Workgroup: Network Working Group
Internet-Draft:
draft-smyshlyaev-tls13-gost-suites-05
Published: 10 December 2021
Intended Status: Informational
Expires: 13 June 2022
Authors: S.V. Smyshlyaev, Ed. E.K. Alekseev
CryptoPro CryptoPro
E.S. Griboedova A.A. Babueva L.O. Nikiforova
CryptoPro CryptoPro CryptoPro

GOST Cipher Suites for Transport Layer Security (TLS) Protocol Version 1.3

Abstract

The purpose of this document is to make the Russian cryptographic standards available to the Internet community for their implementation in the Transport Layer Security (TLS) Protocol Version 1.3.

This specification defines four new cipher suites, seven new signature schemes and a key exchange mechanism for the TLS 1.3 protocol, that are based on Russian cryptographic standards (called GOST algorithms). Additionally, this document specifies a profile of TLS 1.3 with GOST algorithms so that implementers can produce interoperable implementations. This document does not imply IETF endorsement of the cipher suites, signature schemes key exchange mechanism.

Status of This Memo

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

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

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

This Internet-Draft will expire on 13 June 2022.

Copyright Notice

Table of Contents

1. Introduction

This document defines four new cipher suites (the TLS13_GOST cipher suites) and seven new signature schemes (the TLS13_GOST signature schemes) for the Transport Layer Security (TLS) Protocol Version 1.3, that are based on Russian cryptographic standards GOST R 34.12-2015 [RFC7801], GOST R 34.11-2012 [RFC6986] and GOST R 34.10-2012 [RFC7091].

The TLS13_GOST cipher suites (see Section 4) have the following values:

```
TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_L = {0xC1, 0x03};
TLS_GOSTR341112_256_WITH_MAGMA_MGM_L = {0xC1, 0x04};
TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_S = {0xC1, 0x05};
TLS_GOSTR341112_256_WITH_MAGMA_MGM_S = {0xC1, 0x06}.
```

Each TLS13_GOST cipher suite specifies a pair (record protection algorithm, hash algorithm) such that:

- The record protection algorithm is the AEAD algorithm (see Section 4.1.1) based on the GOST R 34.12-2015 block cipher [RFC7801] in the Multilinear Galois Mode (MGM) [RFC9058] and the external re-keying approach (see [RFC8645]) intended for increasing the lifetime of symmetric keys used to protect records.

- The hash algorithm is the GOST R 34.11-2012 algorithm [RFC6986].

Note: The TLS13_GOST cipher suites are divided into two types (depending on the key lifetime limitations, see Section 4.1.2 and Section 4.1.3): the "_S" (strong) cipher suites and the "_L" (light) cipher suites.

The TLS13_GOST signature schemes have the following values:

```
gostr34102012_256a = 0x0709;

gostr34102012_256b = 0x070A;

gostr34102012_256c = 0x070B;

gostr34102012_256d = 0x070C;
```

gostr34102012_512a = 0x070D;

gostr34102012_512b = 0x070E;

gostr34102012_512c = 0x070F.

Each TLS13_GOST signature scheme specifies a pair (signature algorithm, elliptic curve) such that:

- The signature algorithm is the GOST R 34.10-2012 algorithm [RFC7091].

- The elliptic curve is one of the curves defined in Section 5.2.

Also, this document specifies the key exchange mechanism with GOST algorithms for TLS 1.3 protocol (see Section 6.1).

Additionally, this document specifies a TLS13_GOST profile of the TLS 1.3 protocol with GOST algorithms so that implementers can produce interoperable implementations. It uses TLS13_GOST cipher suites, TLS13_GOST signature schemes, key exchange mechanism that defined in this document.

2. Conventions Used in This Document

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.

3. Basic Terms and Definitions

This document uses the following terms and definitions for the sets and operations on the elements of these sets:

B_t the set of byte strings of length t, t >= 0, for t = 0 the B_t set consists of a single empty string of zero length. If A is an element of B_t, then A = (a_1, a_2, ... , a_t), where a_1, a_2, ... , a_t are in {0, ... , 255};

B* the set of all byte strings of a finite length (hereinafter referred to as strings), including the empty string;

A[i..j] the string A[i..j] = (a_i, a_{i+1}, ... , a_j) in B_{j-i+1}, where A = (a_1, a_2, ... , a_t) in B_t and 1<=i<=j<=t;

A[i] the integer a_i in {0, ... , 255}, where A = (a_1, a_2, ... , a_t) in B_t and 1<=i<=t;

|A|
: the length of the byte string A in bytes;

A | C the concatenation of strings A and C both belonging to B*, i.e., a string in B_{|A|+|C|}, where the left substring in B_|A| is equal to A, and the right substring in B_|C| is equal to C;

i & j bitwise AND of integers i and j;

STR_t the byte string STR_t(i) = (i_1, ... , i_t) in B_t corresponding to an integer i = 256^{t-1} * i_1 + ... + 256 * i_{t-1} + i_t (the interpretation of the integer as a byte string in big-endian format);

str_t the byte string str_t(i) = (i_1, ... , i_t) in B_t corresponding to an integer i = 256^{t-1} * i_t + ... + 256 * i_2 + i_1 (the interpretation of the integer as a byte string in little-endian format);

k the length of the block cipher key in bytes;

n the length of the block cipher block in bytes;

IVlen the length of the initialization vector in bytes;

S the length of the authentication tag in bytes;

E_i the elliptic curve indicated by client in "supported_groups" extension;

O_i the zero point of the elliptic curve E_i;

m_i the order of group of points belonging to the elliptic curve E_i;

q_i the cyclic subgroup order of group of points belonging to the elliptic curve E_i;

h_i the cyclic subgroup cofactor which is equal to m_i / q_i;

Q_sign the public key stored in endpoint's certificate;

d_sign the private key that corresponds to the Q_sign key;

P_i the point of the elliptic curve E_i of the order q_i;

(d_C^i, Q_C^i) the client's ephemeral key pair which consists of the private key and the public key corresponding to the elliptic curve E_i;

(d_S^i, Q_S^i)
the server's ephemeral key pair which consists of the private key and the public key corresponding to the elliptic curve E_i.

4. Cipher Suite Definition

The cipher suite value is used to indicate a record protection algorithm and a hash algorithm which an endpoint supports (see Section 4.1.2 of [RFC8446]).

This section defines the following four TLS13_GOST cipher suites that can be used to support Russian cryptographic algorithms:

```
CipherSuite TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_L = {0xC1, 0x03};
CipherSuite TLS_GOSTR341112_256_WITH_MAGMA_MGM_L = {0xC1, 0x04};
CipherSuite TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_S = {0xC1, 0x05};
CipherSuite TLS_GOSTR341112_256_WITH_MAGMA_MGM_S = {0xC1, 0x06};
```

Each cipher suite specifies a pair of the record protection algorithm (see Section 4.1) and the hash algorithm (Section 4.2).

4.1. Record Protection Algorithm

In accordance with Section 5.2 of [RFC8446] the record protection algorithm translates a TLSPlaintext structure into a TLSCiphertext structure. If TLS13_GOST cipher suite is negotiated, the encrypted_record field of the TLSCiphertext structure MUST be set to the AEADEncrypted value computed as follows:

AEADEncrypted = AEAD-Encrypt(sender_record_write_key, nonce, additional_data, plaintext),

where

- the AEAD-Encrypt function is defined in Section 4.1.1;
- the sender_record_write_key is derived from the sender_write_key (see Section 7.3 of [RFC8446]) using TLSTREE function defined in Section 4.1.2 and sequence number seqnum as follows:

  sender_record_write_key = TLSTREE(sender_write_key, seqnum);

- the nonce value is derived from the record sequence number seqnum and the sender_write_iv value (see Section 7.3 of [RFC8446]) in accordance with Section 5.3 of [RFC8446];

*the additional_data value is the record header that is generated in accordance with Section 5.2 of [RFC8446];

*the plaintext value is the TLSInnerPlaintext structure encoded in accordance with Section 5.2 of [RFC8446].

Note1: The AEAD-Encrypt function is exactly the same as the AEAD-Encrypt function defined in [RFC8446] except the key (the first argument) is calculated from the sender_write_key and sequence number seqnum for each message separately to support external re-keying approach according to [RFC8645].

Note2: The record sequence number is the value in the range 0-SNMAX, where the SNMAX value is defined in Section 4.1.3. The SNMAX parameter is specified by the particular TLS13_GOST cipher suite to limit the amount of data that can be encrypted under the same traffic key material (sender_write_key, sender_write_iv).

The record deprotection algorithm reverses the process of the record protection. In order to decrypt and verify the protected record with sequence number seqnum the algorithm takes as input the sender_record_write_key is derived from the sender_write_key, nonce, additional_data and the AEADEncrypted value and outputs the res value which is either the plaintext or an error indicating that the decryption failed. If TLS13_GOST cipher suite is negotiated, the res value MUST be computed as follows:

```
res = AEAD-Decrypt(sender_record_write_key, nonce,
additional_data, AEADEncrypted),
```

where the AEAD-Decrypt function is defined in Section 4.1.1.

Note: The AEAD-Decrypt function is exactly the same as the AEAD-Decrypt function defined in [RFC8446] except the key (the first argument) is calculated from the sender_write_key and sequence number seqnum for each message separately to support external re-keying approach according to [RFC8645].

4.1.1. AEAD Algorithm

The AEAD-Encrypt and AEAD-Decrypt functions are defined as follows.

```
+--------------------------------------------------------------------+
| AEAD-Encrypt(K, nonce, A, P)                                       |
|--------------------------------------------------------------------|
| Input:                                                             |
| - encryption key K in B_k,                                         |
| - unique vector nonce in B_IVlen,                                  |
| - additional authenticated data A in B_r, r >= 0,                  |
| - plaintext P in B_t, t >= 0.                                      |
| Output:                                                            |
| - ciphertext C in B_{|P|},                                         |
| - authentication tag T in B_S.                                     |
|--------------------------------------------------------------------|
| 1. MGMnonce = STR_1(nonce[1] & 0x7f) | nonce[2..IVlen];            |
| 2. (MGMnonce, A, C, T) = MGM-Encrypt(K, MGMnonce, A, P);           |
| 3. Return C | T.                                                   |
+--------------------------------------------------------------------+

+--------------------------------------------------------------------+
| AEAD-Decrypt(K, nonce, A, C | T)                                   |
|--------------------------------------------------------------------|
| Input:                                                             |
| - encryption key K in B_k,                                         |
| - unique vector nonce in B_IVlen,                                  |
| - additional authenticated data A in B_r, r >= 0,                  |
| - ciphertext C in B_t, t >= 0,                                     |
| - authentication tag T in B_S.                                     |
| Output:                                                            |
| - plaintext P in B_{|C|} or FAIL.                                  |
|--------------------------------------------------------------------|
| 1. MGMnonce = STR_1(nonce[1] & 0x7f) | nonce[2..IVlen];            |
| 2. res' = MGM-Decrypt(K, MGMnonce, A, C, T);                       |
| 3. IF res' = FAIL then return FAIL;                                |
| 4. IF res' = (A, P) then return P.                                 |
+--------------------------------------------------------------------+
```

where

*MGM-Encrypt and MGM-Decrypt functions are defined in [RFC9058]. The size of the authentication tag T is equal to n bytes (S = n). The size of the nonce parameter is equal to n bytes (IVlen = n).

The cipher suites TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_L and TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_S MUST use Kuznyechik [RFC7801] as a base block cipher for the AEAD algorithm. The block length n is 16 bytes (n = 16) and the key length k is 32 bytes (k = 32).

The cipher suites TLS_GOSTR341112_256_WITH_MAGMA_MGM_L and TLS_GOSTR341112_256_WITH_MAGMA_MGM_S MUST use Magma [RFC8891] as a base block cipher for the AEAD algorithm. The block length n is 8 bytes (n = 8) and the key length k is 32 bytes (k = 32).

4.1.2. TLSTREE

The TLS13_GOST cipher suites use the TLSTREE function for the external re-keying approach (see [RFC8645]). The TLSTREE function is defined as follows:

TLSTREE(K_root, i) = KDF_3(KDF_2(KDF_1(K_root, STR_8(i & C_1)), STR_8(i & C_2)), STR_8(i & C_3)),

where

*K_root in B_32;

*i in {0, 1, ... , 2^64 - 1};

*KDF_j(K, D), j = 1, 2, 3, is the key derivation function defined as follows:

KDF_1(K, D) = KDF_GOSTR3411_2012_256(K, "level1", D),

KDF_2(K, D) = KDF_GOSTR3411_2012_256(K, "level2", D),

KDF_3(K, D) = KDF_GOSTR3411_2012_256(K, "level3", D),

where the KDF_GOSTR3411_2012_256 function is defined in [RFC7836], K in B_32, D in B_8.

*C_1, C_2, C_3 are constants defined by the particular cipher suite as follows:

CipherSuites	C_1, C_2, C_3
TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_L	C_1=0xf800000000000000 C_2=0xfffffff000000000 C_3=0xfffffffffffe000
TLS_GOSTR341112_256_WITH_MAGMA_MGM_L	C_1=0xffe0000000000000 C_2=0xffffffffc0000000 C_3=0xffffffffffffff80
TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_S	C_1=0xfffffffe0000000 C_2=0xffffffffffff0000 C_3=0xfffffffffffffff8
TLS_GOSTR341112_256_WITH_MAGMA_MGM_S	C_1=0xfffffffffc000000 C_2=0xffffffffffffe000 C_3=0xffffffffffffffff

Table 1

4.1.3. SNMAX parameter

The SNMAX parameter is the maximum number of records encrypted under the same traffic key material (sender_write_key and sender_write_iv) and is defined by the particular cipher suite as follows:

CipherSuites	SNMAX
TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_L	SNMAX = 2^64 - 1
TLS_GOSTR341112_256_WITH_MAGMA_MGM_L	SNMAX = 2^64 - 1
TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_S	SNMAX = 2^42 - 1
TLS_GOSTR341112_256_WITH_MAGMA_MGM_S	SNMAX = 2^39 - 1

Table 2

4.2. Hash Algorithm

The Hash algorithm is used for key derivation process (see Section 7.1 of [RFC8446]), Finished message calculation (see Section 4.4.4 of [RFC8446]), Transcript-Hash function computation (see Section 4.4.1 of [RFC8446]), PSK binder value calculation (see Section

4.2.11.2 of [RFC8446]), external re-keying approach (see Section 4.1.2) and other purposes.

In case of negotiating a TLS13_GOST cipher suite the Hash algorithm MUST be the GOST R 34.11-2012 [RFC6986] hash algorithm with 32-byte (256-bit) hash value.

5. Signature Scheme Definition

The signature scheme value is used to indicate a single signature algorithm and a curve that can be used in digital signature (see Section 4.2.3 of [RFC8446]).

This section defines the following seven TLS13_GOST signature schemes that can be used to support Russian cryptographic algorithms:

```
enum {
    gostr34102012_256a(0x0709),
    gostr34102012_256b(0x070A),
    gostr34102012_256c(0x070B),
    gostr34102012_256d(0x070C),
    gostr34102012_512a(0x070D),
    gostr34102012_512b(0x070E),
    gostr34102012_512c(0x070F)
} SignatureScheme;
```

One of the TLS13_GOST signature schemes listed above SHOULD be used with the TLS13_GOST profile.

