Independent Submission   P. Yang
Request for Comments: 8998   Ant Group
Category: Informational   March 2021
ISSN: 2070-1721

ShangMi (SM) Cipher Suites for TLS 1.3

Abstract

This document specifies how to use the ShangMi (SM) cryptographic algorithms with Transport Layer Security (TLS) protocol version 1.3.

The use of these algorithms with TLS 1.3 is not endorsed by the IETF. The SM algorithms are becoming mandatory in China, so this document provides a description of how to use the SM algorithms with TLS 1.3 and specifies a profile of TLS 1.3 so that implementers can produce interworking implementations.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document describes two new cipher suites, a signature algorithm and a key exchange mechanism for the Transport Layer Security (TLS) protocol version 1.3 (TLS 1.3) ([RFC8446]). These all utilize several ShangMi (SM) cryptographic algorithms to fulfill the authentication and confidentiality requirements of TLS 1.3. The new cipher suites are as follows (see also Section 2):

```
CipherSuite TLS_SM4_GCM_SM3 = { 0x00, 0xC6 };
CipherSuite TLS_SM4_CCM_SM3 = { 0x00, 0xC7 };
```

For a more detailed introduction to SM cryptographic algorithms, please see Section 1.1. These cipher suites follow the TLS 1.3 requirements. Specifically, all the cipher suites use SM4 in either Galois/Counter (GCM) mode or Counter with CBC-MAC (CCM) mode to meet the needs of TLS 1.3 to have an encryption algorithm that is Authenticated Encryption with Associated Data (AEAD) capable. The key exchange mechanism utilizes Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) over the SM2 elliptic curve, and the signature algorithm combines the SM3 hash function and the SM2 elliptic curve signature scheme.

For details about how these mechanisms negotiate shared encryption keys, authenticate the peer(s), and protect the record structure, please see Section 3.

The cipher suites, signature algorithm, and key exchange mechanism defined in this document are not recommended by the IETF. The SM algorithms are becoming mandatory in China, so this document provides a description of how to use them with TLS 1.3 and specifies a profile of TLS 1.3 so that implementers can produce interworking implementations.

1.1. The SM Algorithms

Several different SM cryptographic algorithms are used to integrate with TLS 1.3, including SM2 for authentication, SM4 for encryption, and SM3 as the hash function.

SM2 is a set of cryptographic algorithms based on elliptic curve cryptography, including a digital signature, public key encryption and key exchange scheme. In this document, only the SM2 digital signature algorithm and basic key exchange scheme are involved, which have already been added to ISO/IEC 14888-3:2018 [ISO-SM2] (as well as to [GBT.32918.2-2016]). SM4 is a block cipher defined in [GBT.32907-2016] and now is being standardized by ISO to ISO/IEC 18033-3:2010 [ISO-SM4]. SM3 is a hash function that produces an output of 256 bits. SM3 has already been accepted by ISO in ISO/IEC 10118-3:2018 [ISO-SM3] and has also been described by [GBT.32905-2016].

1.2. Terminology

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.

Although this document is not an IETF Standards Track publication, it adopts the conventions for normative language to provide clarity of instruction to the implementer and to indicate requirement levels for compliant TLS 1.3 implementations.

2. Algorithm Identifiers

The cipher suites defined here have the following identifiers:

```
CipherSuite TLS_SM4_GCM_SM3 = { 0x00, 0xC6 };
CipherSuite TLS_SM4_CCM_SM3 = { 0x00, 0xC7 };
```

To accomplish a TLS 1.3 handshake, additional objects have been introduced along with the cipher suites as follows:

* The combination of the SM2 signature algorithm and SM3 hash function used in the Signature Algorithm extension is defined in Appendix B.3.1.3 of [RFC8446]:

  ```
  SignatureScheme sm2sig_sm3 = { 0x0708 };
  ```

* The SM2 elliptic curve ID used in the Supported Groups extension is defined in Appendix B.3.1.4 of [RFC8446]:

  ```
  NamedGroup curveSM2 = { 41 };
  ```

3. Algorithm Definitions

3.1. TLS Versions

The new cipher suites defined in this document are only applicable to

TLS 1.3. Implementations of this document MUST NOT apply these cipher suites to any older versions of TLS.

