COSE
Internet-Draft
Intended status: Standards Track
Expires: 8 January 2026

E. Lundberg, Ed.
Yubico
M. B. Jones
Self-Issued Consulting

7 July 2025

Split signing algorithms for COSE
draft-lundberg-cose-two-party-signing-algs-02

Abstract

This specification defines COSE algorithm identifiers used when one signing operation is split between two cooperating parties. When performing split signing, the first party typically hashes the data to be signed and the second party signs the hashed data computed by the first party. This can be useful when communication with the party holding the signing private key occurs over a limited-bandwidth channel, such as NFC or Bluetooth Low Energy (BLE), in which it is infeasible to send the complete set of data to be signed. The resulting signatures are identical in structure to those computed by a single party, and can be verified using the same verification algorithm without additional steps to preprocess the signed data.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-lundberg-cose-two-party-signing-algs/.

Discussion of this document takes place on the COSE Working Group mailing list (mailto:cose@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/cose/. Subscribe at https://www.ietf.org/mailman/listinfo/cose/.

Source for this draft and an issue tracker can be found at https://github.com/YubicoLabs/cose-two-party-signing-algs-rfc.

Status of This Memo

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

This Internet-Draft will expire on 8 January 2026.

Copyright Notice

Table of Contents

1. Introduction 3
  1.1. Requirements Notation and Conventions 4
2. Split Signing Algorithms 4
  2.1. ECDSA 4
  2.2. HashEdDSA 5
3. COSE Signing Arguments 6
  3.1. COSE Signing Arguments Common Parameters 7
4. IANA Considerations 8
  4.1. COSE Algorithms Registrations 8
  4.2. COSE Signing Arguments Common Parameters Registry 9
  4.3. COSE Signing Arguments Algorithm Parameters Registry . . 10
5. References 10
  5.1. Normative References 10
  5.2. Informative References 11
Document History 11
Authors' Addresses 12

1. Introduction

CBOR Object Signing and Encryption (COSE) [RFC9052] algorithm identifiers are used to specify the cryptographic operations used to create cryptographic data structures, but do not record internal details of how the cryptography was performed, since those details are typically irrelevant for the recipient. The algorithm identifiers defined by this specification facilitate splitting a signing operation between two cooperating parties, by specifying the division of responsibilities between the two parties. The resulting signature can be verified by the same verification algorithm as if it had been created by a single party, so this division of responsibilities is an implementation detail of the signer. Verifiers therefore do not use these split algorithm identifiers, and instead use the corresponding non-split algorithm identifier which identifies the same verification algorithm as the split algorithm identifier would.

A primary use case for this is splitting a signature operation between a software application and a discrete hardware security module (HSM) holding the private key. In particular, since the data link between them may have limited bandwidth, it may not be practical to send the entire original message to the HSM. Instead, since most signature algorithms begin with digesting the message into a fixed-length intermediate input, this initial digest can be computed by the software application while the HSM computes the rest of the signature algorithm on the digest.

Since different signature algorithms digest the message in different ways and at different stages of the algorithm, there is no one generally-applicable way to define such a division point for every possible signature algorithm. Therefore, this specification defines algorithm identifiers encoding, for a specific set of signature algorithms, which steps of the signature algorithm are performed by the _digester_ (e.g., software application) and which are performed by the _signer_ (e.g., HSM). In general, the _signer_ holds exclusive control of the signing private key.

Note that these algorithm identifiers do not define new "pre-hashed" variants of the base signature algorithm, nor an intermediate "hash envelope" data structure, such as that defined in [I-D.COSE-Hash-Envelope]. Rather, these identifiers correspond to existing signature algorithms that would typically be executed by a single party, but split into two stages. The resulting signatures are identical to those computed by a single party, and can be verified using the same verification algorithms without additional special steps to process the signed data.

