Abstract

The W3C Web Authentication (WebAuthn) specification and the FIDO Alliance FIDO2 Client to Authenticator Protocol (CTAP) specification use CBOR Object Signing and Encryption (COSE) algorithm identifiers. This specification registers the following algorithms (which are used by WebAuthn and CTAP implementations) in the IANA "COSE Algorithms" registry: RSASSA-PKCS1-v1_5 using SHA-256, SHA-384, SHA-512, and SHA-1; and Elliptic Curve Digital Signature Algorithm (ECDSA) using the secp256k1 curve and SHA-256. It registers the secp256k1 elliptic curve in the IANA "COSE Elliptic Curves" registry. Also, for use with JSON Object Signing and Encryption (JOSE), it registers the algorithm ECDSA using the secp256k1 curve and SHA-256 in the IANA "JSON Web Signature and Encryption Algorithms" registry and the secp256k1 elliptic curve in the IANA "JSON Web Key Elliptic Curve" registry.
Copyright Notice

Copyright (c) 2020 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 Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction
   1.1. Requirements Notation and Conventions
2. RSASSA-PKCS1-v1_5 Signature Algorithm
3. Using secp256k1 with JOSE and COSE
   3.1. JOSE and COSE secp256k1 Curve Key Representations
   3.2. ECDSA Signature with secp256k1 Curve
   3.3. Other Uses of the secp256k1 Elliptic Curve
4. IANA Considerations
   4.1. COSE Algorithms Registrations
   4.2. COSE Elliptic Curves Registrations
   4.3. JOSE Algorithms Registrations
   4.4. JSON Web Key Elliptic Curves Registrations
5. Security Considerations
   5.1. RSA Key Size Security Considerations
   5.2. RSASSA-PKCS1-v1_5 with SHA-2 Security Considerations
   5.3. RSASSA-PKCS1-v1_5 with SHA-1 Security Considerations
   5.4. secp256k1 Security Considerations
6. References
   6.1. Normative References
   6.2. Informative References
1. Introduction

This specification defines how to use several algorithms with CBOR Object Signing and Encryption (COSE) [RFC8152] that are used by implementations of the W3C Web Authentication (WebAuthn) [WebAuthn] and FIDO Alliance FIDO2 Client to Authenticator Protocol (CTAP) [CTAP] specifications. This specification registers these algorithms in the IANA "COSE Algorithms" registry [IANA.COSE.Algorithms] and registers an elliptic curve in the IANA "COSE Elliptic Curves" registry [IANA.COSE.Curves]. This specification also registers a corresponding algorithm for use with JSON Object Signing and Encryption (JOSE) [RFC7515] in the IANA "JSON Web Signature and Encryption Algorithms" registry [IANA.JOSE.Algorithms] and registers an elliptic curve in the IANA "JSON Web Key Elliptic Curve" registry [IANA.JOSE.Curves].

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. RSASSA-PKCS1-v1_5 Signature Algorithm

The RSASSA-PKCS1-v1_5 signature algorithm is defined in [RFC8017]. The RSASSA-PKCS1-v1_5 signature algorithm is parameterized with a hash function (h).

A key of size 2048 bits or larger MUST be used with these algorithms. Implementations need to check that the key type is 'RSA' when creating or verifying a signature.

The RSASSA-PKCS1-v1_5 algorithms specified in this document are in the following table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Hash</th>
<th>Description</th>
<th>Recommended</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS256</td>
<td>-257</td>
<td>SHA-256</td>
<td>RSASSA-PKCS1-v1_5 using SHA-256</td>
<td>No</td>
</tr>
<tr>
<td>RS384</td>
<td>-258</td>
<td>SHA-384</td>
<td>RSASSA-PKCS1-v1_5 using SHA-384</td>
<td>No</td>
</tr>
<tr>
<td>RS512</td>
<td>-259</td>
<td>SHA-512</td>
<td>RSASSA-PKCS1-v1_5 using SHA-512</td>
<td>No</td>
</tr>
<tr>
<td>RS1</td>
<td>-65535</td>
<td>SHA-1</td>
<td>RSASSA-PKCS1-v1_5 using SHA-1</td>
<td>Deprecated</td>
</tr>
</tbody>
</table>

*Table 1: RSASSA-PKCS1-v1_5 Algorithm Values*
Security considerations for use of the first three algorithms are in Section 5.2. Security considerations for use of the last algorithm are in Section 5.3.

Note that these algorithms are already present in the IANA "JSON Web Signature and Encryption Algorithms" registry [IANA.JOSE.Algorithms], and so these registrations are only for the IANA "COSE Algorithms" registry [IANA.COSE.Algorithms].

