Network Working Group
Internet-Draft
Intended status: Informational
Expires: 3 October 2026

A. Becker

M. Jenkins
National Security Agency
1 April 2026

Commercial National Security Algorithm (CNSA) Suite 2.0 Profile for TLS 1.3

draft-becker-cnsa2-tls-profile-03

Abstract

This document defines a base profile of TLS 1.3 which is compliant with the US Commercial National Security Algorithm (CNSA) 2.0 Suite, a cybersecurity advisory published by the United States Government which outlines quantum-resistant cryptographic algorithm policy for US national security applications.

This profile applies to the capabilities, configuration, and operation of all components of US National Security Systems that employ TLS 1.3. It is also appropriate for all other US Government systems that process high-value information.

This memo is not an IETF standard, and has not been shown to have IETF community consensus. This profile is made publicly available for use by developers and operators of these and any other system deployments.

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 3 October 2026.

Table of Contents

1. Introduction 3
2. Terminology 3
3. The Commercial National Security Algorithm Suite 2.0 4
4. CNSA 2.0 Suite 4
5. Compliance and Interoperability 5
6. TLS Version 5
"supported_versions" Extension 5
7. Key Exchange Messages 5
7.1. Cipher Suite Negotiation 5
7.2. KEM Requirements 6
7.2.1. "supported_groups" Extension 6
7.2.2. "key_share" Extension 6
8. Authentication Messages 7
8.1. "signature_algorithms" Extension 7
8.2. "signature_algorithms_cert" Extension 7
8.3. CertificateRequest Message 8
8.4. Certificate Selection 8
8.5. CertificateVerify Message 8
9. External Pre-Shared Keys 9
10. Resumption 9
11. Certificate Status 10
12. "early_data" Extension 10
13. DTLS 1.3 10
14. Security Considerations 10
15. IANA Considerations 11
16. References 11
16.1. Normative References 11
16.2. Informative References 12
Authors' Addresses 13

1. Introduction

This document specifies a profile of TLS 1.3 [RFC8446] to comply with the National Security Agency's (NSA) Commercial National Security Algorithm (CNSA) 2.0 Suite [cnsafaq]. This profile applies to the capabilities, configuration, and operation of all components of US National Security Systems (NSS) [SP80059] that employ TLS 1.3. This profile is also appropriate for all other US Government systems that process high-value information, and is made publicly available for use by developers and operators of these and any other system deployments.

This document does not define any cryptographic algorithm, and does not specify how to use any cryptographic algorithm not currently supported by TLS 1.3; instead, it profiles CNSA 2.0-compliant conventions for TLS 1.3, and uses algorithms already specified for use in TLS 1.3 that are also allowed by the CNSA Suite 2.0. This document applies to all CNSA Suite solutions that make use of TLS 1.3.

The reader is assumed to have familiarity with [I-D.ietf-tls-8773bis].

This memo is not an IETF standard, and has not been shown to have IETF community consensus.

This document obsoletes the CSNA 1.0 guidance in [RFC9151] for TLS 1.3. Servers and clients that also support TLS 1.2 (such as, during transition) MUST continue to comply with that guidance.

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. Normative language does not apply beyond the scope of this profile.

This is a profile of TLS 1.3 [RFC8446]. Therefore, the requirements language in this memo may be different than that found in the underlying standards.

All references to "CNSA" in this document refer to CNSA 2.0 [cnsafaq], unless stated otherwise.

3. The Commercial National Security Algorithm Suite 2.0

The CNSA Suite is the set of approved commercial algorithms that can be used by vendors and IT users to meet cybersecurity and interoperability requirements for NSS. The initial suite of CNSA Suite algorithms, Suite B, established a baseline for use of commercial algorithms to protect classified information. The following suite, CNSA 1.0, served as a bridge between the original set and a fully quantum-resistant cryptographic capability. The current suite, CNSA 2.0, seeks to provide fully quantum-resistant protection [cnsafaq].

