Workgroup: SUIT
Internet-Draft: draft-ietf-suit-mti-00
Published: 13 March 2023
Intended Status: Standards Track
Expires: 14 September 2023
Authors: B. Moran Ø. Rønningstad A. Tsukamoto
Arm Limited Nordic Semiconductor

Mandatory-to-Implement Algorithms for Authors and Recipients of Software Update for the Internet of Things manifests

Abstract

This document specifies algorithm profiles for SUIT manifest parsers and authors to ensure better interoperability. These profiles apply specifically to a constrained node software update use case.

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 14 September 2023.

Copyright Notice

Copyright (c) 2023 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

Table of Contents

1. Introduction
2. Algorithms
  2.1. Digest Algorithms
  2.2. Authentication Algorithms
    2.2.1. Symmetric Authentication Algorithm
    2.2.2. Asymmetric Classical Authentication Algorithms
    2.2.3. Asymmetric Post-Quantum Authentication Algorithms
  2.3. Key Exchange Algorithms
    2.3.1. Symmetric
    2.3.2. Asymmetric Classical
  2.4. Encryption Algorithms
3. Profiles
  3.1. Symmetric MTI profile: suit-sha256-hmac-a128-ccm
  3.2. Current Asymmetric MTI Profile 1: suit-sha256-es256-ecdh-a128gcm
  3.3. Current Asymmetric MTI Profile 2: suit-sha256-eddsa-ecdh-a128gcm
  3.4. Future Asymmetric MTI Profile: suit-sha256-hsslms-hpke-a128gcm
  3.5. Other Profiles:
4. Security Considerations
5. IANA Considerations
6. A. Full CDDL
7. References
  7.1. Normative References
  7.2. Informative References
Authors' Addresses

1. Introduction

Mandatory algorithms may change over time due to an evolving threat landscape. Algorithms are grouped into algorithm profiles to account for this. Profiles may be deprecated over time. SUIT will define four choices of MTI profile specifically for constrained node software update. These profiles are:

*One Symmetric MTI profile

*Two "Current" Asymmetric MTI profiles

*One "Future" Asymmetric MTI profile

At least one MTI algorithm in each category MUST be FIPS qualified.

Because SUIT presents an asymmetric communication profile, with powerful/complex manifest authors and constrained manifest recipients, the requirements for Recipients and Authors are different.

Recipients MAY choose which MTI profile they wish to implement. It is RECOMMENDED that they implement the "Future" Asymmetric MTI profile. Recipients MAY implement any number of other profiles.

Authors MUST implement all MTI profiles. Authors MAY implement any number of other profiles.

Other use-cases of SUIT MAY define their own MTI algorithms.

2. Algorithms

The algorithms that form a part of the profiles defined in this document are grouped into:

*Digest Algorithms

*Authentication Algorithms

*Key Exchange Algorithms

*Encryption Algorithms

The COSE algorithm ID [IANA-COSE] for each algorithm is given in parentheses.

2.1. Digest Algorithms

*SHA-256 (-16)

2.2. Authentication Algorithms

Authentication Algorithms are divided into three groups: Symmetric, Asymmetric Classical, and Asymmetric Post-Quantum

2.2.1. Symmetric Authentication Algorithm

*HMAC-256 (5)

2.2.2. Asymmetric Classical Authentication Algorithms

*ES256 (-7)

*EdDSA (-8)

2.2.3. Asymmetric Post-Quantum Authentication Algorithms

*HSS-LMS (-46) [RFC8778]

2.3. Key Exchange Algorithms

Key Exchange Algorithms are divided into two groups: Symmetric, and Asymmetric Classical

2.3.1. Symmetric

*A128 (-3)

2.3.2. Asymmetric Classical

*COSE HPKE (TBD)

*ECDH-ES + HKDF-256 (-25)

2.4. Encryption Algorithms

*A128GCM (1)

3. Profiles

Recognized profiles are defined below.