3. Using secp256k1 with JOSE and COSE

This section defines algorithm encodings and representations enabling the Standards for Efficient Cryptography Group (SECG) elliptic curve secp256k1 [SEC2] to be used for JOSE [RFC7515] and COSE [RFC8152] messages.

3.1. JOSE and COSE secp256k1 Curve Key Representations

The SECG elliptic curve secp256k1 [SEC2] is represented in a JSON Web Key (JWK) [RFC7517] using these values:

- `kty`: EC
- `crv`: secp256k1

plus the values needed to represent the curve point, as defined in Section 6.2.1 of [RFC7518]. As a compressed point encoding representation is not defined for JWK elliptic curve points, the uncompressed point encoding defined there MUST be used. The x and y values represented MUST both be exactly 256 bits, with any leading zeros preserved. Other optional values such as `alg` MAY also be present.

It is represented in a COSE_Key [RFC8152] using these values:

- `kty (1): EC2 (2)`
- `crv (-1): secp256k1 (8)`

plus the values needed to represent the curve point, as defined in Section 13.1.1 of [RFC8152]. Either the uncompressed or compressed point encoding representations defined there can be used. The x value represented MUST be exactly 256 bits, with any leading zeros preserved. If the uncompressed representation is used, the y value represented MUST likewise be exactly 256 bits, with any leading zeros preserved; if the compressed representation is used, the y value is a boolean value, as specified in Section 13.1.1 of [RFC8152]. Other optional values such as `alg (3)` MAY also be present.

3.2. ECDSA Signature with secp256k1 Curve

The ECDSA signature algorithm is defined in [DSS]. This specification defines the ES256K algorithm identifier, which is used to specify the use of ECDSA with the secp256k1 curve and the SHA-256 [DSS] cryptographic hash function. Implementations need to check that the key type is EC for JOSE or EC2 (2) for COSE and that the curve of the key is secp256k1 when creating or verifying a signature.
The ECDSA secp256k1 SHA-256 digital signature is generated as follows:

1. Generate a digital signature of the JWS Signing Input or the COSE Sig_structure using ECDSA secp256k1 SHA-256 with the desired private key. The output will be the pair \((R, S)\), where \(R\) and \(S\) are 256-bit unsigned integers.
2. Turn \(R\) and \(S\) into octet sequences in big-endian order, with each array being 32 octets long. The octet sequence representations **MUST NOT** be shortened to omit any leading zero octets contained in the values.
3. Concatenate the two octet sequences in the order \(R\) and then \(S\). (Note that many ECDSA implementations will directly produce this concatenation as their output.)
4. The resulting 64-octet sequence is the JWS Signature or COSE signature value.

Implementations **SHOULD** use a deterministic algorithm to generate the ECDSA nonce, \(k\), such as the algorithm defined in [RFC6979]. However, in situations where devices are vulnerable to physical attacks, deterministic ECDSA has been shown to be susceptible to fault injection attacks [KUDELSKI17] [EURO-SP18]. Where this is a possibility, implementations **SHOULD** implement appropriate countermeasures. Where there are specific certification requirements (such as FIPS approval), implementors should check whether deterministic ECDSA is an approved nonce generation method.

The ECDSA secp256k1 SHA-256 algorithm specified in this document uses these identifiers:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
<th>Description</th>
<th>Recommended</th>
</tr>
</thead>
<tbody>
<tr>
<td>ES256K</td>
<td>-47</td>
<td>ECDSA using secp256k1 curve and SHA-256</td>
<td>No</td>
</tr>
</tbody>
</table>

*Table 2: ECDSA Algorithm Values*

When using a JWK or COSE_Key for this algorithm, the following checks are made:

- The \(kty\) field **MUST** be present, and it **MUST** be \(EC\) for JOSE or \(EC2\) for COSE.
- The \(crv\) field **MUST** be present, and it **MUST** represent the secp256k1 elliptic curve.
- If the \(alg\) field is present, it **MUST** represent the ES256K algorithm.
- If the \(key_ops\) field is present, it **MUST** include \(sign\) when creating an ECDSA signature.
- If the \(key_ops\) field is present, it **MUST** include \(verify\) when verifying an ECDSA signature.
- If the JWK use field is present, its value **MUST** be \(sig\).  

### 3.3. Other Uses of the secp256k1 Elliptic Curve

This specification defines how to use the secp256k1 curve for ECDSA signatures for both JOSE and COSE implementations. While in theory the curve could also be used for ECDH-ES key agreement, it is beyond the scope of this specification to state whether this is or is not advisable. Thus, whether or not to recommend its use with ECDH-ES is left for experts to decide in future specifications.
When used for ECDSA, the secp256k1 curve **MUST** be used only with the ES256K algorithm identifier and not any others, including not with the COSE ES256 identifier. Note that the ES256K algorithm identifier needed to be introduced for JOSE to sign with the secp256k1 curve because the JOSE ES256 algorithm is defined to be used only with the P-256 curve. The COSE treatment of how to sign with secp256k1 is intentionally parallel to that for JOSE, where the secp256k1 curve **MUST** be used with the ES256K algorithm identifier.