3.2. Authentication

3.2.1. SM2 Signature Scheme

The Chinese government requires the use of the SM2 signature algorithm. This section specifies the use of the SM2 signature algorithm as the authentication method for a TLS 1.3 handshake.

The SM2 signature algorithm is defined in [ISO-SM2]. The SM2 signature algorithm is based on elliptic curves. The SM2 signature algorithm uses a fixed elliptic curve parameter set defined in [GBT.32918.5-2017]. This curve is named "curveSM2" and has been assigned the value 41, as shown in Section 2. Unlike other public key algorithms based on elliptic curve cryptography like the Elliptic Curve Digital Signature Algorithm (ECDSA), SM2 MUST NOT select other elliptic curves. But it is acceptable to write test cases that use other elliptic curve parameter sets for SM2; see Annex F.14 of [ISO-SM2] as a reference.

Implementations of the signature scheme and key exchange mechanism defined in this document MUST conform to what [GBT.32918.5-2017] requires; that is to say, the only valid elliptic curve parameter set for the SM2 signature algorithm (a.k.a. curveSM2) is defined as follows:

curveSM2: A prime field of 256 bits.

y^(2) = x^(3) + ax + b

```
   p  = FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF
        FFFFFFFF 00000000 FFFFFFFF FFFFFFFF
   a  = FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF
        FFFFFFFF 00000000 FFFFFFFF FFFFFFFC
   b  = 28E9FA9E 9D9F5E34 4D5A9E4B CF6509A7
        F39789F5 15AB8F92 DDBCBD41 4D940E93
   n  = FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF
        7203DF6B 21C6052B 53BBF409 39D54123
   Gx = 32C4AE2C 1F198119 5F990446 6A39C994
        8FE30BBF F2660BE1 715A4589 334C74C7
   Gy = BC3736A2 F4F6779C 59BDCEE3 6B692153
        D0A9877C C62A4740 02DF32E5 2139F0A0
```

The SM2 signature algorithm requests an identifier value when generating or verifying a signature. In all uses except when a client of a server needs to verify a peer's SM2 certificate in the Certificate message, an implementation of this document MUST use the following ASCII string value as the SM2 identifier when doing a TLS 1.3 key exchange:

TLSv1.3+GM+Cipher+Suite

If either a client or a server needs to verify the peer's SM2 certificate contained in the Certificate message, then the following

ASCII string value MUST be used as the SM2 identifier according to [GMT.0009-2012]:

```
1234567812345678
```

Expressed as octets, this is:

```
0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38,
0x31, 0x32, 0x33, 0x34, 0x35, 0x36, 0x37, 0x38
```

In practice, the SM2 identifier used in a certificate signature depends on the certificate authority (CA) who signs that certificate. CAs may choose values other than the ones mentioned above. Implementations of this document SHOULD confirm this information by themselves.

3.3. Key Exchange

3.3.1. Hello Messages

The use of the algorithms defined by this document is negotiated during the TLS handshake with information exchanged in the Hello messages.

3.3.1.1. ClientHello

To use the cipher suites defined by this document, a TLS 1.3 client includes the new cipher suites in the "cipher_suites" array of the ClientHello structure defined in Section 4.1.2 of [RFC8446].

Other requirements of this TLS 1.3 profile on the extensions of ClientHello message are as follows:

* For the supported_groups extension, "curveSM2" MUST be included.

* For the signature_algorithms extension, "sm2sig_sm3" MUST be included.

* For the signature_algorithms_cert extension (if present), "sm2sig_sm3" MUST be included.

* For the key_share extension, a KeyShareEntry for the "curveSM2" group MUST be included.

3.3.1.2. ServerHello

If a TLS 1.3 server receives a ClientHello message containing the algorithms defined in this document, it MAY choose to use them. If so, then the server MUST put one of the new cipher suites defined in this document into its ServerHello's "cipher_suites" array and eventually send it to the client side.