This profile uses CNSA 2.0 compliant algorithms: ML-DSA-87 [FIPS204] for signing, and ML-KEM-1024 [FIPS203] for key establishment. ML-DSA-87 and ML-KEM-1024 together with SHA-384, SHA-512, AES-256, LMS, and XMSS comprise the CNSA Suite 2.0.

The NSA is authoring a set of RFCs, including this one, to provide updated guidance for using the CNSA 2.0 Suite in certain IETF protocols. These RFCs can be used in conjunction with other RFCs and cryptographic guidance (e.g., NIST Special Publications) to properly protect Internet traffic and data-at-rest for US Government NSS.

4. CNSA 2.0 Suite

CNSA requires the following algorithms for use in TLS 1.3:

Encryption:

> AES [AES] with 256-bit keys, operating in Galois Counter Mode (GCM) [GCM] ({0x13, 0x02} Appendix B.4 [RFC8446])

Key Establishment:

> ML-KEM-1024 [FIPS203] ({0x0202} MLKEM1024)

Digital Signature

> ML-DSA-87 [FIPS204] ({0x0906} MLDSA87)

The ML-DSA algorithm incorporates an internal hashing function, so there is no need to apply a hashing algorithm before signing. Where an application or implementation makes it more efficient to perform hashing externally, the external-亮 mechanism described in Step 6 of Algorithm 7 of [FIPS204] and Section 8 of [RFC9881] MAY be used. HashML-DSA is not permitted.

CNSA requires the use of SHA-384 [SHS] for the HMAC-based Key Derivation Function (HKDF) in TLS 1.3.

5. Compliance and Interoperability

In some situations, clients or servers configured for CNSA compliance also have to interoperate with non-compliant clients and servers. Compliant clients and servers MUST be configured such that CNSA compliance is preferred even if it is not intended or expected. CNSA-compliant implementations MUST present CNSA compliant algorithms first in the appropriate extensions, and CNSA-compliant algorithms MUST be used when supported by the peer. If any aspect of CNSA compliance is not achieved, the connection is not CNSA compliant. Any connection negotiated to TLS version 1.2 or lower cannot be CNSA compliant.

If interoperability with non-CNSA-compliant clients or servers is not intended, then the session MUST fail if any aspect of CNSA compliance is not achieved.

6. TLS Version

CNSA TLS clients and servers MUST negotiate TLS version 1.3 [RFC8446] when establishing a CNSA-compliant connection. CNSA TLS clients and CNSA TLS servers MUST NOT negotiate TLS versions prior to TLS 1.3 in a CNSA-compliant mode.

"supported_versions" Extension

A CNSA TLS client connecting to a CNSA TLS server MUST include the "supported_versions" extension in the ClientHello (Section 4.2.1 of [RFC8446]) and list TLS 1.3 version value (0x0304) first in the extension.

7. Key Exchange Messages

This section details requirements for CNSA 2.0-compliant algorithm negotiation in TLS 1.3.

7.1. Cipher Suite Negotiation

A CNSA TLS client MUST offer the following cipher suite in the ClientHello:

TLS_AES_256_GCM_SHA384 {0x13, 0x02} (Appendix B.4 [RFC8446])

This CNSA cipher suite MUST be the first (most preferred) cipher suite in the ClientHello message and in the extensions.

A CNSA TLS server MUST select this cipher suite if offered by the client.

Any cipher suite other than TLS_AES_256_GCM_SHA384 offered by the client MUST NOT be accepted by a CNSA TLS server establishing a CNSA-compliant connection.

7.2. KEM Requirements

CNSA 2.0 requires the use of ML-KEM-1024 for key establishment in TLS.

7.2.1. "supported_groups" Extension

A CNSA TLS client MUST offer the following value in named_group_list of the "supported_groups" extension (Section 4.2.7 [RFC8446]) of the ClientHello:

ML-KEM-1024 {0x0202}

ML-KEM-1024 MUST be the first (most preferred) item in named_group_list.