### 4. IANA Considerations

#### 4.1. COSE Algorithms Registrations

IANA has registered the following values in the "COSE Algorithms" registry [IANA.COSE.Algorithms].

Name: RS256  
Value: -257  
Description: RSASSA-PKCS1-v1_5 using SHA-256  
Change Controller: IESG  
Reference: Section 2 of RFC 8812  
Recommended: No

Name: RS384  
Value: -258  
Description: RSASSA-PKCS1-v1_5 using SHA-384  
Change Controller: IESG  
Reference: Section 2 of RFC 8812  
Recommended: No

Name: RS512  
Value: -259  
Description: RSASSA-PKCS1-v1_5 using SHA-512  
Change Controller: IESG  
Reference: Section 2 of RFC 8812  
Recommended: No

Name: RS1  
Value: -65535  
Description: RSASSA-PKCS1-v1_5 using SHA-1  
Change Controller: IESG  
Reference: Section 2 of RFC 8812  
Recommended: Deprecated

Name: ES256K
Value: -47
Description: ECDSA using secp256k1 curve and SHA-256
Change Controller: IESG
Reference: Section 3.2 of RFC 8812
Recommended: No

4.2. COSE Elliptic Curves Registrations
IANA has registered the following value in the "COSE Elliptic Curves" registry [IANA.COSE.Curves].

Name: secp256k1
Value: 8
Key Type: EC2
Description: SECG secp256k1 curve
Change Controller: IESG
Reference: Section 3.1 of RFC 8812
Recommended: No

4.3. JOSE Algorithms Registrations
IANA has registered the following value in the "JSON Web Signature and Encryption Algorithms" registry [IANA.JOSE.Algorithms].

Algorithm Name: ES256K
Algorithm Description: ECDSA using secp256k1 curve and SHA-256
Algorithm Usage Location(s): alg
JOSE Implementation Requirements: Optional
Change Controller: IESG
Reference: Section 3.2 of RFC 8812
Algorithm Analysis Document(s): [SEC2]

4.4. JSON Web Key Elliptic Curves Registrations
IANA has registered the following value in the "JSON Web Key Elliptic Curve" registry [IANA.JOSE.Curves].

Curve Name: secp256k1
Curve Description: SECG secp256k1 curve
JOSE Implementation Requirements: Optional
Change Controller: IESG
Specification Document(s): Section 3.1 of RFC 8812
5. Security Considerations

5.1. RSA Key Size Security Considerations

The security considerations on key sizes for RSA algorithms from Section 6.1 of [RFC8230] also apply to the RSA algorithms in this specification.

5.2. RSASSA-PKCS1-v1_5 with SHA-2 Security Considerations

The security considerations on the use of RSASSA-PKCS1-v1_5 with SHA-2 hash functions (SHA-256, SHA-384, and SHA-512) from Section 8.3 of [RFC7518] also apply to their use in this specification. For that reason, these algorithms are registered as being "Not Recommended". Likewise, the exponent restrictions described in Section 8.3 of [RFC7518] also apply.

5.3. RSASSA-PKCS1-v1_5 with SHA-1 Security Considerations

The security considerations on the use of the SHA-1 hash function from [RFC6194] apply in this specification. For that reason, the "RS1" algorithm is registered as "Deprecated". Likewise, the exponent restrictions described in Section 8.3 of [RFC7518] also apply.

A COSE algorithm identifier for this algorithm is nonetheless being registered because deployed Trusted Platform Modules (TPMs) continue to use it; therefore, WebAuthn implementations need a COSE algorithm identifier for "RS1" when TPM attestations using this algorithm are being represented. New COSE applications and protocols MUST NOT use this algorithm.

5.4. secp256k1 Security Considerations

Care should be taken that a secp256k1 key is not mistaken for a P-256 [RFC7518] key, given that their representations are the same except for the \texttt{crv} value. As described in Section 8.1.1 of [RFC8152], we currently do not have any way to deal with this attack except to restrict the set of curves that can be used.

The procedures and security considerations described in the [SEC1], [SEC2], and [DSS] specifications apply to implementations of this specification.

Timing side-channel attacks are possible if the implementation of scalar multiplication over the curve does not execute in constant time.

There are theoretical weaknesses with this curve that could result in future attacks. While these potential weaknesses are not unique to this curve, they are the reason that this curve is registered as "Recommended: No".

6. References

6.1. Normative References
6.2. Informative References


Michael B. Jones
Microsoft
Email: mbj@microsoft.com
URI: https://self-issued.info/