Each signature scheme specifies a pair of the signature algorithm (see Section 5.1) and the elliptic curve (see Section 5.2). The procedure to calculate the signature value using TLS13_GOST signature schemes is defined in Section 5.3.

5.1. Signature Algorithm

Signature algorithms corresponding to the TLS13_GOST signature schemes are defined as follows:

SignatureScheme Value	Signature Algorithm	References
gostr34102012_256a	GOST R 34.10-2012 , 32-byte key length	RFC 7091
gostr34102012_256b	GOST R 34.10-2012 , 32-byte key length	RFC 7091
gostr34102012_256c	GOST R 34.10-2012 , 32-byte key length	RFC 7091
gostr34102012_256d	GOST R 34.10-2012 , 32-byte key length	RFC 7091
gostr34102012_512a	GOST R 34.10-2012 , 64-byte key length	RFC 7091
gostr34102012_512b	GOST R 34.10-2012 , 64-byte key length	RFC 7091
gostr34102012_512c	GOST R 34.10-2012 , 64-byte key length	RFC 7091

Table 3

5.2. Elliptic Curve

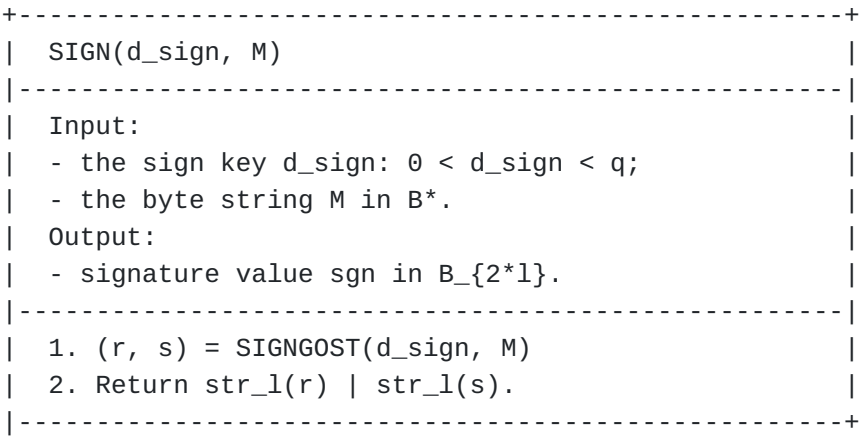
Elliptic curves corresponding to the TLS13_GOST signature schemes are defined as follows:

SignatureScheme Value	Curve Identifier Value	References
gostr34102012_256a	id-tc26-gost-3410-2012-256-paramSetA	RFC 7836
gostr34102012_256b	id-GostR3410-2001-CryptoPro-A-ParamSet	RFC 4357
gostr34102012_256c	id-GostR3410-2001-CryptoPro-B-ParamSet	RFC 4357
gostr34102012_256d	id-GostR3410-2001-CryptoPro-C-ParamSet	RFC 4357
gostr34102012_512a	id-tc26-gost-3410-12-512-paramSetA	RFC 7836
gostr34102012_512b	id-tc26-gost-3410-12-512-paramSetB	RFC 7836
gostr34102012_512c	id-tc26-gost-3410-2012-512-paramSetC	RFC 7836

Table 4

5.3. SIGN function

If TLS13_GOST signature scheme is used, the signature value in CertificateVerify message (see Section 6.3.4) MUST be calculated using the SIGN function defined as follows:



where

*q is the subgroup order of group of points of the elliptic curve;

*l is defined as follows:

-l = 32 for gostr34102012_256a, gostr34102012_256b, gostr34102012_256c and gostr34102012_256d signature schemes;

-l = 64 for gostr34102012_512a, gostr34102012_512b and gostr34102012_512c signature schemes;

*SIGNGOST is an algorithm which takes as an input private key d_sign and message M and returns a pair of integers (r, s) calculated during signature generation process in accordance with the GOST R 34.10-2012 signature algorithm (see Section 6.1 of [RFC7091]).

Note: The signature value sgn is the concatenation of two strings that are byte representations of r and s values in the little-endian format.

6. Key Exchange and Authentication

Key exchange and authentication process in case of using TLS13_GOST profile is defined in Section 6.1, Section 6.2 and Section 6.3.

6.1. Key Exchange

TLS13_GOST profile support three basic key exchange modes which are defined in [RFC8446]: ECDHE, PSK-only and PSK-with-ECDHE.

Note: In accordance with [RFC8446] TLS 1.3 also supports key exchange modes based on Diffie-Hellman protocol over finite fields. However, TLS13_GOST cipher suites MUST use only modes based on Diffie-Hellman protocol over elliptic curves.

In accordance with [RFC8446] PSKs can be divided into two types:

- internal PSKs which can be established during the previous connection;
- external PSKs which can be established out of band.

If TLS13_GOST profile is used, PSK-only key exchange mode SHOULD be established only via the internal PSKs, and external PSKs SHOULD be used only in PSK-with-ECDHE mode (see more in Section 9).

If TLS13_GOST profile is used and ECDHE or PSK-with-ECDHE key exchange mode is used the ECDHE shared secret value should be calculated in accordance with Section 6.1.1 on the basis of one of the elliptic curves defined in Section 6.1.2.

6.1.1. ECDHE Shared Secret Calculation

If TLS13_GOST profile is used, ECDHE shared secret value should be calculated in accordance with Section 6.1.1.1 and Section 6.1.1.2. The public ephemeral keys used to obtain ECDHE shared secret value should be represented in format described in Section 6.1.1.3.

6.1.1.1. ECDHE Shared Secret Calculation on Client Side

The client calculates ECDHE shared secret value in accordance with the following steps:

1. Chooses from all supported curves E_1, ..., E_R the set of curves E_{i_1}, ..., E_{i_r}, 1 <= i_1 <= i_r <= R, where

   - r >= 1 in the case of the first ClientHello message;
   - r = 1 in the case of responding to HelloRetryRequest message, E_{i_1} corresponds to the curve indicated in the "key_share" extension in the HelloRetryRequest message.

2. Generates ephemeral key pairs (d_C^{i_1}, Q_C^{i_1}), ..., (d_C^{i_r}, Q_C^{i_r}) corresponding to the curves E_{i_1}, ..., E_{i_r}, where for each i in {i_1, ..., i_r}:

  *d_C^i is chosen from {1, ..., q_i - 1} at random;

  *Q_C^i = d_C^i * P_i.

3. Sends ClientHello message specified in accordance with Section 4.1.2 of [RFC8446] and Section 6.3.1, which contains:

  *"key_share" extension with public ephemeral keys Q_C^{i_1}, ..., Q_C^{i_r} generated in accordance with Section 4.2.8 of [RFC8446];

  *"supported_groups" extension with supported curves E_1, ..., E_R generated in accordance with Section 4.2.7 of [RFC8446].

Note: Client MAY send an empty "key_share" extension in the first ClientHello in order to request group selection from the server in the HelloRetryRequest message and to generate ephemeral key for the selected group only. The ECDHE value may be calculated without sending HelloRetryRequest, if the "key_share" extension in the first ClientHello message consists the value corresponded to the curve that will be selected by the server.

4. In case of receiving HelloRetryRequest message client MUST return to step 1 and correct parameters in accordance with Section 4.1.2 of [RFC8446]. In case of receiving ServerHello message client proceeds to the next step. In other cases client MUST terminate the connection with "unexpected_message" alert.

5. Extracts curve E_res and ephemeral key Q_S^res, res in {1, ..., R}, from ServerHello message and checks whether the Q_S^res belongs to E_res. If this check fails, the client MUST abort the handshake with "handshake_failure" alert.

6. Generates Q^ECDHE:

   Q^ECDHE = (X^ECDHE, Y^ECDHE) = (h_res * d_C^res) * Q_S^res.

7. Client MUST check whether the computed shared secret Q^ECDHE is not equal to the zero point O_res. If this check fails, the client MUST abort the handshake with "handshake_failure" alert.

8. Shared secret value ECDHE is the byte representation of the coordinate X^ECDHE of point Q^ECDHE in the little-endian format:

   ECDHE = str_{coordinate_length}(X^ECDHE),

where the coordinate_length value is defined by the particular elliptic curve (see Section 6.1.2).

6.1.1.2. ECDHE Shared Secret Calculation on Server Side

Upon receiving the ClientHello message, the server calculates ECDHE shared secret value in accordance with the following steps:

1. Chooses the curve E_res, res in {1, ..., R}, from the list of the curves E_1, ..., E_R indicated in "supported_groups" extension in ClientHello message and the corresponding public ephemeral key value Q_C^res from the list Q_C^{i_1}, ..., Q_C^{i_r}, 1 <= i_1 <= i_r <= R, indicated in "key_share" extension. If no corresponding public ephemeral key value is found (res in {1, ..., R}\{i_1, ..., i_r}), server MUST send HelloRetryRequest message with "key_share" extension indicating the selected elliptic curve E_res and wait for the new ClientHello message.

2. Checks whether Q_C^res belongs to E_res. If this check fails, the server MUST abort the handshake with "handshake_failure" alert.

3. Generates ephemeral key pair (d_S^res, Q_S^res) corresponding to E_res:

   *d_S^res is chosen from {1, ..., q_res - 1} at random;

   *Q_S^res = d_S^res * P_res.

4. Sends ServerHello message generated in accordance with Section 4.1.3 of [RFC8446] and Section 6.3.1 with "key_share" extension which contains public ephemeral key value Q_S^res corresponding to E_res.

5. Generates Q^ECDHE:

   Q^ECDHE = (X^ECDHE, Y^ECDHE) = (h_res * d_S^res) * Q_C^res.

6. Server MUST check whether the computed shared secret Q^ECDHE is not equal to the zero point O_res. If this check fails, the server MUST abort the handshake with "handshake_failure" alert.

7. Shared secret value ECDHE is the byte representation of the coordinate X^ECDHE of point Q^ECDHE in the little-endian format:

   ECDHE = str_{coordinate_length}(X^ECDHE),

where the coordinate_length value is defined by the particular elliptic curve (see Section 6.1.2).

6.1.1.3. Public ephemeral key representation

This section defines the representation format of the public ephemeral keys generated during ECDHE shared secret calculation (see Section 6.1.1.1 and Section 6.1.1.2).

If TLS13_GOST profile is used and ECDHE or PSK-with-ECDHE key exchange mode is used, the public ephemeral key Q indicated in the KeyShareEntry.key_exchange field MUST contain the data defined by the following structure:

```
struct {
    opaque X[coordinate_length];
    opaque Y[coordinate_length];
} PlainPointRepresentation;
```

where X and Y, respectively, contain the byte representations of the x and the y values of point Q (Q = (x, y)) in the little-endian format and are specified as follows:

X = str_{coordinate_length}(x);

Y = str_{coordinate_length}(y).

The coordinate_length value is defined by the particular elliptic curve (see Section 6.1.2).

6.1.2. Values for the TLS Supported Groups Registry

The "supported_groups" extension is used to indicate the set of the elliptic curves supported by an endpoint and is defined in Section 4.2.7 [RFC8446]. This extension is always contained in ClientHello message and optionally in EncryptedExtensions message.

This section defines the following seven elliptic curves that can be used to support Russian cryptographic algorithms:

```
enum {
    GC256A(0x22), GC256B(0x23), GC256C(0x24), GC256D(0x25),
    GC512A(0x26), GC512B(0x27), GC512C(0x28)
} NamedGroup;
```

If TLS13_GOST profile is used and ECDHE or PSK-with-ECDHE key exchange mode is established, one of the elliptic curves listed above SHOULD be used.

Each curve corresponds to the particular identifier and specifies the value of coordinate_length parameter (see "cl" column) as follows:

Description	Curve Identifier Value	cl	Reference
GC256A	id-tc26-gost-3410-2012-256-paramSetA	32	RFC 7836
GC256B	id-GostR3410-2001-CryptoPro-A-ParamSet	32	RFC 4357
GC256C	id-GostR3410-2001-CryptoPro-B-ParamSet	32	RFC 4357
GC256D	id-GostR3410-2001-CryptoPro-C-ParamSet	32	RFC 4357
GC512A	id-tc26-gost-3410-12-512-paramSetA	64	RFC 7836
GC512B	id-tc26-gost-3410-12-512-paramSetB	64	RFC 7836
GC512C	id-tc26-gost-3410-2012-512-paramSetC	64	RFC 7836

Table 5

Note: The identifier values and the corresponding elliptic curves are the same as in [DraftGostTLS12].

6.2. Authentication

In accordance with [RFC8446] authentication can happen via signature with certificate or via symmetric pre-shared key (PSK). The server side of the channel is always authenticated; the client side is optionally authenticated.

PSK-based authentication happens as a side effect of key exchange. If TLS13_GOST profile is used, external PSKs SHOULD be combined only with the mutual authentication (see more in Section 9).

Certificate-based authentication happens via Authentication messages and optional CertificateRequest message (sent if client authentication is required). In case of using TLS13_GOST profile the signature schemes used for certificate-based authentication are defined in Section 5 and the Authentication messages are specified in Section 6.3.3 and Section 6.3.4. The CertificateRequest message is specified in Section 6.3.2.

6.3. Handshake Messages

The TLS13_GOST profile specify the ClientHello, ServerHello, CertificateRequest, Certificate and CertificateVerify handshake messages that are described in further detail below.

6.3.1. Hello Messages

The ClientHello message is sent when a client first connects to a server or responds to a HelloRetryRequest message and is specified in accordance with Section 4.1.2 of [RFC8446].

In case of using the TLS13_GOST profile, the ClientHello message MUST meet the following requirements.

- The ClientHello.cipher_suites field MUST contain the values defined in Section 4.

- If server authentication via a certificate is required, the extension_data field of the "signature_algorithms" extension MUST contain the values defined in Section 5, which correspond to the GOST R 34.10-2012 signature algorithm.

- If server authentication via a certificate is required and the client uses optional "signature_algorithms_cert" extension, the extension_data field of this extension SHOULD contain the values defined in Section 5, which correspond to the GOST R 34.10-2012 signature algorithm.

- If client wants to establish TLS 1.3 connection using ECDHE shared secret value, the extension_data field of the "supported_groups" extension MUST contain the elliptic curve identifiers defined in Section 6.1.2.

The ServerHello message is sent by the server in response to a ClientHello message to negotiate an acceptable set of handshake parameters based on the ClientHello and is specified in accordance with Section 4.1.3 of [RFC8446].

In case of using the TLS13_GOST profile, the ServerHello message MUST meet the following requirements.

- The ServerHello.cipher_suite field MUST contain one of the values defined in Section 4.

- If server decides to establish TLS 1.3 connection using ECDHE shared secret value, the extension_data field of the "key_share"

extension MUST contain the elliptic curve identifier and the public ephemeral key that satisfy the following conditions.

- The elliptic curve identifier corresponds to a value that was provided in the "supported_groups" and the "key_share" extensions in the ClientHello message.

- The elliptic curve identifier is one of the values defined in Section 6.1.2.

- The public ephemeral key corresponds to the elliptic curve specified by the KeyShareEntry.group identifier.

6.3.2. CertificateRequest

This message is sent when server requests client authentication via a certificate and is specified in accordance with Section 4.3.2 of [RFC8446].

