, Ltd.
38, Obraztsova
Moscow 127018
Russia

Phone: +7 (495) 933 11 68
Fax: +7 (495) 933 11 68
Email: lse@CryptoPro.ru
URI: http://www.CryptoPro.ru

Full Copyright Statement

Intellectual Property

Acknowledgment

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