A TLS 1.3 server's choice of what cipher suite to use depends on the configuration of the server. For instance, a TLS 1.3 server may or not be configured to include the new cipher suites defined in this document. Typical TLS 1.3 server applications also provide a mechanism that configures the cipher suite preference on the server side. If a server is not configured to use the cipher suites defined in this document, it SHOULD choose another cipher suite in the list that the TLS 1.3 client provides; otherwise, the server MUST abort the handshake with an "illegal_parameter" alert.

The following extension MUST conform to the new requirements:

* For the key_share extension, a KeyShareEntry with SM2-related values MUST be added if the server wants to conform to this profile.

3.3.2. CertificateRequest

If a CertificateRequest message is sent by the server to require the client to send its certificate for authentication purposes, for conformance to this profile, the following is REQUIRED:

* The only valid signature algorithm present in "signature_algorithms" extension MUST be "sm2sig_sm3". That is to say, if the server chooses to conform to this profile, the signature algorithm for the client's certificate MUST use the SM2/SM3 procedure specified by this document.

3.3.3. Certificate

When a server sends the Certificate message containing the server certificate to the client side, several new rules are added that will affect the certificate selection:

* The public key in the certificate MUST be a valid SM2 public key.

* The signature algorithm used by the CA to sign the current certificate MUST be "sm2sig_sm3".

* The certificate MUST be capable of signing; e.g., the digitalSignature bit of X.509's Key Usage extension is set.

3.3.4. CertificateVerify

In the CertificateVerify message, the signature algorithm MUST be "sm2sig_sm3", indicating that the hash function MUST be SM3 and the signature algorithm MUST be SM2.

3.4. Key Scheduling

As described in Section 1.1, SM2 is actually a set of cryptographic algorithms, including one key exchange protocol that defines methods such as key derivation function, etc. This document does not define an SM2 key exchange protocol, and an SM2 key exchange protocol SHALL NOT be used in the key exchange steps defined in Section 3.3. Implementations of this document MUST always conform to what TLS 1.3 [RFC8446] and its successors require regarding the key derivation and related methods.

3.5. Cipher

The new cipher suites introduced in this document add two new AEAD encryption algorithms, AEAD_SM4_GCM and AEAD_SM4_CCM, which stand for SM4 cipher in Galois/Counter mode and SM4 cipher [GBT.32907-2016] in Counter with CBC-MAC mode, respectively. The hash function for both cipher suites is SM3 ([ISO-SM3]).

This section defines the AEAD_SM4_GCM and AEAD_SM4_CCM AEAD algorithms in a style similar to what [RFC5116] used to define AEAD ciphers based on the AES cipher.

3.5.1. AEAD_SM4_GCM

The AEAD_SM4_GCM authenticated encryption algorithm works as specified in [GCM], using SM4 as the block cipher, by providing the key, nonce, plaintext, and associated data to that mode of operation. An authentication tag conforming to the requirements of TLS 1.3 as specified in Section 5.2 of [RFC8446] MUST be constructed using the details in the TLS record header. The additional data input that forms the authentication tag MUST be the TLS record header. The AEAD_SM4_GCM ciphertext is formed by appending the authentication tag provided as an output to the GCM encryption operation to the ciphertext that is output by that operation. AEAD_SM4_GCM has four inputs: an SM4 key, an initialization vector (IV), a plaintext content, and optional additional authenticated data (AAD). AEAD_SM4_GCM generates two outputs: a ciphertext and message authentication code (also called an authentication tag). To have a common set of terms for AEAD_SM4_GCM and AEAD_SM4_CCM, the AEAD_SM4_GCM IV is referred to as a nonce in the remainder of this document. A simple test vector of AEAD_SM4_GCM and AEAD_SM4_CCM is given in Appendix A of this document.

The nonce is generated by the party performing the authenticated encryption operation. Within the scope of any authenticated encryption key, the nonce value MUST be unique. That is, the set of nonce values used with any given key MUST NOT contain any duplicates. Using the same nonce for two different messages encrypted with the same key destroys the security properties of GCM mode. To generate the nonce, implementations of this document MUST conform to TLS 1.3 (see [RFC8446], Section 5.3).

The input and output lengths are as follows:

The SM4 key length is 16 octets.

The max plaintext length is 2^(36) - 31 octets.

The max AAD length is 2^(61) - 1 octets.

The nonce length is 12 octets.

The authentication tag length is 16 octets.