In case of using the TLS13_GOST profile, the CertificateRequest message MUST meet the following requirements.

* The extension_data field of the "signature_algorithms" extension MUST contain only the values defined in Section 5.

* If server uses optional "signature_algorithms_cert" extension, the extension_data field of this extension SHOULD contain only the values defined in Section 5.

6.3.3. Certificate

This message is sent to convey the endpoint's certificate chain to the peer and is specified in accordance with Section 4.4.2 of [RFC8446].

In case of using the TLS13_GOST profile, the Certificate message MUST meet the following requirements.

* Each endpoint's certificate provided to the peer MUST be signed using the algorithm which corresponds to a signature scheme indicated by the peer in its "signature_algoritms_cert" extension, if present (or in the "signature_algorithms" extension, otherwise).

* The signature algorithm used for signing certificates SHOULD correspond to the one of the signature schemes defined in Section 5.

6.3.4. CertificateVerify

This message is sent to provide explicit proof that an endpoint possesses the private key corresponding to the public key indicated in its certificate and is specified in accordance with Section 4.4.3 of [RFC8446].

In case of using the TLS13_GOST profile, the CertificateVerify message MUST meet the following requirements.

* The CertificateVerify.algorithm field MUST contain the signature scheme identifier which corresponds to the value indicated in the peer's "signature_algorithms" extension and which is one of the values defined in Section 5.

* The CertificateVerify.signature field contains the sgn value, which is computed as follows:

  sgn = SIGN(d_sign, M),

* where

  - the SIGN function is defined in Section 5.3,

  - d_sign is the sender long-term private key that corresponds to the sender long-term public key Q_sign from the sender's certificate,

  - the message M is defined in accordance with Section 4.4.3 of [RFC8446].

7. IANA Considerations

IANA has added numbers {0xC1, 0x03}, {0xC1, 0x04}, {0xC1, 0x05} and {0xC1, 0x06} with the names
TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_L,
TLS_GOSTR341112_256_WITH_MAGMA_MGM_L,
TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_S,
TLS_GOSTR341112_256_WITH_MAGMA_MGM_S to the "TLS Cipher Suites" registry with this document as reference, as shown below.

Value	Description	DTLS-OK	Reference
0xC1, 0x03	TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_L	N	this RFC
0xC1, 0x04	TLS_GOSTR341112_256_WITH_MAGMA_MGM_L	N	this RFC
0xC1, 0x05	TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_S	N	this RFC
0xC1, 0x06	TLS_GOSTR341112_256_WITH_MAGMA_MGM_S	N	this RFC

Table 6

IANA has added numbers 0x0709, 0x070A, 0x070B, 0x070C, 0x070D, 0x070E and 0x070F with the names gostr34102012_256a, gostr34102012_256b, gostr34102012_256c, gostr34102012_256d, gostr34102012_512a, gostr34102012_512b, gostr34102012_512c to the "TLS SignatureScheme" registry, as shown below.

Value	Description	DTLS-OK	Reference
0x0709	gostr34102012_256a	N	this RFC
0x070A	gostr34102012_256b	N	this RFC
0x070B	gostr34102012_256c	N	this RFC
0x070C	gostr34102012_256d	N	this RFC
0x070D	gostr34102012_512a	N	this RFC
0x070E	gostr34102012_512b	N	this RFC
0x070F	gostr34102012_512c	N	this RFC

Table 7

8. Historical considerations

Due to historical reasons in addition to the curve identifier values listed in Table 5 there exist some additional identifier values that correspond to the signature schemes as follows.

Description	Curve Identifier Value
gostr34102012_256b	id-GostR3410_2001-CryptoPro-XchA-ParamSet
	id-tc26-gost-3410-2012-256-paramSetB
gostr34102012_256c	id-tc26-gost-3410-2012-256-paramSetC
gostr34102012_256d	id-GostR3410-2001-CryptoPro-XchB-ParamSet
	id-tc26-gost-3410-2012-256-paramSetD

Table 8

Client should be prepared to handle any of them correctly if corresponding signature scheme is included in the "signature_algorithms" or "signature_algorithms_cert" extensions.

9. Security Considerations

In order to create an effective implementation client and server SHOULD follow the rules below.

1. While using TLSTREE algorithm function KDF_j, j = 1, 2, 3, SHOULD be invoked only if the record sequence number seqnum reaches such a value that

   seqnum & C_j != (seqnum - 1) & C_j.

Otherwise the previous value should be used.

2. For each pre-shared key value PSK the binder_key value should be computed only once within all connections where ClientHello message contains a "pre_shared_key" extension indicating this PSK value.

In order to ensure the secure TLS 1.3 connection client and server SHOULD fulfil the following requirements.

1. An internal PSK value is NOT RECOMMENDED to be used to establish more than one TLS 1.3 connection.

2. 0-RTT data SHOULD NOT be sent during TLS 1.3 connection. The reasons for this restriction are that the 0-RTT data is not forward

secret and is not resistant to replay attacks (see more in Section 2.3 of [RFC8446]).

3. If client authentication is required, server SHOULD NOT send Application Data, NewSessionTicket and KeyUpdate messages prior to receiving the client's Authentication messages since any data sent at that point is being sent to an unauthenticated peer.

4. External PSKs SHOULD be used only in PSK-with-ECDHE mode. In case of using external PSK in PSK-only mode the attack described in [Selfie] is possible which leads to the situation when client establishes connection to itself. One of the mitigations proposed in [Selfie] is to use certificates, however, in that case, an impersonation attack as in [AASS19] occurs. If the connections are established with additional usage of key_share extension (in PSK-with-ECDHE mode), then the adversary which just echoes messages cannot reveal the traffic key material (as long as the used group is secure).

5. In case of using external PSK, the mutual authentication MUST be provided by the external PSK distribution mechanism between the endpoints which guarantees that the derived external PSK is unknown to anyone but the endpoints. In addition, the endpoint roles (i.e. client and server) MUST be fixed during this mechanism and each role can match only to one endpoint during the whole external PSK lifetime.

10. References

10.1. Normative References

[DraftGostTLS12] Smyshlyaev, S., Belyavsky, D., Saarinen, M., and E. Alekseev, "GOST Cipher Suites for Transport Layer Security (TLS) Protocol Version 1.2", 2021, <https://tools.ietf.org/html/draft-smyshlyaev-tls12-gost-suites-18>.

[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>.

[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>.

[RFC7801] Dolmatov, V., Ed., "GOST R 34.12-2015: Block Cipher "Kuznyechik"", RFC 7801, DOI 10.17487/RFC7801, March 2016, <https://www.rfc-editor.org/info/rfc7801>.

[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>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/info/rfc8446>.

[RFC8645] Smyshlyaev, S., Ed., "Re-keying Mechanisms for Symmetric Keys", RFC 8645, DOI 10.17487/RFC8645, August 2019, <https://www.rfc-editor.org/info/rfc8645>.

[RFC8891] Dolmatov, V., Ed. and D. Baryshkov, "GOST R 34.12-2015: Block Cipher "Magma"", RFC 8891, DOI 10.17487/RFC8891, September 2020, <https://www.rfc-editor.org/info/rfc8891>.

[RFC9058] Smyshlyaev, S., Ed., Nozdrunov, V., Shishkin, V., and E. Griboedova, "Multilinear Galois Mode (MGM)", RFC 9058, DOI 10.17487/RFC9058, June 2021, <https://www.rfc-editor.org/info/rfc9058>.

10.2. Informative References

[AASS19] Akhmetzyanova, L., Alekseev, E., Smyshlyaeva, E., and A. Sokolov, "Continuing to reflect on TLS 1.3 with external PSK", Cryptology ePrint Archive Report 2019/421, April 2019, <https://eprint.iacr.org/2019/421.pdf>.

[Selfie] Drucker, N. and S. Gueron, "Selfie: reflections on TLS 1.3 with PSK", Cryptology ePrint Archive Report 2019/347, April 2019, <https://eprint.iacr.org/2019/347.pdf>.

Appendix A. Test Examples

A.1. Test case

Test examples are given for the following order of using the TLS_GOST profile for the TLS 1.3 protocol.