A CNSA TLS server MUST select ML-KEM-1024 if it is included in the "supported_groups" extension sent by a CNSA TLS client in the ClientHello.

Any algorithm other than ML-KEM-1024 offered by the client MUST NOT be accepted by a CNSA TLS server establishing a CNSA-compliant connection.

7.2.2. "key_share" Extension

The "key_share" extension in a CNSA TLS client ClientHello MUST contain the ML-KEM-1024 public key .

The ML-KEM-1024 public key must be the first (most preferred) key in the list of key shares.

A CNSA TLS server MUST select the ML-KEM-1024 key share for key establishment, and the "key_share" extension in a CNSA TLS server ServerHello MUST contain the ML-KEM-1024 ciphertext associated to the public key provided in the ClientHello .

8. Authentication Messages

A CNSA TLS client MUST require the server to authenticate itself. In all cases, a CNSA TLS client MUST authenticate the server using X.509 certificates; PSK-based authentication MAY be used in addition to certificate-based authentication as detailed in Section 7.

A CNSA TLS server MAY also authenticate the client as needed by the specific application. If mutual authentication is desired, X.509 certificates MUST be used in all cases.

8.1. "signature_algorithms" Extension

A CNSA TLS client MUST offer the following value in the "signature_algorithms" extension of the ClientHello:

ML-DSA-87 {0x0906}

The ML-DSA-87 algorithm must be the first (most preferred) value in the extension.

Any signature algorithm values offered other than ML-DSA-87 MUST NOT be accepted by a CNSA TLS server establishing a CNSA-compliant connection.

8.2. "signature_algorithms_cert" Extension

Per [RFC8446], the "signature_algorithms_cert" extension applies to signatures in certificates, while the "signature_algorithms" extension (Section 6.1) applies to signatures in CertificateVerify messages.

The "signature_algorithms_cert" extension is not required for a CNSA-compliant connection, as ML-DSA-87 is the only allowed signature algorithm that can be used for both signatures in the certificate and in the CertificateVerify message under CNSA compliance.

However, if the "signature_algorithms_cert" extension is included, the CNSA TLS client MUST offer the following value first in the "supported_signature_algorithms" field of the extension:

ML-DSA-87 {0x0906}

Any signature algorithm offered other than ML-DSA-87 MUST NOT be accepted by a CNSA TLS server establishing a CNSA-compliant connection.

8.3. CertificateRequest Message

A CNSA TLS server MAY authenticate the client as needed by the specific application.

If the CNSA TLS server requires mutual authentication, it MUST authenticate the client using X.509 certificates.

The "signature_algorithms" extension sent by the CNSA TLS server in the CertificateRequest message MUST include:

ML-DSA-87 {0x0906}

The "signature_algorithms_cert" extension is not required, but if it is included by the CNSA TLS server in the CertificateRequest message, then it MUST include:

ML-DSA-87 {0x0906}

ML-DSA-87 MUST be the first (most preferred) algorithm in the "signature_algorithms" and "signature_algorithms_cert" extensions.

8.4. Certificate Selection

Certificates sent by a CNSA TLS server (and CNSA TLS client, when required) MUST be signed by ML-DSA-87, and MUST be compliant with the CNSA Certificate and Certificate Revocation List Profile [I-D.jenkins-cnsa2-pkix-profile]. If the client cannot validate the server certificate for any reason, it must abort the handshake.

If a CNSA TLS server sends a CertificateRequest message to the client, the client MUST respond with a valid X.509 certificate suitable for authenticating the client. If the CNSA TLS client sends an empty Certificate message, the CNSA TLS server MUST abort the handshake (Section 4.4.2.4 of [RFC8446]). Also, if some aspect of the certificate chain was unacceptable (e.g., it was not signed by a known, trusted CA), the server MUST abort the handshake.