The max ciphertext length is 2^(36) - 15 octets.

A security analysis of GCM is available in [MV04].

3.5.2. AEAD_SM4_CCM

The AEAD_SM4_CCM authenticated encryption algorithm works as specified in [CCM] using SM4 as the block cipher. AEAD_SM4_CCM has four inputs: an SM4 key, a nonce, a plaintext, and optional additional authenticated data (AAD). AEAD_SM4_CCM generates two outputs: a ciphertext and a message authentication code (also called an authentication tag). The formatting and counter generation functions are as specified in Appendix A of [CCM], and the values of the parameters identified in that appendix are as follows:

The nonce length n is 12.

The tag length t is 16.

The value of q is 3.

An authentication tag is also used in AEAD_SM4_CCM. The generation of the authentication tag MUST conform to TLS 1.3 (See [RFC8446], Section 5.2). The AEAD_SM4_CCM ciphertext is formed by appending the authentication tag provided as an output to the CCM encryption operation to the ciphertext that is output by that operation. The input and output lengths are as follows:

The SM4 key length is 16 octets.

The max plaintext length is 2^(24) - 1 octets.

The max AAD length is 2^(64) - 1 octets.

The max ciphertext length is 2^(24) + 15 octets.

To generate the nonce, implementations of this document MUST conform to TLS 1.3 (see [RFC8446], Section 5.3).

A security analysis of CCM is available in [J02].

4. IANA Considerations

IANA has assigned the values {0x00,0xC6} and {0x00,0xC7} with the names "TLS_SM4_GCM_SM3" and "TLS_SM4_CCM_SM3" to the "TLS Cipher Suites" registry with this document as reference:

Value	Description	DTLS-OK	Recommended	Reference
0x00,0xC6	TLS_SM4_GCM_SM3	No	No	RFC 8998
0x00,0xC7	TLS_SM4_CCM_SM3	No	No	RFC 8998

Table 1

IANA has assigned the value 0x0708 with the name "sm2sig_sm3" to the "TLS SignatureScheme" registry:

Value	Description	Recommended	Reference
0x0708	sm2sig_sm3	No	RFC 8998

Table 2

IANA has assigned the value 41 with the name "curveSM2" to the "TLS Supported Groups" registry:

Value	Description	DTLS-OK	Recommended	Reference
41	curveSM2	No	No	RFC 8998

Table 3

5. Security Considerations

At the time of writing, there are no known weak keys for SM cryptographic algorithms SM2, SM3 and SM4, and no security issues have been found for these algorithms.

A security analysis of GCM is available in [MV04].

A security analysis of CCM is available in [J02].

6. References

6.1. Normative References

[CCM] Dworkin, M., "Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality", Special Publication 800-38C, DOI 10.6028/NIST.SP.800-38C, May 2004, <http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C.pdf>.

[GCM] Dworkin, M., "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC", Special Publication 800-38D, DOI 10.6028/NIST.SP.800-38D, November 2007, <http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf>.

[ISO-SM2] International Organization for Standardization, "IT Security techniques -- Digital signatures with appendix -- Part 3: Discrete logarithm based mechanisms", ISO/IEC 14888-3:2018, November 2018, <https://www.iso.org/standard/76382.html>.

[ISO-SM3] International Organization for Standardization, "IT Security techniques -- Hash-functions -- Part 3: Dedicated hash-functions", ISO/IEC 10118-3:2018, October 2018,

<https://www.iso.org/standard/67116.html>.

[ISO-SM4] International Organization for Standardization, "Information technology -- Security techniques -- Encryption algorithms -- Part 3: Block ciphers", ISO/IEC 18033-3:2010, December 2010, <https://www.iso.org/standard/54531.html>.

[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>.

[RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", RFC 5116, DOI 10.17487/RFC5116, January 2008, <https://www.rfc-editor.org/info/rfc5116>.

[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>.

6.2. Informative References

[GBT.32905-2016] Standardization Administration of China, "Information security technology --- SM3 cryptographic hash algorithm", GB/T 32905-2016, March 2017, <http://www.gmbz.org.cn/upload/2018-07-24/1532401392982079739.pdf>.