1. Full TLS Handshake is used.

2. ECDHE key exchange mode is used. The elliptic curve GC512C is used for ECDHE shared secret calculation.

3. The server side only authentication is used. The signature scheme gost34102012_256b (0x070A) is used.

4. TLS_GOSTR341112_256_WITH_KUZNYECHIK_MGM_S cipher suite is negotiated.

5. Application Data is sent prior to sending the Finished message by client.

6. NewSessionTicket is sent after establishing a secure connection.

7. Four records of the Application data type are sent during the operation of the Record protocol.

8. Alert protocol is used for closure of the connection.

A.2. Test examples

```
---------------------------Client---------------------------

ClientHello message:
msg_type:               01
length:                 0000DE
body:
  legacy_version:       0303
  random:               0303030303030303030303030303030303
                        0303030303030303030303030303030303
  legasy_session_id:
    length:             00
    vector:             --
  cipher_suites:
    length:             0002
    vector:
      CipherSuite:      C105
  compression_methods:
    length:             01
    vector:
      CompressionMethod: 00
  extensions:
    length:             00B3
    vector:
      Extension: /* supported_groups */
        extension_type:  000A
        extension_data:
          length:       0004
          vector:
            named_group_list:
              length:   0002
              vector:
               /* GC512C */
                        0028
      Extension: /* signature_algorithms */
        extension_type:  000D
        extension_data:
          length:       0010
          vector:
            supported_signature_algorithms:
              length:   000E
              vector:
               /* gost34102012256a */
                        0709
               /* gost34102012256b */
                        070A
               /* gost34102012256c */
                        070B
```

```
              /* gost34102012256d */
                       070C
              /* gost34102012512a */
                       070D
              /* gost34102012512b */
                       070E
              /* gost34102012512c */
                       070F
        Extension: /* supported_versions */
          extension_type:  002B
          extension_data:
            length:        0003
            vector:
              versions:
                length:    02
                vector:
                       0304
        Extension: /* psk_key_exchange_modes */
          extension_type:  002D
          extension_data:
            length:        0002
            vector:
              ke_modes:
                length:    01
                vector:
                 /* psk_ke */
                       00
        Extension: /* key_share */
          extension_type:  0033
          extension_data:
            length:        0086
            vector:
              length:      0084
              vector:
                group:     0028
                key_exchange:
                  length:  0080
                  vector:
                           05EEBDF3DDC1D2F5F3822433241284E7
                           7641487938EA88721F26203E9792B5CB
                           97EB70EF02E8F72B7491D4F2CFDC332A
                           DF7F1778E854A88DDC2113FEC527A151
                           71A04CB0C573793A7AEF9BBCA486B6B0
                           46B2149B46F4332903E5B7C438ADD05E
                           185EFBF45557475A8CCBF6ACED1A2EB4
                           16F916729D7CEF9CBD8334989304AFAE

0000:   01 00 00 DE 03 03 03 03 03 03 03 03 03 03 03 03
0010:   03 03 03 03 03 03 03 03 03 03 03 03 03 03 03 03
```

```
0020:   03 03 03 03 03 03 00 00 02 C1 05 01 00 00 B3 00
0030:   0A 00 04 00 02 00 28 00 0D 00 10 00 0E 07 09 07
0040:   0A 07 0B 07 0C 07 0D 07 0E 07 0F 00 2B 00 03 02
0050:   03 04 00 2D 00 02 01 00 00 33 00 86 00 84 00 28
0060:   00 80 05 EE BD F3 DD C1 D2 F5 F3 82 24 33 24 12
0070:   84 E7 76 41 48 79 38 EA 88 72 1F 26 20 3E 97 92
0080:   B5 CB 97 EB 70 EF 02 E8 F7 2B 74 91 D4 F2 CF DC
0090:   33 2A DF 7F 17 78 E8 54 A8 8D DC 21 13 FE C5 27
00A0:   A1 51 71 A0 4C B0 C5 73 79 3A 7A EF 9B BC A4 86
00B0:   B6 B0 46 B2 14 9B 46 F4 33 29 03 E5 B7 C4 38 AD
00C0:   D0 5E 18 5E FB F4 55 57 47 5A 8C CB F6 AC ED 1A
00D0:   2E B4 16 F9 16 72 9D 7C EF 9C BD 83 34 98 93 04
00E0:   AF AE

Record layer message:
type:                   16
legacy_record_version:  0303
length:                 00E2
fragment:               010000DE030303030303030303030303
                        030303030303030303030303030303
                        030303030303000002C105010000B300
                        0A00040002002800 0D0010000E070907
                        0A070B070C070D070E070F002B000302
                        0304002D00020100003300 8600840028
                        008005EEBDF3DDC1D2F5F38224332412
                        84E77641487938EA88721F26203E9792
                        B5CB97EB70EF02E8F72B7491D4F2CFDC
                        332ADF7F1778E854A88DDC2113FEC527
                        A15171A04CB0C573793A7AEF9BBCA486
                        B6B046B2149B46F4332903E5B7C438AD
                        D05E185EFBF45557475A8CCBF6ACED1A
                        2EB416F916729D7CEF9CBD8334989304
                        AFAE

00000:   16 03 01 00 E2 01 00 00 DE 03 03 03 03 03 03 03
00010:   03 03 03 03 03 03 03 03 03 03 03 03 03 03 03 03
00020:   03 03 03 03 03 03 03 03 03 03 03 00 00 02 C1 05
00030:   01 00 00 B3 00 0A 00 04 00 02 00 28 00 0D 00 10
00040:   00 0E 07 09 07 0A 07 0B 07 0C 07 0D 07 0E 07 0F
00050:   00 2B 00 03 02 03 04 00 2D 00 02 01 00 00 33 00
00060:   86 00 84 00 28 00 80 05 EE BD F3 DD C1 D2 F5 F3
00070:   82 24 33 24 12 84 E7 76 41 48 79 38 EA 88 72 1F
00080:   26 20 3E 97 92 B5 CB 97 EB 70 EF 02 E8 F7 2B 74
00090:   91 D4 F2 CF DC 33 2A DF 7F 17 78 E8 54 A8 8D DC
000A0:   21 13 FE C5 27 A1 51 71 A0 4C B0 C5 73 79 3A 7A
000B0:   EF 9B BC A4 86 B6 B0 46 B2 14 9B 46 F4 33 29 03
000D0:   E5 B7 C4 38 AD D0 5E 18 5E FB F4 55 57 47 5A 8C
000E0:   CB F6 AC ED 1A 2E B4 16 F9 16 72 9D 7C EF 9C BD
000F0:   83 34 98 93 04 AF AE
```

```
---------------------------Server---------------------------

ServerHello message:
msg_type:                02
length:                  0000B6
body:
  legacy_version:        0303
  random:                838383838383838383838383838383
                         838383838383838383838383838383

  legasy_session_id:
    length:              00
    vector:              --
  cipher_suite:
      CipherSuite:       C105
  compression_method:
      CompressionMethod: 00
  extensions:
    length:              008E
    vector:
      Extension: /* supported_versions */
        extension_type:  002B
        extension_data:
          length:        0002
          vector:
            selected_version:
                         0304
      Extension: /* key_share */
        extension_type:  0033
        extension_data:
          length:        0084
          vector:
            group:       0028
            key_exchange:
              length:    0080
              vector:
                         2F3C663FE74735A1C421160DF0F43266
                         185FD30B6E5D6E88FC4061FAEACAB338
                         B10A1BD20CB0B4EE757E74A0027D409F
                         E937F01633A1E3F9A5518DEFD0F89F9D
                         3D9F6CC651413DEC2C74366D83C47EE1
                         DE4E421F65CD1163E94EA0C2E19ED45D
                         35558B937D9BFDC5ECC2B2A21B4EC3D5
                         3B29579A8FD5E074811028FBCF17994F

00000:   02 00 00 B6 03 03 83 83 83 83 83 83 83 83 83 83
00010:   83 83 83 83 83 83 83 83 83 83 83 83 83 83 83 83
00020:   83 83 83 83 83 83 00 C1 05 00 00 8E 00 2B 00 02
```

```
00030:   03 04 00 33 00 84 00 28 00 80 2F 3C 66 3F E7 47
00040:   35 A1 C4 21 16 0D F0 F4 32 66 18 5F D3 0B 6E 5D
00050:   6E 88 FC 40 61 FA EA CA B3 38 B1 0A 1B D2 0C B0
00060:   B4 EE 75 7E 74 A0 02 7D 40 9F E9 37 F0 16 33 A1
00070:   E3 F9 A5 51 8D EF D0 F8 9F 9D 3D 9F 6C C6 51 41
00080:   3D EC 2C 74 36 6D 83 C4 7E E1 DE 4E 42 1F 65 CD
00090:   11 63 E9 4E A0 C2 E1 9E D4 5D 35 55 8B 93 7D 9B
000A0:   FD C5 EC C2 B2 A2 1B 4E C3 D5 3B 29 57 9A 8F D5
000B0:   E0 74 81 10 28 FB CF 17 99 4F

Record layer message:
type:                   16
legacy_record_version:  0303
length:                 00BA
fragment:               020000410303933EA21E49C31BC3A345
                        6165889684CAA5576CE7924A24F58113
                        808DBD9EF85610C3802A561550EC78D6
                        ED51AC2439D7E7C101000009FF010001
                        0000170000

00000:   16 03 03 00 BA 02 00 00 B6 03 03 83 83 83 83 83
00010:   83 83 83 83 83 83 83 83 83 83 83 83 83 83 83 83
00020:   83 83 83 83 83 83 83 83 83 83 00 C1 05 00 00
00030:   8E 00 2B 00 02 03 04 00 33 00 84 00 28 00 80 2F
00040:   3C 66 3F E7 47 35 A1 C4 21 16 0D F0 F4 32 66 18
00050:   5F D3 0B 6E 5D 6E 88 FC 40 61 FA EA CA B3 38 B1
00060:   0A 1B D2 0C B0 B4 EE 75 7E 74 A0 02 7D 40 9F E9
00070:   37 F0 16 33 A1 E3 F9 A5 51 8D EF D0 F8 9F 9D 3D
00080:   9F 6C C6 51 41 3D EC 2C 74 36 6D 83 C4 7E E1 DE
00090:   4E 42 1F 65 CD 11 63 E9 4E A0 C2 E1 9E D4 5D 35
000A0:   55 8B 93 7D 9B FD C5 EC C2 B2 A2 1B 4E C3 D5 3B
000B0:   29 57 9A 8F D5 E0 74 81 10 28 FB CF 17 99 4F

---------------------------Client---------------------------

d_C^res:
00000:   04 04 04 04 04 04 04 04 04 04 04 04 04 04 04 04
00010:   04 04 04 04 04 04 04 04 04 04 04 04 04 04 04 04
00020:   04 04 04 04 04 04 04 04 04 04 04 04 04 04 04 04
00030:   04 04 04 04 04 04 04 04 04 04 04 04 04 04 04 04

Q_S^res:
00000:   2F 3C 66 3F E7 47 35 A1 C4 21 16 0D F0 F4 32 66
00010:   18 5F D3 0B 6E 5D 6E 88 FC 40 61 FA EA CA B3 38
00020:   B1 0A 1B D2 0C B0 B4 EE 75 7E 74 A0 02 7D 40 9F
00030:   E9 37 F0 16 33 A1 E3 F9 A5 51 8D EF D0 F8 9F 9D
00040:   3D 9F 6C C6 51 41 3D EC 2C 74 36 6D 83 C4 7E E1
00050:   DE 4E 42 1F 65 CD 11 63 E9 4E A0 C2 E1 9E D4 5D
00060:   35 55 8B 93 7D 9B FD C5 EC C2 B2 A2 1B 4E C3 D5
```

```
00070:   3B 29 57 9A 8F D5 E0 74 81 10 28 FB CF 17 99 4F

ECDHE:
00000:   4D E6 0D 21 EA 8F B9 22 0D 14 64 23 B4 90 DA 40
00010:   CC EB C4 3B C5 89 DB 79 B8 31 A4 7D 6B 06 30 07
00020:   DD 03 40 5A 1B 79 76 B6 23 DC AA 69 B0 11 AE 10
00030:   6E 7E 41 74 38 5F 86 26 E1 21 B5 99 43 63 C9 9F

---------------------------Server---------------------------

d_S^res:
00000:   AA 3C A4 F4 A5 0A C0 5B 37 42 B1 35 B5 30 A9 F2
00010:   2A E4 F5 E1 85 30 1D EC 83 2E 77 BA 3B CD 6A F1
00020:   84 84 84 84 84 84 84 84 84 84 84 84 84 84 84 84
00030:   84 84 84 84 84 84 84 84 84 84 84 84 84 84 84 04

Q_C^res:
00000:   05 EE BD F3 DD C1 D2 F5 F3 82 24 33 24 12 84 E7
00010:   76 41 48 79 38 EA 88 72 1F 26 20 3E 97 92 B5 CB
00020:   97 EB 70 EF 02 E8 F7 2B 74 91 D4 F2 CF DC 33 2A
00030:   DF 7F 17 78 E8 54 A8 8D DC 21 13 FE C5 27 A1 51
00040:   71 A0 4C B0 C5 73 79 3A 7A EF 9B BC A4 86 B6 B0
00050:   46 B2 14 9B 46 F4 33 29 03 E5 B7 C4 38 AD D0 5E
00060:   18 5E FB F4 55 57 47 5A 8C CB F6 AC ED 1A 2E B4
00070:   16 F9 16 72 9D 7C EF 9C BD 83 34 98 93 04 AF AE

ECDHE:
00000:   4D E6 0D 21 EA 8F B9 22 0D 14 64 23 B4 90 DA 40
00010:   CC EB C4 3B C5 89 DB 79 B8 31 A4 7D 6B 06 30 07
00020:   DD 03 40 5A 1B 79 76 B6 23 DC AA 69 B0 11 AE 10
00030:   6E 7E 41 74 38 5F 86 26 E1 21 B5 99 43 63 C9 9F

---------------------------Server---------------------------

EncryptedExtensions message:
msg_type:               08
length:                 000002
body:
  extensions:
    length:             0000
    vector:             --

00000:   08 00 00 02 00 00

Record payload protection:

  EarlySecret = HKDF-Extract(Salt: 0^256, IKM: 0^256):
  00000:   FB DE FB E5 27 FE EA 66 5A AB 92 77 A2 16 3B 83
  00010:   43 08 4F D1 91 C4 60 66 26 0F AC 6F D1 43 6C 72
```

```
  Derived #0 = Derive-Secret(EarlySecret, "derived", "") =
  HKDF-Expand-Label(EarlySecret, "derived", "", 32):
  00000:   DB C3 C8 26 D8 77 A3 B7 D2 D2 45 3D BF DC 6C FB
  00010:   FB 11 51 B3 E8 4F 0C 8F 26 01 1D 8D 5B F3 ED F7

  HandshakeSecret = HKDF-Extract(Salt: Derived #0, IKM: ECDHE):
  00000:   44 24 5E 2C 43 32 D1 F7 8B 0F 8D 16 F4 03 EB 69
  00010:   ED 2A 40 53 84 7C DC 39 FA 8B 3D 29 74 F7 45 E7

  HM1 = (ClientHello, ServerHello)
  TH1 = Transcript-Hash(HM1):
  00000:   99 3B A7 22 12 4A F3 CB FD 47 71 E7 FA E3 2A C1
  00010:   D0 E9 27 8C F7 84 3F CB C6 20 E1 A0 08 5A 87 A1

  server_handshake_traffic_secret (SHTS):
    SHTS = Derive-Secret(HandshakeSecret, "s hs traffic", HM1) =
    HKDF-Expand-Label(HandshakeSecret, "s hs traffic", TH1, 32):
    00000:   70 A5 F2 46 3D F6 0D BA A2 36 8B 67 FD 45 AE FF
    00010:   7C 1A 0B A4 2D 8A BD 72 41 5E CD 1D 94 E9 EF 54

  server_write_key_hs = HKDF-Expand-Label(SHTS, "key", "", 32):
  00000:   E1 37 64 B5 4B 9E 1B 47 D4 33 98 D6 D2 16 DF 24
  00010:   C2 89 A3 96 AB 6C 5B 52 4B BB 9C 06 F3 9F EF 01

  server_write_iv_hs = HKDF-Expand-Label(SHTS, "iv", "", 16):
  00000:   69 69 FF AA A4 52 52 81 EE BB EB 4C BD 0B 64 0E

  server_record_write_key = TLSTREE(server_write_key_hs, 0):
  00000:   56 EE 18 13 72 72 49 C9 DC DF 35 13 78 7E DB 93
  00010:   DF 62 C6 1E E7 B1 26 C5 0F 26 C0 AA AF AE 00 E1

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  nonce:
  00000:   69 69 FF AA A4 52 52 81 EE BB EB 4C BD 0B 64 0E

  additional_data:
  00000:   17 03 03 00 17

  TLSInnerPlaintext:
  00000:   08 00 00 02 00 00 16

  TLSCiphertext:
  00000:   17 03 03 00 17 94 0E 5D 2C 75 3A E5 FE BD 20 01
  00010:   2C C9 E3 EB 24 A3 79 84 1E 02 AB BE

Record layer message:
type:                   17
legacy_record_version:  0303
```

```
length:                  0017
encrypted_record:        940E5D2C753AE5FEBD20012CC9E3EB24
                         A379841E02ABBE

00000:   17 03 03 00 17 94 0E 5D 2C 75 3A E5 FE BD 20 01
00010:   2C C9 E3 EB 24 A3 79 84 1E 02 AB BE

---------------------------Server---------------------------

Certificate message:
msg_type:                0B
length:                  000151
body:
  certificate_list:
    length:              00014D
    vector:
      ASN.1Cert:
        length:          000148
        vector:          308201443081F2A00302010202023039
                         300A06082A85030701010302301B3119
                         30170603550403131067F73742E6578
                         616D706C652E636F6D301E170D323030
                         3232383131303833375A170D33303032
                         3235313130383337 5A301B3119301706
                         035504031310676F73742E6578616D70
                         6C652E636F6D305E301706082A850307
                         01010101300B06092A85030701020101
                         020343000440F383CEE83048B4EB14C7
                         1A7F6DE44A37CE11A6AC1750F1CFB8DA
                         D8A38CCDD8FD06656F7CFC075F4083C3
                         716221478F1EE24C6B1B70CCE3C72AFD
                         2ACE65C775BCA321301F301D0603551D
                         0E04160414F330FA7166DF095AF3A073
                         BC3B8EA356D7DFAC71300A06082A8503
                         0701010302034100AB2EDA23F49B4862
                         3B0CFF5906B7DD3C23B473570B296A08
                         71DD15EF9A33201B97904A5CFA6C931C
                         5473DC0C5A5F2FBB2E50CF587AE27C4D
                         8E52EB80189DD08B
        extensions:
          length:        0000
          vector:        --

00000:   0B 00 01 51 00 00 01 4D 00 01 48 30 82 01 44 30
00010:   81 F2 A0 03 02 01 02 02 02 30 39 30 0A 06 08 2A
00020:   85 03 07 01 01 03 02 30 1B 31 19 30 17 06 03 55
00030:   04 03 13 10 67 6F 73 74 2E 65 78 61 6D 70 6C 65
00040:   2E 63 6F 6D 30 1E 17 0D 32 30 30 32 32 38 31 31
00050:   30 38 33 37 5A 17 0D 33 30 30 32 32 35 31 31 30
```

```
00060:   38 33 37 5A 30 1B 31 19 30 17 06 03 55 04 03 13
00070:   10 67 6F 73 74 2E 65 78 61 6D 70 6C 65 2E 63 6F
00080:   6D 30 5E 30 17 06 08 2A 85 03 07 01 01 01 01 30
00090:   0B 06 09 2A 85 03 07 01 02 01 01 02 03 43 00 04
000A0:   40 F3 83 CE E8 30 48 B4 EB 14 C7 1A 7F 6D E4 4A
000B0:   37 CE 11 A6 AC 17 50 F1 CF B8 DA D8 A3 8C CD D8
000C0:   FD 06 65 6F 7C FC 07 5F 40 83 C3 71 62 21 47 8F
000D0:   1E E2 4C 6B 1B 70 CC E3 C7 2A FD 2A CE 65 C7 75
000E0:   BC A3 21 30 1F 30 1D 06 03 55 1D 0E 04 16 04 14
000F0:   F3 30 FA 71 66 DF 09 5A F3 A0 73 BC 3B 8E A3 56
00100:   D7 DF AC 71 30 0A 06 08 2A 85 03 07 01 01 03 02
00110:   03 41 00 AB 2E DA 23 F4 9B 48 62 3B 0C FF 59 06
00120:   B7 DD 3C 23 B4 73 57 0B 29 6A 08 71 DD 15 EF 9A
00130:   33 20 1B 97 90 4A 5C FA 6C 93 1C 54 73 DC 0C 5A
00140:   5F 2F BB 2E 50 CF 58 7A E2 7C 4D 8E 52 EB 80 18
00150:   9D D0 8B 00 00
```