However some signature algorithms, such as PureEdDSA [RFC8032], cannot be split in this way and therefore cannot be assigned split signing algorithm identifiers. However, if such a signature algorithm defines a "pre-hashed" variant, such as Ed25519ph [RFC8032], that "pre-hashed" algorithm can also be assigned a split signing algorithm identifier, enabling the hashing step to be performed by the _digester_ and the signing step to be executed by the _signer_.

1.1. Requirements Notation and Conventions

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.

2. Split Signing Algorithms

This section defines divisions of signing algorithm steps between a _digester_ and a _signer_ in a split signing protocol, and assigns algorithm identifiers to these algorithm divisions. The _digester_ performs the first part of the split algorithm and does not have access to the signing private key, while the _signer_ performs the second part of the split algorithm and has access to the signing private key. For signing algorithms that format the message to insert domain separation tags, as described in Section 2.2.5 of [RFC9380], this message formatting is also performed by the _signer_.

The algorithm identifiers defined in this specification MAY appear in COSE structures used internally between the _digester_ and the _signer_ in a split signing protocol, but SHOULD NOT appear in COSE structures consumed by signature verifiers. COSE structures consumed by signature verifiers SHOULD instead use the corresponding conventional algorithm identifiers for the verification algorithm. These are listed in the "Verification algorithm" column in the tables defining split signing algorithm identifiers.

2.1. ECDSA

ECDSA [FIPS-186-5] split signing uses the following division between the _digester_ and the _signer_ of the steps of the ECDSA signature generation algorithm [FIPS-186-5]:

* The signing procedure is defined in Section 6.4.1 of [FIPS-186-5].

* The _digester_ performs Step 1 of the signing procedure - hashing the message, producing the value _H_.

* The message input to the _signer_ is the value _H_ defined in the signing procedure.

* The _signer_ resumes the signing procedure from Step 2.

The following algorithm identifiers are defined:

Name	COSE Value	Verification algorithm	Description
ESP256-split	TBD	ESP256	ESP256 [I-D.jose-fully-spec-algs] split signing as defined here
ESP384-split	TBD	ESP384	ESP384 [I-D.jose-fully-spec-algs] split signing as defined here
ESP512-split	TBD	ESP512	ESP512 [I-D.jose-fully-spec-algs] split signing as defined here

Table 1

Note: This is distinct from the similarly named Split-ECDSA (SECDSA) [SECDSA], although SECDSA can be implemented using this split procedure as a component.

2.2. HashEdDSA

Split HashEdDSA [RFC8032] uses the following division between the _digester_ and the _signer_ of the steps of the HashEdDSA signing algorithm [RFC8032]:

* HashEdDSA is a combination of the EdDSA signing procedure and the PureEdDSA signing procedure. The EdDSA signing procedure is defined in the first paragraph of Section 3.3 of [RFC8032]. The PureEdDSA signing procedure is defined in the second paragraph of Section 3.3 of [RFC8032].

* The _digester_ computes the value PH(M) defined in the EdDSA signing procedure.

* The message input to the _signer_ is the value PH(M) defined in the EdDSA signing procedure. This value is represented as M in the PureEdDSA signing procedure.

* The _signer_ executes the PureEdDSA signing procedure, where the value denoted M in the PureEdDSA signing procedure takes the value denoted PH(M) in the EdDSA signing procedure.

PureEdDSA [RFC8032] cannot be divided in this way since such a division would require that the _digester_ has access to the private key.

The following algorithm identifiers are defined:

Name	COSE Value	Verification algorithm	Description
Ed25519ph-split	TBD	Ed25519ph	Ed25519ph [I-D.jose-fully-spec-algs] split signing as defined here (NOTE: Ed25519ph not yet registered)
Ed448ph-split	TBD	Ed448ph	Ed448ph [I-D.jose-fully-spec-algs] split signing as defined here (NOTE: Ed448ph not yet registered)

Table 2

3. COSE Signing Arguments

While many signature algorithms take the private key and data to be signed as the only two parameters, some signature algorithms have additional parameters that must also be set. For example, to sign using a key derived by ARKG [I-D.bradleylundberg-ARKG], two additional arguments kh and info are needed in ARKG-Derive-Private-Key to derive the signing private key.