8.5. CertificateVerify Message

A CNSA TLS client or CNSA TLS server MUST use ML-DSA-87 when sending the CertificateVerify message. CNSA TLS clients or servers MUST accept only ML-DSA-87 in the CertificateVerify message when establishing a CNSA-compliant connection.

9. External Pre-Shared Keys

RFC 8773 [I-D.ietf-tls-8773bis] specifies an extension that allows an external PSK to be used for authentication in addition to (and not in lieu of) certificate-based authentication during the initial handshake. The PSK also contributes to the TLS 1.3 key schedule (Section 7.1, [RFC8446]). PSK-only authentication is not compliant beyond resumption as covered in Section 10.

A CNSA TLS client that supports RFC 8773 MUST include the "tls_cert_with_extern_psk" extension (Section 4, [I-D.ietf-tls-8773bis]) in the ClientHello message if it desires an external PSK to be used for authentication with certificate-based authentication. This extension is accompanied by the "key_share", "psk_key_exchange_modes", and "pre_shared_key" extensions defined in (Section 4.2.9 and 4.2.11, respectively, of [RFC8446]).

The "psk_key_exchange_modes" extension MUST NOT include psk_ke mode. The "psk_key_exchange_modes" extension MUST be psk_dhe_key using ML-KEM-1024.

PSKs shall be at least 256 bits in length and generated from a NIST approved random bit generator that supports 256-bits of entropy [SP800-90c].

If the CNSA TLS server recognizes the "tls_cert_with_extern_psk" extension and possesses at least one of the external PSKs offered by the client in the "pre_shared_key" extension in the ClientHello, then the CNSA TLS server MUST select a CNSA compliant PSK and respond with the "tls_cert_with_extern_psk", "key_share", and "pre_shared_key" extensions in the ServerHello message (Section 4, [I-D.ietf-tls-8773bis]).

10. Resumption

A CNSA TLS server MAY send a CNSA TLS client a NewSessionTicket message to enable resumption. A CNSA TLS client MUST request "psk_dhe_ke" via the "psk_key_exchange_modes" extension in the ClientHello to resume a session. The "supported_groups" and "key_share" extensions MUST comply with those detailed in Section 5.2.1 and Section 5.2.2 of this document, respectively.

11. Certificate Status

A CNSA TLS server (and CNSA TLS client, if client authentication is required) MUST provide certificate status information either via a Certificate Revocation List (CRL) distribution points or by the Online Certificate Status Protocol (OCSP) (with or without stapling). For details on CNSA requirements for these methods, see [I-D.jenkins-cnsa2-pkix-profile].

12. "early_data" Extension

A CNSA TLS client or server MUST NOT include the "early_data" extension.

13. DTLS 1.3

CNSA DTLS clients and servers MUST negotiate DTLS version 1.3, [RFC9147] when establishing a CNSA-compliant connection. DTLS 1.3, 0xfefc, MUST be listed first in the "supported _versions" extension sent by the CNSA DTLS client. CNSA DTLS clients and CNSA DTLS servers MUST NOT negotiate DTLS versions prior to DTLS 1.3 in a CNSA-compliant mode.

A CNSA DTLS client MUST offer the cipher suite TLS_AES_256_GCM_SHA384 {0x13, 0x02} as the first (most preferred) cipher suite in the ClientHello message.

For key establishment, CNSA DTLS clients and servers MUST negotiate use of ML-KEM-1024 {0x0202} .

For authentication, CNSA DTLS clients and servers MUST negotiate use of ML-DSA-87 as the signature algorithm used in the "signature_algorithms" extension (and the "signature_algorithms_cert" extension, if present).

DTLS implementations MUST be consistent with the requirements of TLS defined in this document, except as specified in this section.

14. Security Considerations

Most of the security considerations for this document are described in [RFC8446], [I-D.ietf-tls-8773bis].