Record payload protection:

server_record_write_key = TLSTREE(server_write_key_hs, 1):

```
00000:   56 EE 18 13 72 72 49 C9 DC DF 35 13 78 7E DB 93
00010:   DF 62 C6 1E E7 B1 26 C5 0F 26 C0 AA AF AE 00 E1
```

seqnum:

```
00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01
```

nonce:

```
00000:   69 69 FF AA A4 52 52 81 EE BB EB 4C BD 0B 64 0F
```

additional_data:

```
00000:   17 03 03 01 66
```

TLSInnerPlaintext:

```
00000:   0B 00 01 51 00 00 01 4D 00 01 48 30 82 01 44 30
00010:   81 F2 A0 03 02 01 02 02 02 30 39 30 0A 06 08 2A
00020:   85 03 07 01 01 03 02 30 1B 31 19 30 17 06 03 55
00030:   04 03 13 10 67 6F 73 74 2E 65 78 61 6D 70 6C 65
00040:   2E 63 6F 6D 30 1E 17 0D 32 30 30 32 32 38 31 31
00050:   30 38 33 37 5A 17 0D 33 30 30 32 32 35 31 31 30
00060:   38 33 37 5A 30 1B 31 19 30 17 06 03 55 04 03 13
00070:   10 67 6F 73 74 2E 65 78 61 6D 70 6C 65 2E 63 6F
00080:   6D 30 5E 30 17 06 08 2A 85 03 07 01 01 01 01 30
00090:   0B 06 09 2A 85 03 07 01 02 01 01 02 03 43 00 04
000A0:   40 F3 83 CE E8 30 48 B4 EB 14 C7 1A 7F 6D E4 4A
000B0:   37 CE 11 A6 AC 17 50 F1 CF B8 DA D8 A3 8C CD D8
000C0:   FD 06 65 6F 7C FC 07 5F 40 83 C3 71 62 21 47 8F
000D0:   1E E2 4C 6B 1B 70 CC E3 C7 2A FD 2A CE 65 C7 75
000E0:   BC A3 21 30 1F 30 1D 06 03 55 1D 0E 04 16 04 14
000F0:   F3 30 FA 71 66 DF 09 5A F3 A0 73 BC 3B 8E A3 56
```

```
  00100:   D7 DF AC 71 30 0A 06 08 2A 85 03 07 01 01 03 02
  00110:   03 41 00 AB 2E DA 23 F4 9B 48 62 3B 0C FF 59 06
  00120:   B7 DD 3C 23 B4 73 57 0B 29 6A 08 71 DD 15 EF 9A
  00130:   33 20 1B 97 90 4A 5C FA 6C 93 1C 54 73 DC 0C 5A
  00140:   5F 2F BB 2E 50 CF 58 7A E2 7C 4D 8E 52 EB 80 18
  00150:   9D D0 8B 00 00 16

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 0166
encrypted_record:       F57944FE9A599A76E7FE9C26E3FCE5BB
                        AC4DDCF68EF2E77624E33E80B6743E39
                        10502EE419A219B3BB6A1712D15458BB
                        897D3DAC7A48769945C89237DFB86620
                        CC31C456B4374B075905E42AB5333742
                        3463819982DC6D76A067C4FD83BD3E47
                        9CD9B7FD2926A5A63B1E88B1525DB976
                        C7F409190F955AE9F0AC5F976A471F23
                        675DEB9B24E162D24F494ECDC483A070
                        7129F3BD17D0FAC4944F2B3BF140D616
                        D654709297495B23898893B211505856
                        EEC1A96BC4DCF78A016798E5500D662C
                        54A74BDF6A7F300AC9B72299B4F15F6F
                        449F396CE1D0C9243CBC1C86BECD5CAB
                        BFDF50197B7AFF4BE903D7E3311B729B
                        C32D09D2D0DCE06622985AE037DC2F87
                        CB0C492F2D5106B259CC86E227CC8338
                        C1DF6C63576B17DB9655FD255F156E1F
                        4F767FAFB74471731E4225256818DE94
                        64218263D7CF7B87EB5222E76DE6C951
                        E462CCCCC53E06387BB4FEDEFD34B9C1
                        3AB4EE3D49057CD2672F852A5F692408
                        29B92341CDC9

TLSCiphertext:
  00000:   17 03 03 01 66 F5 79 44 FE 9A 59 9A 76 E7 FE 9C
  00010:   26 E3 FC E5 BB AC 4D DC F6 8E F2 E7 76 24 E3 3E
  00020:   80 B6 74 3E 39 10 50 2E E4 19 A2 19 B3 BB 6A 17
  00030:   12 D1 54 58 BB 89 7D 3D AC 7A 48 76 99 45 C8 92
  00040:   37 DF B8 66 20 CC 31 C4 56 B4 37 4B 07 59 05 E4
  00050:   2A B5 33 37 42 34 63 81 99 82 DC 6D 76 A0 67 C4
  00060:   FD 83 BD 3E 47 9C D9 B7 FD 29 26 A5 A6 3B 1E 88
  00070:   B1 52 5D B9 76 C7 F4 09 19 0F 95 5A E9 F0 AC 5F
  00080:   97 6A 47 1F 23 67 5D EB 9B 24 E1 62 D2 4F 49 4E
  00090:   CD C4 83 A0 70 71 29 F3 BD 17 D0 FA C4 94 4F 2B
  000A0:   3B F1 40 D6 16 D6 54 70 92 97 49 5B 23 89 88 93
  000B0:   B2 11 50 58 56 EE C1 A9 6B C4 DC F7 8A 01 67 98
  000C0:   E5 50 0D 66 2C 54 A7 4B DF 6A 7F 30 0A C9 B7 22
```

```
  000D0:   99 B4 F1 5F 6F 44 9F 39 6C E1 D0 C9 24 3C BC 1C
  000E0:   86 BE CD 5C AB BF DF 50 19 7B 7A FF 4B E9 03 D7
  000F0:   E3 31 1B 72 9B C3 2D 09 D2 D0 DC E0 66 22 98 5A
  00100:   E0 37 DC 2F 87 CB 0C 49 2F 2D 51 06 B2 59 CC 86
  00110:   E2 27 CC 83 38 C1 DF 6C 63 57 6B 17 DB 96 55 FD
  00120:   25 5F 15 6E 1F 4F 76 7F AF B7 44 71 73 1E 42 25
  00130:   25 68 18 DE 94 64 21 82 63 D7 CF 7B 87 EB 52 22
  00140:   E7 6D E6 C9 51 E4 62 CC CC C5 3E 06 38 7B B4 FE
  00150:   DE FD 34 B9 C1 3A B4 EE 3D 49 05 7C D2 67 2F 85
  00160:   2A 5F 69 24 08 29 B9 23 41 CD C9

--------------------------Server---------------------------

M = HMCertificateVerify = (ClientHello, ServerHello,
  EncryptedExtensions, Certificate)
Transcript-Hash(HMCertificateVerify):
  00000:   E0 CC 4B C1 4B EC 5D 13 19 2C DC 66 22 B4 FD A9
  00010:   67 6A 1B 50 E4 56 83 0B B5 F0 7E 01 21 22 73 06

k (random for signature algorithm):
  00000:   85 85 85 85 85 85 85 85 85 85 85 85 85 85 85 85
  00010:   85 85 85 85 85 85 85 85 85 85 85 85 85 85 85 85

sgn:
  00000:   A0 AA 13 91 5C 5B 80 C6 02 E2 FD 85 80 4F 99 2C
  00010:   77 15 97 AD 37 85 7A D6 BC 2A 9D 7B C5 FE BE C3
  00020:   7C 72 94 BA A2 3C F6 9D 03 E4 71 0B D7 08 13 FD
  00030:   AC 59 6B C1 58 E7 56 BD 37 1C 44 2E 95 22 DE 87

CertificateVerify message:
msg_type:                0F
length:                  000000
body:
  algorithm:             070A
  signature:
    length:              0040
    vector:
                         A0AA13915C5B80C602E2FD85804F992C
                         771597AD37857AD6BC2A9D7BC5FEBEC3
                         7C7294BAA23CF69D03E4710BD70813FD
                         AC596BC158E756BD371C442E9522DE87

00000:   0F 00 00 44 07 0A 00 40 A0 AA 13 91 5C 5B 80 C6
00010:   02 E2 FD 85 80 4F 99 2C 77 15 97 AD 37 85 7A D6
00020:   BC 2A 9D 7B C5 FE BE C3 7C 72 94 BA A2 3C F6 9D
00030:   03 E4 71 0B D7 08 13 FD AC 59 6B C1 58 E7 56 BD
00040:   37 1C 44 2E 95 22 DE 87

Record payload protection:
```

```
  server_record_write_key = TLSTREE(server_write_key_hs, 2):
  00000:   56 EE 18 13 72 72 49 C9 DC DF 35 13 78 7E DB 93
  00010:   DF 62 C6 1E E7 B1 26 C5 0F 26 C0 AA AF AE 00 E1

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02

  nonce:
  00000:   69 69 FF AA A4 52 52 81 EE BB EB 4C BD 0B 64 0C

  additional_data:
  00000:   17 03 03 00 59

  TLSInnerPlaintext:
  00000:   0F 00 00 44 07 0A 00 40 A0 AA 13 91 5C 5B 80 C6
  00010:   02 E2 FD 85 80 4F 99 2C 77 15 97 AD 37 85 7A D6
  00020:   BC 2A 9D 7B C5 FE BE C3 7C 72 94 BA A2 3C F6 9D
  00030:   03 E4 71 0B D7 08 13 FD AC 59 6B C1 58 E7 56 BD
  00040:   37 1C 44 2E 95 22 DE 87 16

Record layer message:
type:                    17
legacy_record_version:   0303
length:                  0059
encrypted_record:        52631D5BFDF48254BDFB5F9E02A6A527
                         0163BCE1E0D818E8D74176535C6CDD25
                         2DE065AE77984A65ADBA036D59CF45B9
                         A0047BABCCD0B28044D34BCFD09E6E46
                         27044B26FE5CA734FCB08607146F41A8
                         71C3F95384B48ADABC

TLSCiphertext:
  00000:   17 03 03 00 59 52 63 1D 5B FD F4 82 54 BD FB 5F
  00010:   9E 02 A6 A5 27 01 63 BC E1 E0 D8 18 E8 D7 41 76
  00020:   53 5C 6C DD 25 2D E0 65 AE 77 98 4A 65 AD BA 03
  00030:   6D 59 CF 45 B9 A0 04 7B AB CC D0 B2 80 44 D3 4B
  00040:   CF D0 9E 6E 46 27 04 4B 26 FE 5C A7 34 FC B0 86
  00050:   07 14 6F 41 A8 71 C3 F9 53 84 B4 8A DA BC

--------------------------Server---------------------------

server_finished_key = HKDF-Expand-Label(SHTS, "finished", "", 32):
00000:   53 F1 C0 38 8F 8A 70 C0 BC A0 DD 21 A0 30 F2 38
00010:   1C 34 37 CD 0E 7E C9 3D 0A 96 5E 25 63 2D D7 9A

HMFinished = (ClientHello, ServerHello, EncryptedExtensions
  Certificate, CertificateVerify)
Transcript-Hash(HMFinished):
00000:   03 EC 9B 1D 0B 37 41 42 45 72 BA C9 DF 3A A5 2C
00010:   03 EF E9 E9 58 07 69 43 AF D8 58 19 BC 60 2F 46
```

```
FinishedHash =
HMAC(server_finished_key,Transcript-Hash(HMFinished)):
00000:   E0 BA A3 36 14 E0 69 69 7E 4D FA B0 71 B9 72 57
00010:   73 F8 FE 1A 32 6A 66 2D 0F 52 30 9B 45 B6 E0 31

Finished message:
msg_type:               14
length:                 000020
body:
  verify_data:          E0BAA33614E069697E4DFAB071B97257
                        73F8FE1A326A662D0F52309B45B6E031

00000:   14 00 00 20 E0 BA A3 36 14 E0 69 69 7E 4D FA B0
00010:   71 B9 72 57 73 F8 FE 1A 32 6A 66 2D 0F 52 30 9B
00020:   45 B6 E0 31

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_hs, 3):
  00000:   56 EE 18 13 72 72 49 C9 DC DF 35 13 78 7E DB 93
  00010:   DF 62 C6 1E E7 B1 26 C5 0F 26 C0 AA AF AE 00 E1

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03

  nonce:
  00000:   69 69 FF AA A4 52 52 81 EE BB EB 4C BD 0B 64 0D

  additional_data:
  00000:   17 03 03 00 35

  TLSInnerPlaintext:
  00000:   14 00 00 20 E0 BA A3 36 14 E0 69 69 7E 4D FA B0
  00010:   71 B9 72 57 73 F8 FE 1A 32 6A 66 2D 0F 52 30 9B
  00020:   45 B6 E0 31 16

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 0035
encrypted_record:       57B1706C4918F67EACCA457F7D5B537C
                        CE5036B4004C778022B97EE802320398
                        119404506680ADD7D6A6CD7C8153B755
                        3C6E646AD6

TLSCiphertext:
  00000:   17 03 03 00 35 57 B1 70 6C 49 18 F6 7E AC CA 45
  00010:   7F 7D 5B 53 7C CE 50 36 B4 00 4C 77 80 22 B9 7E
  00020:   E8 02 32 03 98 11 94 04 50 66 80 AD D7 D6 A6 CD
```

```
  00030:   7C 81 53 B7 55 3C 6E 64 6A D6

---------------------------Server---------------------------
Application Data:
  HELO gost.example.com\r\n

Record payload protection:

  Derived #1 = Derive-Secret(HandshakeSecret, "derived", "") =
    HKDF-Expand-Label(HandshakeSecret, "derived", "", 32):
    00000:   EA 3C 54 BB D1 4E F9 D7 50 77 6F AB E3 95 BE 2A
    00010:   BD DB BB B7 1C 13 C2 BD 60 9E 35 15 79 4A FA 02

  MasterSecret = HKDF-Extract(Salt: Derived #1, IKM: 0^256):
  00000:   31 BB 1D 61 2C CD 53 32 68 8A 55 1A 48 CA 25 0F
  00010:   24 78 3D 4A B0 B4 A7 6D 3F E5 06 7A 26 16 A4 A3

  HM2 = (ClientHello, ServerHello, EncryptedExtensions, Certificate,
   CertificateVerify, Server Finished)
  TH2 = Transcript-Hash(HM2):
  00000:   9E BC 5F BE 32 D9 F4 0D 48 F8 EE CE BB 62 31 A5
  00010:   33 C2 C0 EF 24 32 77 B9 6D 6F 7A D3 BB FD 14 94

  server_application_traffic_secret (SATS):
  SATS = Derive-Secret(MasterSecret, "s ap traffic", HM2) =
    HKDF-Expand-Label(MasterSecret, "s ap traffic", TH2, 32):
    00000:   87 73 4F 4B 4C FD 17 B9 7B 83 4D 82 2D 9D 73 79
    00010:   F6 F5 E0 3B 80 B5 2A EB 2A FF 51 0E DD 83 DB D2

  server_write_key_ap = HKDF-Expand-Label(SATS, "key", "", 32):
  00000:   47 5E 4C 51 4C C6 31 8C 3A 5F 00 0F 12 65 BD 1A
  00010:   B5 F0 DE 1A F3 57 ED 00 79 EC 5F F0 AF BD 03 0C

  server_write_iv_ap = HKDF-Expand-Label(SATS, "iv", "", 16):
  00000:   AF E9 1F 71 18 35 40 26 31 7E 1A B4 D8 22 17 B8

  server_record_write_key = TLSTREE(server_write_key_ap, 0):
  00000:   C8 FC 93 D7 C5 86 F2 B0 A3 10 1B AA 6A 97 9E 4E
  00010:   38 86 70 65 51 E8 11 87 E9 78 80 40 9C 7E 8E E9

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  nonce:
  00000:   2F E9 1F 71 18 35 40 26 31 7E 1A B4 D8 22 17 B8

  additional_data:
  00000:   17 03 03 00 28

  TLSInnerPlaintext:
```

```
  00000:   48 45 4C 4F 20 67 6F 73 74 2E 65 78 61 6D 70 6C
  00010:   65 2E 63 6F 6D 0D 0A 17

Record layer message:
type:                    17
legacy_record_version:   0303
length:                  0028
encrypted_record:        ABB8C372C79681DCE5C3C909DD039D59
                         8161FD3E6CE5D6F9CA5715BD6B5C1824
                         7FB26AC1AB396A4E

TLSCiphertext:
  00000:   17 03 03 00 28 AB B8 C3 72 C7 96 81 DC E5 C3 C9
  00010:   09 DD 03 9D 59 81 61 FD 3E 6C E5 D6 F9 CA 57 15
  00020:   BD 6B 5C 18 24 7F B2 6A C1 AB 39 6A 4E