3.1. Symmetric MTI profile: suit-sha256-hmac-a128-ccm

This profile requires the following algorithms:

*SHA-256

*HMAC-256

*A128W Key Wrap

*AES-CCM-16-128-128

3.2. Current Asymmetric MTI Profile 1: suit-sha256-es256-ecdh-a128gcm

This profile requires the following algorithms:

*SHA-256

*ES256

*ECDH

*AES-128-GCM

3.3. Current Asymmetric MTI Profile 2: suit-sha256-eddsa-ecdh-a128gcm

This profile requires the following algorithms:

*SHA-256

*EDDSA

*ECDH

*AES-128-GCM

3.4. Future Asymmetric MTI Profile: suit-sha256-hsslms-hpke-a128gcm

This profile requires the following algorithms:

*SHA-256

*HSS-LMS

*HPKE

*AES-128-GCM

3.5. Other Profiles:

Optional classical and PQC profiles are defined below.

*suit-sha256-eddsa-ecdh-es-chacha-poly

-SHA-256

-EdDSA

-ECDH-ES + HKDF-256

-ChaCha20 + Poly1305

*suit-sha256-falcon512-hpke-a128gcm

-SHA-256

-Falcon-512

-HPKE

-AES-128-GCM

*suit-shake256-dilithium-kyber-a128gcm

-SHAKE256

-Crystals-Dilithium

-Crystal-Kyber

-AES-128GCM

4. Security Considerations

For the avoidance of doubt, there are scenarios where payload or manifest encryption are not required. In these scenarios, the encryption element of the selected profile is simply not used.

5. IANA Considerations

TODO

-- back

6. A. Full CDDL

The following CDDL creates a subset of COSE for use with SUIT. Both tagged and untagged messages are defined. SUIT only uses tagged COSE messages, but untagged messages are also defined for use in protocols that share a ciphersuite with SUIT.

To be valid, the following CDDL MUST have the COSE CDDL appended to it. The COSE CDDL can be obtained by following the directions in [RFC9052], Section 1.4.

```
SUIT_COSE_tool_tweak /= suit-sha256-hmac-a128-ccm
SUIT_COSE_tool_tweak /= suit-sha256-es256-ecdh-a128gcm
SUIT_COSE_tool_tweak /= suit-sha256-eddsa-ecdh-a128gcm
SUIT_COSE_tool_tweak /= suit-sha256-hsslms-hpke-a128gcm

SUIT_COSE_tool_tweak /= SUIT_COSE_Profile_HMAC_A128CCM
SUIT_COSE_tool_tweak /= SUIT_COSE_Profile_ES256_ECDH_A128GCM
SUIT_COSE_tool_tweak /= SUIT_COSE_Profile_EDDSA_ECDH_A128GCM
SUIT_COSE_tool_tweak /= SUIT_COSE_Profile_HSSLMS_HPKE_A128GCM

suit-sha256-hmac-a128-ccm = [5, 12]
suit-sha256-es256-ecdh-a128gcm = [-7, 1]
suit-sha256-eddsa-ecdh-a128gcm = [-8, 1]
suit-sha256-hsslms-hpke-a128gcm = [-46, 1]

SUIT_COSE_Profile_HMAC_A128CCM = SUIT_COSE_Profile<5,12> .and COSE_Messa
SUIT_COSE_Profile_ES256_ECDH_A128GCM = SUIT_COSE_Profile<-7,1> .and COSE
SUIT_COSE_Profile_EDDSA_ECDH_A128GCM = SUIT_COSE_Profile<-8,1> .and COSE
SUIT_COSE_Profile_HSSLMS_HPKE_A128GCM = SUIT_COSE_Profile<-46,1> .and CO

SUIT_COSE_Profile<authid, encid> = SUIT_COSE_Messages<authid,encid>

SUIT_COSE_Messages<authid, encid> = SUIT_COSE_Untagged_Message<authid, e
    SUIT_COSE_Tagged_Message<authid, encid>

SUIT_COSE_Untagged_Message<authid, encid> = SUIT_COSE_Sign<authid> /
    SUIT_COSE_Sign1<authid> / SUIT_COSE_Encrypt<encid> /
    SUIT_COSE_Encrypt0<encid> / SUIT_COSE_Mac<authid> /
    SUIT_COSE_Mac0<authid>

SUIT_COSE_Tagged_Message<authid, encid> = SUIT_COSE_Sign_Tagged<authid>
    SUIT_COSE_Sign1_Tagged<authid> / SUIT_COSE_Encrypt_Tagged<encid> /
    SUIT_COSE_Encrypt0_Tagged<encid> / SUIT_COSE_Mac_Tagged<authid> /
    SUIT_COSE_Mac0_Tagged<authid>

; Note: This is not the same definition as is used in COSE.
; It restricts a COSE header definition further without
; repeating the COSE definition. It should be merged
; with COSE by using the CDDL .and operator.
SUIT_COSE_Profile_Headers<algid> = (
    protected : bstr .cbor SUIT_COSE_alg_map<algid>,
    unprotected : SUIT_COSE_header_map
)
SUIT_COSE_alg_map<algid> = {
    1 => algid,
    * int => any
}

SUIT_COSE_header_map = {
    * int => any
```

```
}

SUIT_COSE_Sign_Tagged<authid> = #6.98(SUIT_COSE_Sign<authid>)


SUIT_COSE_Sign<authid> = [
    SUIT_COSE_Profile_Headers<authid>,
    payload : bstr / nil,
    signatures : [+ SUIT_COSE_Signature<authid>]
]


SUIT_COSE_Signature<authid> =  [
    SUIT_COSE_Profile_Headers<authid>,
    signature : bstr
]


SUIT_COSE_Sign1_Tagged<authid> = #6.18(SUIT_COSE_Sign1<authid>)


SUIT_COSE_Sign1<authid> = [
    SUIT_COSE_Profile_Headers<authid>,
    payload : bstr / nil,
    signature : bstr
]


SUIT_COSE_Encrypt_Tagged<encid> = #6.96(SUIT_COSE_Encrypt<encid>)


SUIT_COSE_Encrypt<encid> = [
    SUIT_COSE_Profile_Headers<encid>,
    ciphertext : bstr / nil,
    recipients : [+SUIT_COSE_recipient<encid>]
]


SUIT_COSE_recipient<encid> = [
    SUIT_COSE_Profile_Headers<encid>,
    ciphertext : bstr / nil,
    ? recipients : [+SUIT_COSE_recipient<encid>]
]


SUIT_COSE_Encrypt0_Tagged<encid> = #6.16(SUIT_COSE_Encrypt0<encid>)


SUIT_COSE_Encrypt0<encid> = [
    SUIT_COSE_Profile_Headers<encid>,
```

```
    ciphertext : bstr / nil,
]


SUIT_COSE_Mac_Tagged<authid> = #6.97(SUIT_COSE_Mac<authid>)


SUIT_COSE_Mac<authid> = [
   SUIT_COSE_Profile_Headers<authid>,
   payload : bstr / nil,
   tag : bstr,
   recipients :[+SUIT_COSE_recipient<authid>]
]


SUIT_COSE_Mac0_Tagged<authid> = #6.17(SUIT_COSE_Mac0<authid>)


SUIT_COSE_Mac0<authid> = [
   SUIT_COSE_Profile_Headers<authid>,
   payload : bstr / nil,
   tag : bstr,
]
```