As noted in TLS version 1.3 [RFC8446], TLS does not provide inherent replay protections for early data. For this reason, this profile forbids the use of early data.

15. IANA Considerations

This document has no IANA considerations.

16. References

16.1. Normative References

[AES] National Institute of Standards and Technology, "Announcing the ADVANCED ENCRYPTION STANDARD (AES)", FIPS 197, DOI 10.6028/NIST.FIPS.197, November 2001, <https://nvlpubs.nist.gov/nistpubs/fips/NIST.FIPS.197.pdf>.

[cnsafaq] National Security Agency, "The Commercial National Security Algorithm Suite 2.0 and Quantum Computing FAQ", December 2024, <https://media.defense.gov/2022/Sep/07/2003071836/-1/-1/0/CSI_CNSA_2.0_FAQ_.PDF>.

[FIPS203] National Institute of Standards and Technology (2024), "Module-Lattice-Based Key-Encapsulation Mechanism Standard", (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publication (FIPS), NIST FIPS 203, <https://doi.org/10.6028/NIST.FIPS.203>.

[FIPS204] National Institute of Standards and Technology (2024), "Module-Lattice-Based Digital Signature Standard", (Department of Commerce, Washington, D.C.), Federal Information Processing Standards Publication (FIPS), NIST FIPS 204, <https://doi.org/10.6028/NIST.FIPS.204>.

[GCM] Dworkin, M., "Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC", NIST Special Publication 800-38D, DOI 10.6028/NIST.SP.800-38D, November 2007, <https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf>.

[I-D.ietf-tls-8773bis] Housley, R., "TLS 1.3 Extension for Using Certificates with an External Pre-Shared Key", July 2024, <https://datatracker.ietf.org/doc/draft-ietf-tls-8773bis/02/>.

[I-D.jenkins-cnsa2-pkix-profile] Jenkins, M. and A. Becker, "Commercial National Security Algorithm Suite Certificate and Certificate Revocation

List Profile", January 2025,
<https://datatracker.ietf.org/doc/draft-jenkins-cnsa2-pkix-profile/>.

[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>.

[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>.

[RFC9147] Rescorla, E., Tschofenig, H., and N. Modadugu, "The Datagram Transport Layer Security (DTLS) Protocol Version 1.3", RFC 9147, DOI 10.17487/RFC9147, April 2022, <https://www.rfc-editor.org/info/rfc9147>.

[RFC9151] Cooley, D., "Commercial National Security Algorithm (CNSA) Suite Profile for TLS and DTLS 1.2 and 1.3", RFC 9151, DOI 10.17487/RFC9151, April 2022, <https://www.rfc-editor.org/info/rfc9151>.

[RFC9881] Massimo, J., Kampanakis, P., Turner, S., and B. E. Westerbaan, "Internet X.509 Public Key Infrastructure -- Algorithm Identifiers for the Module-Lattice-Based Digital Signature Algorithm (ML-DSA)", RFC 9881, DOI 10.17487/RFC9881, October 2025, <https://www.rfc-editor.org/info/rfc9881>.

[SHS] National Institute of Standards and Technology (NIST), "Secure Hash Standard (SHS)", DOI 10.6028/NIST.FIPS.180-4, FIPS PUB 180-4, August 2015, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf>.

16.2. Informative References

[SP800-90c]
National Institute of Standards and Technology, "Recommendation for Random Bit Generator (RBG) Constructions.", NIST SP 800-90C 4pd, <https://doi.org/10.6028/NIST.SP.800-90C.4pd>.

[SP80059] National Institute of Standards and Technology, "Guideline for Identifying an Information System as a National Security System", Special Publication 59, DOI 10.6028/NIST.SP.800-59, August 2003, <https://csrc.nist.gov/publications/detail/sp/800-59/final>.

Authors' Addresses

Alison Becker
Email: aebecke@uwe.nsa.gov

Michael Jenkins
National Security Agency
Email: mjjenki@cyber.nsa.gov