---------------------------Client---------------------------

client_finished_key = HKDF-Expand-Label(CHTS, "finished", "", 32):
00000:   2F 21 54 8C F5 27 78 69 AE 49 0D E7 BC 15 AC E6
00010:   39 F6 57 E3 58 2A 5A 63 4B 0A 91 56 95 D5 4C 42

HM2 = (ClientHello, ServerHello, EncryptedExtensions, Certificate,
  CertificateVerify, Server Finished)
Transcript-Hash(HM2):
00000:   9E BC 5F BE 32 D9 F4 0D 48 F8 EE CE BB 62 31 A5
00010:   33 C2 C0 EF 24 32 77 B9 6D 6F 7A D3 BB FD 14 94

FinishedHash =
HMAC(client_finished_key,Transcript-Hash(HM2)):
00000:   08 5F C7 FD 79 B6 D1 11 CD 8D 3F F6 B2 3A 06 5A
00010:   7A F7 A6 38 73 42 A5 F3 57 68 14 CD 00 47 19 D2

Finished message:
msg_type:                14
length:                  000020
body:
  verify_data:           085FC7FD79B6D111CD8D3FF6B23A065A
                         7AF7A6387342A5F3576814CD004719D2

00000:   14 00 00 20 08 5F C7 FD 79 B6 D1 11 CD 8D 3F F6
00010:   B2 3A 06 5A 7A F7 A6 38 73 42 A5 F3 57 68 14 CD
00020:   00 47 19 D2

Record payload protection:

  EarlySecret = HKDF-Extract(Salt: 0^256, IKM: 0^256):
  00000:   FB DE FB E5 27 FE EA 66 5A AB 92 77 A2 16 3B 83
  00010:   43 08 4F D1 91 C4 60 66 26 0F AC 6F D1 43 6C 72
```

```
  Derived #0 = Derive-Secret(EarlySecret, "derived", "") =
  HKDF-Expand-Label(EarlySecret, "derived", "", 32):
  00000:   DB C3 C8 26 D8 77 A3 B7 D2 D2 45 3D BF DC 6C FB
  00010:   FB 11 51 B3 E8 4F 0C 8F 26 01 1D 8D 5B F3 ED F7

  HandshakeSecret = HKDF-Extract(Salt: Derived #0, IKM: ECDHE):
  00000:   44 24 5E 2C 43 32 D1 F7 8B 0F 8D 16 F4 03 EB 69
  00010:   ED 2A 40 53 84 7C DC 39 FA 8B 3D 29 74 F7 45 E7

  HM1 = (ClientHello, ServerHello)
  TH1 = Transcript-Hash(HM1):
  00000:   99 3B A7 22 12 4A F3 CB FD 47 71 E7 FA E3 2A C1
  00010:   D0 E9 27 8C F7 84 3F CB C6 20 E1 A0 08 5A 87 A1

  client_handshake_traffic_secret (CHTS):
    CHTS = Derive-Secret(HandshakeSecret, "c hs traffic", HM1) =
    HKDF-Expand-Label(HandshakeSecret, "c hs traffic", TH1, 32):
    00000:   B3 F7 11 3D 35 26 55 4F E6 55 E5 6F AB 79 B1 A0
    00010:   3D E3 35 96 E3 30 88 C7 78 37 19 A9 A4 B0 DC CD

  client_write_key_hs = HKDF-Expand-Label(CHTS, "key", "", 32):
  00000:   58 16 88 D7 6E FE 12 2B B5 5F 62 B3 8E F0 1B CC
  00010:   8C 88 DB 83 E9 EA 4D 55 D3 89 8C 53 72 1F C3 84

  client_write_iv_hs = HKDF-Expand-Label(CHTS, "iv", "", 16):
  00000:   43 9A 07 45 3D 0B EA 0C 1D 1B EB 73 8E B5 B8 DD

  client_record_write_key = TLSTREE(client_write_key_hs, 0):
  00000:   E1 C5 9B 41 69 D8 96 10 7F 78 45 68 93 A3 75 1E
  00010:   15 73 54 3D AD 8C B7 40 69 E6 81 4A 51 3B BB 1C

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  nonce:
  00000:   43 9A 07 45 3D 0B EA 0C 1D 1B EB 73 8E B5 B8 DD

  additional_data:
  00000:   17 03 03 00 35

  TLSInnerPlaintext:
  00000:   14 00 00 20 08 5F C7 FD 79 B6 D1 11 CD 8D 3F F6
  00010:   B2 3A 06 5A 7A F7 A6 38 73 42 A5 F3 57 68 14 CD
  00020:   00 47 19 D2 16

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 0035
encrypted_record:       C9C65EAAB4A80E04849A122EB0CC26A9
```

```
                          CA6B5DD4DB7AD6813949F629FC09E052
                          2FF7ACDBBA93926C20008B8CCE865422
                          7B31D439F8

TLSCiphertext:
  00000:   17 03 03 00 35 C9 C6 5E AA B4 A8 0E 04 84 9A 12
  00010:   2E B0 CC 26 A9 CA 6B 5D D4 DB 7A D6 81 39 49 F6
  00020:   29 FC 09 E0 52 2F F7 AC DB BA 93 92 6C 20 00 8B
  00030:   8C CE 86 54 22 7B 31 D4 39 F8

---------------------------Server---------------------------

NewSessionTicket message:
msg_type:                 04
length:                   000035
body:
  ticket_lifetime:        00093A80
  ticket_age_add:         86868686
  ticket_nonce:
    length:               08
    vector:               0000000000000000
  ticket:
    length:               0020
    vector:               88888888888888888888888888888888
                          88888888888888888888888888888888
  extensions:
    length:               0000
    vector:               --

00000:   04 00 00 35 00 09 3A 80 86 86 86 86 08 00 00 00
00010:   00 00 00 00 00 00 20 88 88 88 88 88 88 88 88 88
00020:   88 88 88 88 88 88 88 88 88 88 88 88 88 88 88 88
00030:   88 88 88 88 88 88 88 00 00

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_ap, 1):
  00000:   C8 FC 93 D7 C5 86 F2 B0 A3 10 1B AA 6A 97 9E 4E
  00010:   38 86 70 65 51 E8 11 87 E9 78 80 40 9C 7E 8E E9

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01

  nonce:
  00000:   2F E9 1F 71 18 35 40 26 31 7E 1A B4 D8 22 17 B9

  additional_data:
  00000:   17 03 03 00 4A

  TLSInnerPlaintext:
```

```
  00000:   04 00 00 35 00 09 3A 80 86 86 86 86 08 00 00 00
  00010:   00 00 00 00 00 00 20 88 88 88 88 88 88 88 88 88
  00020:   88 88 88 88 88 88 88 88 88 88 88 88 88 88 88 88
  00030:   88 88 88 88 88 88 88 00 00 16

Record layer message:
type:                    17
legacy_record_version:   0303
length:                  004A
encrypted_record:        CA6688A5DC22208DC8A8DE7E597292E3
                         CB5D454945B8F06C7C50F1823D7B6BB0
                         021178AE3ADB2DE3994539FD696945CF
                         AA6919F3F1294CD41ED2A8EA75302869
                         ACB994F3920B09D67186

TLSCiphertext:
  00000:   17 03 03 00 4A CA 66 88 A5 DC 22 20 8D C8 A8 DE
  00010:   7E 59 72 92 E3 CB 5D 45 49 45 B8 F0 6C 7C 50 F1
  00020:   82 3D 7B 6B B0 02 11 78 AE 3A DB 2D E3 99 45 39
  00030:   FD 69 69 45 CF AA 69 19 F3 F1 29 4C D4 1E D2 A8
  00040:   EA 75 30 28 69 AC B9 94 F3 92 0B 09 D6 71 86

---------------------------Server---------------------------

Application data:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Pad: 15360 bytes

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_ap, 2):
  00000:   C8 FC 93 D7 C5 86 F2 B0 A3 10 1B AA 6A 97 9E 4E
  00010:   38 86 70 65 51 E8 11 87 E9 78 80 40 9C 7E 8E E9

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02

  nonce:
  00000:   2F E9 1F 71 18 35 40 26 31 7E 1A B4 D8 22 17 BA

  additional_data:
  00000:   17 03 03 40 11

  TLSInnerPlaintext:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000400:  17 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
  00000410:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  00004000:  00

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 4011
encrypted_record:       9B3AD6939F05A403EEB1A636E13989D9
                        1CCA6A45BE5B7CB5C980020627A1B2AD
                        34AC4B5AAE5BD445C91C28325E4C7149
                        188D55EF27016D80AF440704820BCE22
                        CE501EA619A4FF7CD9F722A28391CE8B
                        B86BF87D5A85555BEF59A9C9A1572F38
                        114E64FD04A0DB2E1787A585EA51DCAB
                        B95DAFB73D0B3FE3F0702C5E1AA01571
                        17D884783E5E6113F6CA8352F6CF49F9
                        DB3B3DAB380BFD7BE04B0A  [...]
                        64E7027D926E0F95AB7F133B5921F996
                        A81EB67B78449DD32F4511E013206524
                        AD4AFACF0B1C1622282CB20A965E670C
                        C9A17E13F343AF3825AFD58B06915BDC
                        9E49477F02830694F5AC7CC99C887F62
                        CDAAEF0053766FB12BC9A082C157C347
                        21C5400C376088A660EE4329ED645D7C
                        07D4DA1ABDF6F9A1B9D51BF3E09CFCC1
                        A59CD96F07FC9ACF004EA1B20E6BBDAD
                        7BBF0C9E2A1B

TLSCiphertext:
  00000000:   17 03 03 40 11 9B 3A D6 93 9F 05 A4 03 EE B1 A6
  00000010:   36 E1 39 89 D9 1C CA 6A 45 BE 5B 7C B5 C9 80 02
  00000020:   06 27 A1 B2 AD 34 AC 4B 5A AE 5B D4 45 C9 1C 28
  00000030:   32 5E 4C 71 49 18 8D 55 EF 27 01 6D 80 AF 44 07
  00000040:   04 82 0B CE 22 CE 50 1E A6 19 A4 FF 7C D9 F7 22
  00000050:   A2 83 91 CE 8B B8 6B F8 7D 5A 85 55 5B EF 59 A9
  00000060:   C9 A1 57 2F 38 11 4E 64 FD 04 A0 DB 2E 17 87 A5
  00000070:   85 EA 51 DC AB B9 5D AF B7 3D 0B 3F E3 F0 70 2C
  00000080:   5E 1A A0 15 71 17 D8 84 78 3E 5E 61 13 F6 CA 83
  00000090:   52 F6 CF 49 F9 DB 3B 3D AB 38 0B FD 7B E0 4B 0A
  [...]
  00003F80:   64 E7 02 7D 92 6E 0F 95 AB 7F 13 3B 59 21 F9 96
  00003F90:   A8 1E B6 7B 78 44 9D D3 2F 45 11 E0 13 20 65 24
  00003FA0:   AD 4A FA CF 0B 1C 16 22 28 2C B2 0A 96 5E 67 0C
  00003FB0:   C9 A1 7E 13 F3 43 AF 38 25 AF D5 8B 06 91 5B DC
  00003FC0:   9E 49 47 7F 02 83 06 94 F5 AC 7C C9 9C 88 7F 62
  00003FD0:   CD AA EF 00 53 76 6F B1 2B C9 A0 82 C1 57 C3 47
  00003FE0:   21 C5 40 0C 37 60 88 A6 60 EE 43 29 ED 64 5D 7C
  00003FF0:   07 D4 DA 1A BD F6 F9 A1 B9 D5 1B F3 E0 9C FC C1
```

```
  00004000:   A5 9C D9 6F 07 FC 9A CF 00 4E A1 B2 0E 6B BD AD
  00004010:   7B BF 0C 9E 2A 1B

---------------------------Server---------------------------

Application data:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Pad: 15360 bytes

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_ap, 3):
  00000:   C8 FC 93 D7 C5 86 F2 B0 A3 10 1B AA 6A 97 9E 4E
  00010:   38 86 70 65 51 E8 11 87 E9 78 80 40 9C 7E 8E E9

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 03

  nonce:
  00000:   2F E9 1F 71 18 35 40 26 31 7E 1A B4 D8 22 17 BB

  additional_data:
  00000:   17 03 03 40 11

  TLSInnerPlaintext:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000400:  17 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000410:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  00004000:  00

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 4011
encrypted_record:       F06C5032F8A7AD58CED14D5383ED969E
                        628DE4F35CF9B6FCF5047D9B02261F56
                        F724DE961F8FF9C27AE76FBAC0A18E96
                        AA30CA7D8EBAD5B5135A0962515CC4F2
                        A16EBAB9A08886ED4EFD9DFAEC158F94
                        EFB0F90725C9114D9D8D904A18ABF184
                        E74B07150B2F2F27CB8032064943C957
                        11E480E4F4FCE8E9F020D5C90489E734
                        AEA10D91C7097AEC8CD6D5E3EEC764B0
                        CD447FC07735F0F8D9D490  [...]
                        3A79E7B3BCFD2B2478092911073A7CC9
```

```
                          6AC626C30DD0A5612DBBFF26E35AF0BB
                          5CEC24EED391100533FB999D4873ED5D
                          5E4693C5EEDC3ECC3C6EFF041B0A7F42
                          25A1092F4AADD9A508C7A56CB13A3F33
                          E844E28C8ADCD45250F4EE29834C5CAA
                          C50B5EBF94501785664E78AE9B5FDBFA
                          DF730DED97985D659135F5DABAD883FF
                          AC6046A0F629881F76147646D8E2A867
                          3B14295621F7


TLSCiphertext:
  00000000:   17 03 03 40 11 F0 6C 50 32 F8 A7 AD 58 CE D1 4D
  00000010:   53 83 ED 96 9E 62 8D E4 F3 5C F9 B6 FC F5 04 7D
  00000020:   9B 02 26 1F 56 F7 24 DE 96 1F 8F F9 C2 7A E7 6F
  00000030:   BA C0 A1 8E 96 AA 30 CA 7D 8E BA D5 B5 13 5A 09
  00000040:   62 51 5C C4 F2 A1 6E BA B9 A0 88 86 ED 4E FD 9D
  00000050:   FA EC 15 8F 94 EF B0 F9 07 25 C9 11 4D 9D 8D 90
  00000060:   4A 18 AB F1 84 E7 4B 07 15 0B 2F 2F 27 CB 80 32
  00000070:   06 49 43 C9 57 11 E4 80 E4 F4 FC E8 E9 F0 20 D5
  00000080:   C9 04 89 E7 34 AE A1 0D 91 C7 09 7A EC 8C D6 D5
  00000090:   E3 EE C7 64 B0 CD 44 7F C0 77 35 F0 F8 D9 D4 90
  [...]
  00003F80:   3A 79 E7 B3 BC FD 2B 24 78 09 29 11 07 3A 7C C9
  00003F90:   6A C6 26 C3 0D D0 A5 61 2D BB FF 26 E3 5A F0 BB
  00003FA0:   5C EC 24 EE D3 91 10 05 33 FB 99 9D 48 73 ED 5D
  00003FB0:   5E 46 93 C5 EE DC 3E CC 3C 6E FF 04 1B 0A 7F 42
  00003FC0:   25 A1 09 2F 4A AD D9 A5 08 C7 A5 6C B1 3A 3F 33
  00003FD0:   E8 44 E2 8C 8A DC D4 52 50 F4 EE 29 83 4C 5C AA
  00003FE0:   C5 0B 5E BF 94 50 17 85 66 4E 78 AE 9B 5F DB FA