While such additional arguments are simple to provide to the API of the signing procedure in a single-party context, in a split signing context these additional arguments also need to be conveyed from the _digester_ to the _signer_. For this purpose, we define a new COSE structure COSE_Sign_Args for "COSE signing arguments". This enables defining a unified, algorithm-agnostic protocol between the

digester and the _signer_, rather than requiring a distinct protocol for each signature algorithm for the sake of conveying algorithm-specific parameters.

COSE_Sign_Args is built on a CBOR map. The set of common parameters that can appear in a COSE_Sign_Args can be found in the IANA "COSE Signing Arguments Common Parameters" registry (TODO). Additional parameters defined for specific signing algorithms can be found in the IANA "COSE Signing Arguments Algorithm Parameters" registry (TODO).

The CDDL grammar describing COSE_Sign_Args, using the CDDL fragment defined in Section 1.5 of [RFC9052], is:

```
COSE_Sign_Args = {
    3 ^ => tstr / int,  ; alg
    * label => values,
}
```

3.1. COSE Signing Arguments Common Parameters

This document defines a set of common parameters for a COSE Signing Arguments object. Table 3 provides a summary of the parameters defined in this section.

Name	Label	CBOR Type	Value Registry	Description
alg	3	tstr / int	COSE Algorithms	Signing algorithm to use

Table 3: Common parameters of the COSE_Sign_Args structure.

* alg: This parameter identifies the signing algorithm the additional arguments apply to. The signer MUST verify that this algorithm matches any key usage restrictions set on the key to be used. If the algorithms do not match, then the signature operation MUST be aborted with an error.

Definitions of COSE algorithms MAY define additional algorithm-specific parameters for COSE_Sign_Args.

The following CDDL example conveys additional arguments for signing data using the ESP256-split algorithm (see Section 2.1) and a key derived using ARKG-P256 [I-D.bradleylundberg-ARKG]:

```
{
  3: -65539,   ; alg: ESP256-split with ARKG-P256 (placeholder value)

               ; ARKG-P256 key handle
               ; (HMAC-SHA-256-128 followed by
                  SEC1 uncompressed ECDH public key)
  -1: h'27987995f184a44cfa548d104b0a461d
        0487fc739dbcdabc293ac5469221da91b220e04c681074ec4692a76ffacb9043de
          c2847ea9060fd42da267f66852e63589f0c00dc88f290d660c65a65a50c86361',

               ; info argument to ARKG-Derive-Private-Key
  -2: 'ARKG-P256.test vectors',
}
```

4. IANA Considerations

4.1. COSE Algorithms Registrations

This section registers the following values in the IANA "COSE Algorithms" registry [IANA.COSE]:

* Name: ESP256-split
  - Value: TBD (Requested Assignment -300)
  - Description: ESP256 [I-D.jose-fully-spec-algs] split signing
  - Capabilities: [kty]
  - Change Controller: IETF
  - Reference: Section 2.1 of this specification
  - Recommended: Yes
* Name: ESP384-split
  - Value: TBD (Requested Assignment -301)
  - Description: ESP384 [I-D.jose-fully-spec-algs] split signing
  - Capabilities: [kty]
  - Change Controller: IETF
  - Reference: Section 2.1 of this specification
  - Recommended: Yes