[GBT.32907-2016] Standardization Administration of the People's Republic of China, "Information security technology -- SM4 block cipher algorithm", GB/T 32907-2016, March 2017, <http://www.gmbz.org.cn/upload/2018-04-04/1522788048733065051.pdf>.

[GBT.32918.2-2016] Standardization Administration of the People's Republic of China, "Information security technology --- Public key cryptographic algorithm SM2 based on elliptic curves --- Part 2: Digital signature algorithm", GB/T 32918.2-2016, March 2017, <http://www.gmbz.org.cn/upload/2018-07-24/1532401673138056311.pdf>.

[GBT.32918.5-2017] Standardization Administration of the People's Republic of China, "Information security technology --- Public key cryptographic algorithm SM2 based on elliptic curves --- Part 5: Parameter definition", GB/T 32918.5-2017, December 2017, <http://www.gmbz.org.cn/upload/2018-07-24/1532401863206085511.pdf>.

[GMT.0009-2012]
State Cryptography Administration, "SM2 cryptography algorithm application specification", GM/T 0009-2012, November 2012, <http://www.gmbz.org.cn/main/viewfile/2018011001400692565.html>.

[J02] Jonsson, J., "On the Security of CTR + CBC-MAC", DOI 10.1007/3-540-36492-7_7, February 2003, <https://link.springer.com/chapter/10.1007%2F3-540-36492-7_7>.

[MV04] McGrew, D. and J. Viega, "The Security and Performance of the Galois/Counter Mode of Operation", DOI 10.1007/978-3-540-30556-9_27, December 2004, <http://eprint.iacr.org/2004/193>.

Appendix A. Test Vectors

All values are in hexadecimal and are in network byte order (big endian).

A.1. SM4-GCM Test Vectors

```
Initialization Vector:   00001234567800000000ABCD
Key:                     0123456789ABCDEFFEDCBA9876543210
Plaintext:               AAAAAAAAAAAAAAAABBBBBBBBBBBBBBBB
                         CCCCCCCCCCCCCCCCDDDDDDDDDDDDDDDD
                         EEEEEEEEEEEEEEEEFFFFFFFFFFFFFFFF
                         EEEEEEEEEEEEEEEEAAAAAAAAAAAAAAAA
Associated Data:         FEEDFACEDEADBEEFFEEDFACEDEADBEEFABADDAD2
CipherText:              17F399F08C67D5EE19D0DC9969C4BB7D
                         5FD46FD3756489069157B282BB200735
                         D82710CA5C22F0CCFA7CBF93D496AC15
                         A56834CBCF98C397B4024A2691233B8D
Authentication Tag:      83DE3541E4C2B58177E065A9BF7B62EC
```

A.2. SM4-CCM Test Vectors

```
Initialization Vector:   00001234567800000000ABCD
Key:                     0123456789ABCDEFFEDCBA9876543210
Plaintext:               AAAAAAAAAAAAAAAABBBBBBBBBBBBBBBB
                         CCCCCCCCCCCCCCCCDDDDDDDDDDDDDDDD
                         EEEEEEEEEEEEEEEEFFFFFFFFFFFFFFFF
                         EEEEEEEEEEEEEEEEAAAAAAAAAAAAAAAA
Associated Data:         FEEDFACEDEADBEEFFEEDFACEDEADBEEFABADDAD2
CipherText:              48AF93501FA62ADBCD414CCE6034D895
                         DDA1BF8F132F042098661572E7483094
                         FD12E518CE062C98ACEE28D95DF4416B
                         ED31A2F04476C18BB40C84A74B97DC5B
Authentication Tag:      16842D4FA186F56AB33256971FA110F4
```

Contributors

Qin Long
Ant Group

Email: zhuolong.lq@antfin.com

Kepeng Li
Ant Group

Email: kepeng.lkp@antfin.com

Ke Zeng
Ant Group

Email: william.zk@antfin.com

Han Xiao
Ant Group

Email: han.xiao@antfin.com

Zhi Guan
Peking University

Email: guan@pku.edu.cn

Author's Address

Paul Yang
Ant Group
No. 77 Xueyuan Road
Hangzhou
310000
China

Phone: +86-571-2688-8888
Email: kaishen.yy@antfin.com