  00003FF0:   DF 73 0D ED 97 98 5D 65 91 35 F5 DA BA D8 83 FF
  00004000:   AC 60 46 A0 F6 29 88 1F 76 14 76 46 D8 E2 A8 67
  00004010:   3B 14 29 56 21 F7

---------------------------Server---------------------------

Application data:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Pad: 15360 bytes

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_ap, 8):
  00000:   D3 CD 87 D5 68 74 07 82 39 78 34 4C 06 B9 28 A8
  00010:   58 98 B7 39 A3 1D 3D E5 FF 2B 78 8E F3 91 96 ED

  seqnum:
```

```
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 08

  nonce:
  00000:   2F E9 1F 71 18 35 40 26 31 7E 1A B4 D8 22 17 B0

  additional_data:
  00000:   17 03 03 40 11

  TLSInnerPlaintext:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000400:  17 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000410:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  00004000:  00

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 4011
encrypted_record:       E3DF00F169A76FA19FE55FA304E0A552
                        5A28FDBD3DD4CA654B89140EFD69E263
                        28A65A77F5D8B2E2F73568F7A677E5DF
                        8D225FAA8ED5FED98F09963FF1E82161
                        81595E9FA6989CCABC2150CA668D70EA
                        8CB6F62BCC528D26B52FB27AB70F194A
                        30E5C9085D9323D3874093070D15650
                        52468045F3398DC5BF93D6A983956E1D
                        3077337B773DAF4B9A6BA5BC569A251D
                        34FE23DF7B9343A0550094  [...]
                        2B516EE4A4971FD26EFB9293981435E9
                        FCC560B618B8ED0A52589E7342C25325
                        11C3D7C145559B8119BC02CB22CBF1EB
                        915578E8468806B2D0728C591B617354
                        CC47D51FF2363197A559018AD3498846
                        AD167DD086BD12EF52179D45ABF06C28
                        97B0C1D8AAD49413E0CCC086586D537A
                        296F9CEEB7E7E1DD2537540232C6BD71
                        619FC93BAE3FD8B0C95EA9915B6127B9
                        9F87884541F7

TLSCiphertext:
  00000000:   17 03 03 40 11 E3 DF 00 F1 69 A7 6F A1 9F E5 5F
  00000010:   A3 04 E0 A5 52 5A 28 FD BD 3D D4 CA 65 4B 89 14
  00000020:   0E FD 69 E2 63 28 A6 5A 77 F5 D8 B2 E2 F7 35 68
  00000030:   F7 A6 77 E5 DF 8D 22 5F AA 8E D5 FE D9 8F 09 96
  00000040:   3F F1 E8 21 61 81 59 5E 9F A6 98 9C CA BC 21 50
  00000050:   CA 66 8D 70 EA 8C B6 F6 2B CC 52 8D 26 B5 2F B2
```

```
  00000060:   7A B7 0F 19 4A 30 E5 C9 08 5D 93 23 D3 87 45 09
  00000070:   30 70 D1 56 50 52 46 80 45 F3 39 8D C5 BF 93 D6
  00000080:   A9 83 95 6E 1D 30 77 33 7B 77 3D AF 4B 9A 6B A5
  00000090:   BC 56 9A 25 1D 34 FE 23 DF 7B 93 43 A0 55 00 94
  [...]
  00003F80:   2B 51 6E E4 A4 97 1F D2 6E FB 92 93 98 14 35 E9
  00003F90:   FC C5 60 B6 18 B8 ED 0A 52 58 9E 73 42 C2 53 25
  00003FA0:   11 C3 D7 C1 45 55 9B 81 19 BC 02 CB 22 CB F1 EB
  00003FB0:   91 55 78 E8 46 88 06 B2 D0 72 8C 59 1B 61 73 54
  00003FC0:   CC 47 D5 1F F2 36 31 97 A5 59 01 8A D3 49 88 46
  00003FD0:   AD 16 7D D0 86 BD 12 EF 52 17 9D 45 AB F0 6C 28
  00003FE0:   97 B0 C1 D8 AA D4 94 13 E0 CC C0 86 58 6D 53 7A
  00003FF0:   29 6F 9C EE B7 E7 E1 DD 25 37 54 02 32 C6 BD 71
  00004000:   61 9F C9 3B AE 3F D8 B0 C9 5E A9 91 5B 61 27 B9
  00004010:   9F 87 88 45 41 F7

---------------------------Server---------------------------

Application data:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Pad: 15360 bytes

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_ap, 9):
  00000:   D3 CD 87 D5 68 74 07 82 39 78 34 4C 06 B9 28 A8
  00010:   58 98 B7 39 A3 1D 3D E5 FF 2B 78 8E F3 91 96 ED

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 09

  nonce:
  00000:   2F E9 1F 71 18 35 40 26 31 7E 1A B4 D8 22 17 B1

  additional_data:
  00000:   17 03 03 40 11

  TLSInnerPlaintext:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000400:  17 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000410:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  00004000:  00

Record layer message:
type:                   17
```

```
legacy_record_version:   0303
length:                  4011
encrypted_record:        4AFCD1257E2C8D4626BC0BFBB30F2F9C
                         A57A9D0DEC090B4248CAADDFE7AA4AEB
                         770F285384FEA308CADE2EEF318148C2
                         BED4870ABEE1955CCA41CE8344C3EDA4
                         7C2512CDD19FD54C7E0260BBC7BD8DD1
                         EE9D4EBADD3D7915D0A029D7847CA05D
                         078068CC8A792FED69A4E655A6E6D22D
                         A134ECA2BDECA1E59D3AE7313E45E785
                         AF89A8F1890BFCC59F03F39C4FB2337C
                         326D94FA04F5548619D6DC  [...]
                         79B6F56B6EBF8B8860436EFF9D8F03CC
                         73BDF446D30F918AF8FF8BA2D078D243
                         1AC04657D7871203F15969F160820D7D
                         FCA78F65FF954CE5549F2C78AA3A0885
                         04527FC561B6AE06020A8772B75CE933
                         6CAC35B536A50DB26930BFA21E9EB56E
                         A20E39CC2BBBA66D41C2E8720AA0143D
                         298D8036D7B0090A0214F58C5B1890A7
                         5B4783820395E39421F4357A49597EB0
                         64123818EACE

TLSCiphertext:
  00000000:   17 03 03 40 11 4A FC D1 25 7E 2C 8D 46 26 BC 0B
  00000010:   FB B3 0F 2F 9C A5 7A 9D 0D EC 09 0B 42 48 CA AD
  00000020:   DF E7 AA 4A EB 77 0F 28 53 84 FE A3 08 CA DE 2E
  00000030:   EF 31 81 48 C2 BE D4 87 0A BE E1 95 5C CA 41 CE
  00000040:   83 44 C3 ED A4 7C 25 12 CD D1 9F D5 4C 7E 02 60
  00000050:   BB C7 BD 8D D1 EE 9D 4E BA DD 3D 79 15 D0 A0 29
  00000060:   D7 84 7C A0 5D 07 80 68 CC 8A 79 2F ED 69 A4 E6
  00000070:   55 A6 E6 D2 2D A1 34 EC A2 BD EC A1 E5 9D 3A E7
  00000080:   31 3E 45 E7 85 AF 89 A8 F1 89 0B FC C5 9F 03 F3
  00000090:   9C 4F B2 33 7C 32 6D 94 FA 04 F5 54 86 19 D6 DC
  [...]
  00003F80:   79 B6 F5 6B 6E BF 8B 88 60 43 6E FF 9D 8F 03 CC
  00003F90:   73 BD F4 46 D3 0F 91 8A F8 FF 8B A2 D0 78 D2 43
  00003FA0:   1A C0 46 57 D7 87 12 03 F1 59 69 F1 60 82 0D 7D
  00003FB0:   FC A7 8F 65 FF 95 4C E5 54 9F 2C 78 AA 3A 08 85
  00003FC0:   04 52 7F C5 61 B6 AE 06 02 0A 87 72 B7 5C E9 33
  00003FD0:   6C AC 35 B5 36 A5 0D B2 69 30 BF A2 1E 9E B5 6E
  00003FE0:   A2 0E 39 CC 2B BB A6 6D 41 C2 E8 72 0A A0 14 3D
  00003FF0:   29 8D 80 36 D7 B0 09 0A 02 14 F5 8C 5B 18 90 A7
  00004000:   5B 47 83 82 03 95 E3 94 21 F4 35 7A 49 59 7E B0
  00004010:   64 12 38 18 EA CE

----------------------------Client-------------------------------

Application data:
```

```
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Pad: 15360 bytes

Record payload protection:

  Derived #1 = Derive-Secret(HandshakeSecret, "derived", "") =
    HKDF-Expand-Label(HandshakeSecret, "derived", "", 32):
    00000:   EA 3C 54 BB D1 4E F9 D7 50 77 6F AB E3 95 BE 2A
    00010:   BD DB BB B7 1C 13 C2 BD 60 9E 35 15 79 4A FA 02

  MasterSecret = HKDF-Extract(Salt: Derived #1, IKM: 0^256):
  00000:   31 BB 1D 61 2C CD 53 32 68 8A 55 1A 48 CA 25 0F
  00010:   24 78 3D 4A B0 B4 A7 6D 3F E5 06 7A 26 16 A4 A3

  HM2 = (ClientHello, ServerHello, EncryptedExtensions, Certificate,
   CertificateVerify, Server Finished)
  TH2 = Transcript-Hash(HM2):
  00000:   9E BC 5F BE 32 D9 F4 0D 48 F8 EE CE BB 62 31 A5
  00010:   33 C2 C0 EF 24 32 77 B9 6D 6F 7A D3 BB FD 14 94

  client_application_traffic_secret (CATS):
  CATS = Derive-Secret(MasterSecret, "c ap traffic", HM2) =
    HKDF-Expand-Label(MasterSecret, "c ap traffic", TH2, 32):
    00000:   8A CF 74 6B EC 31 17 6C BD 14 2C 75 80 6C 27 0A
    00010:   0A EF 6F C3 8E 0D 8F DC B5 A8 85 25 36 3A DE 81

  client_write_key_ap = HKDF-Expand-Label(CATS, "key", "", 32):
  00000:   7B E6 4E 2C 12 78 7B 5B 8C 87 56 C4 3D 92 FA EF
  00010:   64 F1 5A 3A 3C 10 81 AD 34 BC A5 06 F0 32 24 15

  client_write_iv_ap = HKDF-Expand-Label(CATS, "iv", "", 16):
  00000:   31 09 57 EF 71 31 44 33 F5 76 CC 9B 00 AD 93 54

  server_record_write_key = TLSTREE(server_write_key_ap, 0):
  00000:   D4 9A 57 15 49 E7 48 94 9F A2 4B 88 34 23 2C A8
  00010:   75 D3 7A 26 C4 BB 5C 62 A2 61 DA B3 72 65 05 26

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

  nonce:
  00000:   31 09 57 EF 71 31 44 33 F5 76 CC 9B 00 AD 93 54

  additional_data:
  00000:   17 03 03 40 11

  TLSInnerPlaintext:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000400:  17 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000410:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  00004000:  00

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 4011
encrypted_record:       EA6CB652C7CBE6B50560D0364DC94D90
                        2560DFE55D8B83C8AA919F5A1E5492E7
                        4CA5156F18BEC8EAB6971CAA2D2C1FF1
                        46EA5FEF5D62682601868FFCD2688F34
                        83899C31F6BA87538682E7F895F653C0
                        9BFE95ABF3EEDF7EBB261CCC593DFCB0
                        04F05119567148BB35F3C7B4F09713A6
                        247A047EF29B05F7720E375A6E3264F4
                        7922EAEBE3AA6D1E80832806D5F20E7C
                        56662A7128B191829597DB  [...]
                        6A5184907D9FF8D3FC0994A3C850DDBC
                        2D0420EB66EA177FCDD78F16246E2076
                        039C525604F79A007F472AC7A20A4574
                        1B9D96E38E565899D40A724B8A37FF68
                        702BF9A645D04962BBC9C66A35FFD219
                        A08A385FE4CDD0A1F3F080BECDF01E45
                        68C338FAD2C850DFEAA98A7F1B95ECA1
                        72BA7F7526E3BFF2EFF2395CE4561867
                        DC9DE8FD10F38BCA1E44B0207AF4CCE8
                        8155836330BC

TLSCiphertext:
  00000000:   17 03 03 40 11 EA 6C B6 52 C7 CB E6 B5 05 60 D0
  00000010:   36 4D C9 4D 90 25 60 DF E5 5D 8B 83 C8 AA 91 9F
  00000020:   5A 1E 54 92 E7 4C A5 15 6F 18 BE C8 EA B6 97 1C
  00000030:   AA 2D 2C 1F F1 46 EA 5F EF 5D 62 68 26 01 86 8F
  00000040:   FC D2 68 8F 34 83 89 9C 31 F6 BA 87 53 86 82 E7
  00000050:   F8 95 F6 53 C0 9B FE 95 AB F3 EE DF 7E BB 26 1C
  00000060:   CC 59 3D FC B0 04 F0 51 19 56 71 48 BB 35 F3 C7
  00000070:   B4 F0 97 13 A6 24 7A 04 7E F2 9B 05 F7 72 0E 37
  00000080:   5A 6E 32 64 F4 79 22 EA EB E3 AA 6D 1E 80 83 28
  00000090:   06 D5 F2 0E 7C 56 66 2A 71 28 B1 91 82 95 97 DB
  [...]
  00003F80:   6A 51 84 90 7D 9F F8 D3 FC 09 94 A3 C8 50 DD BC
  00003F90:   2D 04 20 EB 66 EA 17 7F CD D7 8F 16 24 6E 20 76
  00003FA0:   03 9C 52 56 04 F7 9A 00 7F 47 2A C7 A2 0A 45 74
  00003FB0:   1B 9D 96 E3 8E 56 58 99 D4 0A 72 4B 8A 37 FF 68
  00003FC0:   70 2B F9 A6 45 D0 49 62 BB C9 C6 6A 35 FF D2 19
```

```
  00003FD0:   A0 8A 38 5F E4 CD D0 A1 F3 F0 80 BE CD F0 1E 45
  00003FE0:   68 C3 38 FA D2 C8 50 DF EA A9 8A 7F 1B 95 EC A1
  00003FF0:   72 BA 7F 75 26 E3 BF F2 EF F2 39 5C E4 56 18 67
  00004000:   DC 9D E8 FD 10 F3 8B CA 1E 44 B0 20 7A F4 CC E8
  00004010:   81 55 83 63 30 BC

---------------------------Client-----------------------------

Application data:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Pad: 15360 bytes

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_ap, 1):
  00000:   D4 9A 57 15 49 E7 48 94 9F A2 4B 88 34 23 2C A8
  00010:   75 D3 7A 26 C4 BB 5C 62 A2 61 DA B3 72 65 05 26

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 01

  nonce:
  00000:   31 09 57 EF 71 31 44 33 F5 76 CC 9B 00 AD 93 55

  additional_data:
  00000:   17 03 03 40 11

  TLSInnerPlaintext:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000400:  17 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000410:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  00004000:  00