* Name: ESP512-split
  - Value: TBD (Requested Assignment -302)
  - Description: ESP512 [I-D.jose-fully-spec-algs] split signing
  - Capabilities: [kty]
  - Change Controller: IETF
  - Reference: Section 2.1 of this specification
  - Recommended: Yes
* Name: Ed25519ph-split
  - Value: TBD (Requested Assignment -303)
  - Description: Ed25519ph [I-D.jose-fully-spec-algs] split signing
  - Capabilities: [kty]
  - Change Controller: IETF
  - Reference: Section 2.2 of this specification
  - Recommended: Yes
* Name: Ed448ph-split
  - Value: TBD (Requested Assignment -304)
  - Description: Ed448ph [I-D.jose-fully-spec-algs] split signing
  - Capabilities: [kty]
  - Change Controller: IETF
  - Reference: Section 2.2 of this specification
  - Recommended: Yes

4.2. COSE Signing Arguments Common Parameters Registry

TODO

4.3. COSE Signing Arguments Algorithm Parameters Registry

TODO

5. References

5.1. Normative References

[I-D.bradleylundberg-ARKG]
Lundberg, E. and J. Bradley, "The Asynchronous Remote Key Generation (ARKG) algorithm", Work in Progress, Internet-Draft, draft-bradleylundberg-cfrg-arkg-08, 29 April 2025, <https://datatracker.ietf.org/doc/html/draft-bradleylundberg-cfrg-arkg-08>.

[I-D.jose-fully-spec-algs]
Jones, M. B. and O. Steele, "Fully-Specified Algorithms for JOSE and COSE", Work in Progress, Internet-Draft, draft-ietf-jose-fully-specified-algorithms-13, 11 May 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-jose-fully-specified-algorithms-13>.

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

[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/rfc/rfc2119>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <https://www.rfc-editor.org/rfc/rfc8032>.

[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/rfc/rfc8174>.

[RFC8610] Birkholz, H., Vigano, C., and C. Bormann, "Concise Data Definition Language (CDDL): A Notational Convention to Express Concise Binary Object Representation (CBOR) and JSON Data Structures", RFC 8610, DOI 10.17487/RFC8610, June 2019, <https://www.rfc-editor.org/rfc/rfc8610>.

[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/rfc/rfc9052>.

[SEC1] Certicom Research, "SEC 1: Elliptic Curve Cryptography", May 2009, <https://www.secg.org/sec1-v2.pdf>.

5.2. Informative References

[FIPS-186-5] National Institute of Standards and Technology, "Digital Signature Standard (DSS)", February 2023, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf>.

[FIPS-204] National Institute of Standards and Technology, "Module-Lattice-Based Digital Signature Standard", August 2024, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.204.pdf>.

[I-D.COSE-Hash-Envelope] Steele, O., Lasker, S., and H. Birkholz, "COSE Hash Envelope", Work in Progress, Internet-Draft, draft-ietf-cose-hash-envelope-05, 28 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-cose-hash-envelope-05>.

[RFC9380] Faz-Hernandez, A., Scott, S., Sullivan, N., Wahby, R. S., and C. A. Wood, "Hashing to Elliptic Curves", RFC 9380, DOI 10.17487/RFC9380, August 2023, <https://www.rfc-editor.org/rfc/rfc9380>.

[SECDSA] Verheul, E., "SECDSA: Mobile signing and authentication under classical "sole control"", July 2021, <https://eprint.iacr.org/2021/910>.

Document History

-02

* Renamed document from "COSE Algorithms for Two-Party Signing" to "Split signing algorithms for COSE" and updated introduction and terminology accordingly.

* Dropped definitions for HashML-DSA, as split variants of ML-DSA are being actively discussed in other IETF groups.

* Changed "Base algorithm" heading in definition tables to "Verification algorithm".

* Remodeled COSE_Key_Ref as COSE_Sign_Args.

  - Dropped definitions of reference types for COSE Key Types registry.

-01

* Added IANA registration requests for algorithms defined.

* Updated references and other editorial tweaks.

-00

* Initial individual draft

Authors' Addresses

Emil Lundberg (editor)
Yubico
G辰vlegatan 22
Stockholm
Sweden
Email: emil@emlun.se

Michael B. Jones
Self-Issued Consulting
United States
Email: michael_b_jones@hotmail.com
URI: https://self-issued.info/