7. References

7.1. Normative References

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

[RFC8778] Housley, R., "Use of the HSS/LMS Hash-Based Signature Algorithm with CBOR Object Signing and Encryption (COSE)", RFC 8778, DOI 10.17487/RFC8778, April 2020, <https://www.rfc-editor.org/info/rfc8778>.

[RFC9052] Schaad, J., "CBOR Object Signing and Encryption (COSE): Structures and Process", STD 96, RFC 9052, DOI 10.17487/RFC9052, August 2022, <https://www.rfc-editor.org/info/rfc9052>.

7.2. Informative References

[I-D.ietf-suit-manifest] Moran, B., Tschofenig, H., Birkholz, H., Zandberg, K., and O. Rønningstad, "A Concise Binary Object Representation (CBOR)-based Serialization Format for the Software Updates for Internet of Things (SUIT) Manifest", Work in Progress, Internet-Draft, draft-ietf-suit-manifest-22, 27 February 2023, <https://datatracker.ietf.org/doc/html/draft-ietf-suit-manifest-22>.

[IANA-COSE] "CBOR Object Signing and Encryption (COSE)", 2022, <https://www.iana.org/assignments/cose/cose.xhtml>.

Authors' Addresses

Brendan Moran
Arm Limited

Email: brendan.moran.ietf@gmail.com

Øyvind Rønningstad
Nordic Semiconductor

Email: oyvind.ronningstad@gmail.com

Akira Tsukamoto

Email: akira.tsukamoto@gmail.com