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 4011
encrypted_record:       0D486A03D03A020296EA0AD1D684A9F4
                        AE35824129141D3434CEE064FD5E966F
                        88D6E8903913417658E46C49B18BB0CC
                        B29B663D3F380A2CF9E5234BCD27F3A4
                        E12EBF3A3C69DB7661B08FC1685FADDE
                        50F68028A6E85EE12729D6F9CAD762FF
                        A6BAB5FC94AC65BAA36885DAF85C9B27
                        C68F9E97AB85ECFA760CDD22F9A8C0BA
```

```
                          6097D7960587CA708834516D9588592D
                          D1B8E05210BAA640FE6540  [...]
                          55A9C5A6557D35B8F1A9804BFA0F2789
                          3EDC6AA0350E9630AFF6C9B06C3CE01D
                          5BE51E87EBFFAC58230D074BE121F077
                          9D08F8177AFFFBB36DCEFDD0D0696873
                          A772B9A1DA73C681B0F8359EC1C74B6E
                          0452095C622C4C797F450CAA4F26975A
                          311F41F31C6A617747298CC052A6376F
                          A46191658FEE5BD8D7A998E7F12E8838
                          7365BAAD4BA490114733FC15A58148E6
                          186484821A94

TLSCiphertext:
  00000000:   17 03 03 40 11 0D 48 6A 03 D0 3A 02 02 96 EA 0A
  00000010:   D1 D6 84 A9 F4 AE 35 82 41 29 14 1D 34 34 CE E0
  00000020:   64 FD 5E 96 6F 88 D6 E8 90 39 13 41 76 58 E4 6C
  00000030:   49 B1 8B B0 CC B2 9B 66 3D 3F 38 0A 2C F9 E5 23
  00000040:   4B CD 27 F3 A4 E1 2E BF 3A 3C 69 DB 76 61 B0 8F
  00000050:   C1 68 5F AD DE 50 F6 80 28 A6 E8 5E E1 27 29 D6
  00000060:   F9 CA D7 62 FF A6 BA B5 FC 94 AC 65 BA A3 68 85
  00000070:   DA F8 5C 9B 27 C6 8F 9E 97 AB 85 EC FA 76 0C DD
  00000080:   22 F9 A8 C0 BA 60 97 D7 96 05 87 CA 70 88 34 51
  00000090:   6D 95 88 59 2D D1 B8 E0 52 10 BA A6 40 FE 65 40
  [...]
  00003F80:   55 A9 C5 A6 55 7D 35 B8 F1 A9 80 4B FA 0F 27 89
  00003F90:   3E DC 6A A0 35 0E 96 30 AF F6 C9 B0 6C 3C E0 1D
  00003FA0:   5B E5 1E 87 EB FF AC 58 23 0D 07 4B E1 21 F0 77
  00003FB0:   9D 08 F8 17 7A FF FB B3 6D CE FD D0 D0 69 68 73
  00003FC0:   A7 72 B9 A1 DA 73 C6 81 B0 F8 35 9E C1 C7 4B 6E
  00003FD0:   04 52 09 5C 62 2C 4C 79 7F 45 0C AA 4F 26 97 5A
  00003FE0:   31 1F 41 F3 1C 6A 61 77 47 29 8C C0 52 A6 37 6F
  00003FF0:   A4 61 91 65 8F EE 5B D8 D7 A9 98 E7 F1 2E 88 38
  00004000:   73 65 BA AD 4B A4 90 11 47 33 FC 15 A5 81 48 E6
  00004010:   18 64 84 82 1A 94

---------------------------Client-----------------------------

Application data:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Pad: 15360 bytes

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_ap, 8):
  00000:   B8 2D 78 25 D1 5F AE 18 A7 01 32 28 B3 1C B0 C5
  00010:   97 52 C6 40 9C 5F 78 99 EC C6 95 0F 74 63 C0 90
```

```
  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 08

  nonce:
  00000:   31 09 57 EF 71 31 44 33 F5 76 CC 9B 00 AD 93 5C

  additional_data:
  00000:   17 03 03 40 11

  TLSInnerPlaintext:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000400:  17 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000410:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  00004000:  00

Record layer message:
type:                    17
legacy_record_version:   0303
length:                  4011
encrypted_record:        F8B5732A300C8EF05FB712A2972F4DB8
                         4BE783A959090398E989516B6A54F333
                         331049283186BD1C42EFD98003A476A2
                         408EACE0D7047FB536979386C26B5523
                         F933A4F5BD7048B094EC5F5627EDFA98
                         99DE1AF8D9A493E481BA5DA0857BE15A
                         3F21CA01E22092159BAA770569CFBE54
                         F653BEFB4A8B32295DEFE992258F4581
                         257E936AF549E82D54EA6C09EF0D987B
                         F3A3E8453C1548CEF1C349  [...]
                         0EF4E88899AA3481AEDAE0E257449F80
                         A20CBDF070EC02211B6B9CBA9248B192
                         CF75C88A085DBFF77ABCFB1D82DAA421
                         1B487A48230350CBA4F338DD0BFD36D8
                         AAC5EE709456B7E317C78E7198FB7264
                         5B45EEFD3F93BF1C021F9E74A2ED2BCC
                         1CF5D367B553C7E7E9D80DD2447C7D13
                         D0345FEF2976696DFE579E5F71740C71
                         3124CFBAD66C7BB5BC21AAAE2F1E0860
                         5C248ADAF8BA

TLSCiphertext:
  00000000:   17 03 03 40 11 F8 B5 73 2A 30 0C 8E F0 5F B7 12
  00000010:   A2 97 2F 4D B8 4B E7 83 A9 59 09 03 98 E9 89 51
  00000020:   6B 6A 54 F3 33 33 10 49 28 31 86 BD 1C 42 EF D9
  00000030:   80 03 A4 76 A2 40 8E AC E0 D7 04 7F B5 36 97 93
```

```
  00000040:   86 C2 6B 55 23 F9 33 A4 F5 BD 70 48 B0 94 EC 5F
  00000050:   56 27 ED FA 98 99 DE 1A F8 D9 A4 93 E4 81 BA 5D
  00000060:   A0 85 7B E1 5A 3F 21 CA 01 E2 20 92 15 9B AA 77
  00000070:   05 69 CF BE 54 F6 53 BE FB 4A 8B 32 29 5D EF E9
  00000080:   92 25 8F 45 81 25 7E 93 6A F5 49 E8 2D 54 EA 6C
  00000090:   09 EF 0D 98 7B F3 A3 E8 45 3C 15 48 CE F1 C3 49
  [...]
  00003F80:   0E F4 E8 88 99 AA 34 81 AE DA E0 E2 57 44 9F 80
  00003F90:   A2 0C BD F0 70 EC 02 21 1B 6B 9C BA 92 48 B1 92
  00003FA0:   CF 75 C8 8A 08 5D BF F7 7A BC FB 1D 82 DA A4 21
  00003FB0:   1B 48 7A 48 23 03 50 CB A4 F3 38 DD 0B FD 36 D8
  00003FC0:   AA C5 EE 70 94 56 B7 E3 17 C7 8E 71 98 FB 72 64
  00003FD0:   5B 45 EE FD 3F 93 BF 1C 02 1F 9E 74 A2 ED 2B CC
  00003FE0:   1C F5 D3 67 B5 53 C7 E7 E9 D8 0D D2 44 7C 7D 13
  00003FF0:   D0 34 5F EF 29 76 69 6D FE 57 9E 5F 71 74 0C 71
  00004000:   31 24 CF BA D6 6C 7B B5 BC 21 AA AE 2F 1E 08 60
  00004010:   5C 24 8A DA F8 BA

---------------------------Client-----------------------------

Application data:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
Pad: 15360 bytes

Record payload protection:

  server_record_write_key = TLSTREE(server_write_key_ap, 9):
  00000:   B8 2D 78 25 D1 5F AE 18 A7 01 32 28 B3 1C B0 C5
  00010:   97 52 C6 40 9C 5F 78 99 EC C6 95 0F 74 63 C0 90

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 09

  nonce:
  00000:   31 09 57 EF 71 31 44 33 F5 76 CC 9B 00 AD 93 5D

  additional_data:
  00000:   17 03 03 40 11

  TLSInnerPlaintext:
  00000000:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  000003F0:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000400:  17 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  00000410:  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
  [...]
  00004000:  00
```

```
Record layer message:
type:                   17
legacy_record_version:  0303
length:                 4011
encrypted_record:       C1719B62D4F5E295AB8A4A2CBD6BBEF3
                        0F07297D96004EBABE315090247510A6
                        BEE6395676956B4249B16B52CE9FE171
                        B1F4693F48B3446D48A99B6224537FBB
                        9BC8BF54AEA688D21E39F17840DB9F33
                        632EA196922B7E15D6AE080F9F3B33F2
                        FABE63BB66E21C590785EFAEBE75BB1E
                        17C9E5F58A1B1D1101DE95F9BF346C62
                        1C63CABEB6D7245DB75F18DA495F129A
                        652CE6B7E0FE47FB210D6A  [...]
                        2AF9D515B26C3D8F37F9BF5F3A766D8B
                        03189A78605069179FB9CF9B1A449DC0
                        4F0FE37E67FDF9A0341B1F0D64AA2871
                        D4DFEF10EC7DFE7475CFE364BB4D9453
                        A9F176829887148F3E8C0EEE858F9C17
                        C0B753C145D13BD2A96B23822F73DC6C
                        FD623DE3CB70F8D507E436C20E393940
                        F3A36C913C0BCDFE672C903C5522AA41
                        0B318DD1268D035C59D3E11FF273B1D7
                        715E2FBF3ACA

TLSCiphertext:
  00000000:   17 03 03 40 11 C1 71 9B 62 D4 F5 E2 95 AB 8A 4A
  00000010:   2C BD 6B BE F3 0F 07 29 7D 96 00 4E BA BE 31 50
  00000020:   90 24 75 10 A6 BE E6 39 56 76 95 6B 42 49 B1 6B
  00000030:   52 CE 9F E1 71 B1 F4 69 3F 48 B3 44 6D 48 A9 9B
  00000040:   62 24 53 7F BB 9B C8 BF 54 AE A6 88 D2 1E 39 F1
  00000050:   78 40 DB 9F 33 63 2E A1 96 92 2B 7E 15 D6 AE 08
  00000060:   0F 9F 3B 33 F2 FA BE 63 BB 66 E2 1C 59 07 85 EF
  00000070:   AE BE 75 BB 1E 17 C9 E5 F5 8A 1B 1D 11 01 DE 95
  00000080:   F9 BF 34 6C 62 1C 63 CA BE B6 D7 24 5D B7 5F 18
  00000090:   DA 49 5F 12 9A 65 2C E6 B7 E0 FE 47 FB 21 0D 6A
  [...]
  00003F80:   2A F9 D5 15 B2 6C 3D 8F 37 F9 BF 5F 3A 76 6D 8B
  00003F90:   03 18 9A 78 60 50 69 17 9F B9 CF 9B 1A 44 9D C0
  00003FA0:   4F 0F E3 7E 67 FD F9 A0 34 1B 1F 0D 64 AA 28 71
  00003FB0:   D4 DF EF 10 EC 7D FE 74 75 CF E3 64 BB 4D 94 53
  00003FC0:   A9 F1 76 82 98 87 14 8F 3E 8C 0E EE 85 8F 9C 17
  00003FD0:   C0 B7 53 C1 45 D1 3B D2 A9 6B 23 82 2F 73 DC 6C
  00003FE0:   FD 62 3D E3 CB 70 F8 D5 07 E4 36 C2 0E 39 39 40
  00003FF0:   F3 A3 6C 91 3C 0B CD FE 67 2C 90 3C 55 22 AA 41
  00004000:   0B 31 8D D1 26 8D 03 5C 59 D3 E1 1F F2 73 B1 D7
  00004010:   71 5E 2F BF 3A CA

---------------------------Server---------------------------
```

```
Alert message:
level:                  01
description:            00

00000:   01 00

Record payload protection:

 server_record_write_key = TLSTREE(server_write_key_ap, 10):
  00000:   D3 CD 87 D5 68 74 07 82 39 78 34 4C 06 B9 28 A8
  00010:   58 98 B7 39 A3 1D 3D E5 FF 2B 78 8E F3 91 96 ED

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A

  nonce:
  00000:   2F E9 1F 71 18 35 40 26 31 7E 1A B4 D8 22 17 B2

  additional_data:
  00000:   17 03 03 00 13

  TLSInnerPlaintext:
  00000:   01 00 15

Record layer message:
type:                   17
legacy_record_version:  0303
length:                 0013
encrypted_record:       7CBC00AD5D29E301739394D31942C6A1
                        6658E9

TLSCiphertext:
  00000:   17 03 03 00 13 7C BC 00 AD 5D 29 E3 01 73 93 94
  00010:   D3 19 42 C6 A1 66 58 E9

---------------------------Client---------------------------
Alert message:
level:                  01
description:            00

00000:   01 00

Record payload protection:

  client_record_write_key = TLSTREE(client_write_key_ap, 10):
  00000:   D3 CD 87 D5 68 74 07 82 39 78 34 4C 06 B9 28 A8
  00010:   58 98 B7 39 A3 1D 3D E5 FF 2B 78 8E F3 91 96 ED

  seqnum:
  00000:   00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 0A
```

```
  nonce:
  00000:   31 09 57 EF 71 31 44 33 F5 76 CC 9B 00 AD 93 5E

  additional_data:
  00000:   17 03 03 00 13

  TLSInnerPlaintext:
  00000:   01 00 15

Record layer message:
type:                    17
legacy_record_version:   0303
length:                  0013
encrypted_record:        CB19F306C3641754BE4FC95390DF06F9
                         CD44AA

TLSCiphertext:
  00000:   17 03 03 00 13 CB 19 F3 06 C3 64 17 54 BE 4F C9
  00010:   53 90 DF 06 F9 CD 44 AA
```

Appendix B. Contributors

*Lilia Akhmetzyanova

CryptoPro

lah@cryptopro.ru

*Alexandr Sokolov

CryptoPro

sokolov@cryptopro.ru

*Vasily Nikolaev

CryptoPro

nikolaev@cryptopro.ru

Appendix C. Acknowledgments

Authors' Addresses

Stanislav Smyshlyaev (editor)
CryptoPro
18, Suschevsky val
Moscow

127018
Russian Federation

Phone: +7 (495) 995-48-20
Email: svs@cryptopro.ru

Evgeny Alekseev
CryptoPro
18, Suschevsky val
Moscow
127018
Russian Federation

Email: alekseev@cryptopro.ru

Ekaterina Griboedova
CryptoPro
18, Suschevsky val
Moscow
127018
Russian Federation

Email: griboedova.e.s@gmail.com

Alexandra Babueva
CryptoPro
18, Suschevsky val
Moscow
127018
Russian Federation

Email: babueva@cryptopro.ru

Lidiia Nikiforova
CryptoPro
18, Suschevsky val
Moscow
127018
Russian Federation

Email: nikiforova@cryptopro.ru