JOSE Working Group	M. Jones
Internet-Draft	Microsoft
Intended status: Standards Track	December 27, 2012
Expires: June 30, 2013	

TOC

JSON Private and Symmetric Key draft-jones-jose-json-private-and-symmetric-key-00

Abstract

The JSON Private Key specification extends the JSON Web Key (JWK) and JSON Web Algorithms (JWA) specifications to define JavaScript Object Notation (JSON) representations of private keys and symmetric keys.

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 http://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 June 30, 2013.

Table of Contents

---

1. Introduction

TOC

The JSON Private Key specification extends the JSON Web Key (JWK) **[JWK]** and JSON Web Algorithms (JWA) **[JWA]** specifications to define JavaScript Object Notation (JSON) **[RFC4627]** representations of private and symmetric keys.

1.1. Notational Conventions

TOC

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in Key words for use in RFCs to Indicate Requirement Levels **[RFC2119]**.

---

2. Terminology

TOC

This specification uses the same terminology as the JSON Web Key (JWK) **[JWK]** and JSON Web Algorithms (JWA) **[JWA]** specifications.

---

3. JWK Parameters for Private Keys

TOC

This section defines additional JSON Web Key parameters that enable JWKs to represent private keys.

3.1. JWK Parameters for Elliptic Curve Private Keys

TOC

When the JWK `kty` member value is `EC`, the following member MAY be used to represent an Elliptic Curve private key:

3.1.1. "d" (ECC Private Key) Parameter

TOC

The `d` (ECC private key) member contains the Elliptic Curve private key value. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array. The array representation MUST not be shortened to omit any leading zero bytes. For instance, when representing 521 bit integers, the byte array to be base64url

encoded MUST contain 66 bytes, including any leading zero bytes.

3.2. JWK Parameters for RSA Private Keys

TOC

When the JWK `kty` member value is `RSA`, the following member MAY be used to represent an RSA private key:

3.2.1. "d" (Private Exponent) Parameter

TOC

The `d` (private exponent) member contains the private exponent value for the RSA private key. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array. The array representation MUST not be shortened to omit any leading zero bytes. For instance, when representing 2048 bit integers, the byte array to be base64url encoded MUST contain 256 bytes, including any leading zero bytes.

3.2.2. "p" (First Prime Factor) Parameter

TOC

The `p` (first prime factor) member contains the first prime factor, a positive integer. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array.

3.2.3. "q" (Second Prime Factor) Parameter

TOC

The `q` (second prime factor) member contains the second prime factor, a positive integer. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array.

3.2.4. "dp" (First Factor CRT Exponent) Parameter

TOC

The `dp` (first factor CRT exponent) member contains the Chinese Remainder Theorem (CRT) exponent of the first factor, a positive integer. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array.

3.2.5. "dq" (Second Factor CRT Exponent) Parameter

TOC

The `dq` (second factor CRT exponent) member contains the Chinese Remainder Theorem (CRT) exponent of the second factor, a positive integer. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array.

3.2.6. "qi" (First CRT Coefficient) Parameter

TOC

The `dp` (first CRT coefficient) member contains the Chinese Remainder Theorem (CRT) coefficient of the second factor, a positive integer. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array.

TOC

3.2.7. "oth" (Other Primes Info) Parameter

TOC

The `oth` (other primes info) member contains an array of information about any third and subsequent primes, should they exist. When only two primes have been used (the normal case), this parameter MUST be omitted. When three or more primes have been used, the number of array elements MUST be the number of primes used minus two. Each array element MUST be an object with the following members:

3.2.7.1. "r" (Prime Factor)

TOC

The `r` (prime factor) parameter within an `oth` array member represents the value of a subsequent prime factor, a positive integer. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array.

3.2.7.2. "d" (Factor CRT Exponent)

TOC

The `d` (Factor CRT Exponent) parameter within an `oth` array member represents the CRT exponent of the corresponding prime factor, a positive integer. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array.

3.2.7.3. "t" (Factor CRT Coefficient)

TOC

The `t` (factor CRT coefficient) parameter within an `oth` array member represents the CRT coefficient of the corresponding prime factor, a positive integer. It is represented as the base64url encoding of the value's unsigned big endian representation as a byte array.

---

4. JWK Parameters for Symmetric Keys

TOC

When the JWK `kty` member value is `oct` (octet sequence), the following member MAY be used to represent a symmetric key (or another key whose value is a single octet sequence):

4.1. "k" (Key Value) Parameter

TOC

The `k` (key value) member contains the value of the symmetric (or other single-valued) key. It is represented as the base64url encoding of the octet sequence containing the key value.

---

5. Example Private Keys

TOC

The following example JWK Set contains two keys represented as JWKs containing both public and private key values: one using an Elliptic Curve algorithm and a second one using an RSA algorithm. This example extends the example in Section 3 of **[JWK]**, adding private key values. (Line breaks are for display purposes only.)

```
{"keys":
  [
    {"kty":"EC",
     "crv":"P-256",
     "x":"MKBCTNIcKUSDii11ySs3526iDZ8AiTo7Tu6KPAqv7D4",
     "y":"4Etl6SRW2YiLUrN5vfvVHuhp7x8PxltmWWlbbM4IFyM",
     "d":"870MB6gfuTJ4HtUnUvYMyJpr5eUZNP4Bk43bVdj3eAE",
```

```
      "use":"enc",
      "kid":"1"},

    {"kty":"RSA",
      "n":"0vx7agoebGcQSuuPiLJXZptN9nndrQmbXEps2aiAFbWhM78LhWx4
cbbfAAtVT86zwu1RK7aPFFxuhDR1L6tSoc_BJECPebWKRXjBZCiFV4n3oknjhMst
n64tZ_2W-5JsGY4Hc5n9yBXArwl93lqt7_RN5w6Cf0h4QyQ5v-65YGjQR0_FDW2Q
vzqY368QQMicAtaSqzs8KJZgnYb9c7d0zgdAZHzu6qMQvRL5hajrn1n91CbOpbIS
D08qNLyrdkt-bFTWhAI4vMQFh6WeZu0fM4lFd2NcRwr3XPksINHaQ-G_xBniIqbw
0Ls1jF44-csFCur-kEgU8awapJzKnqDKgw",
      "e":"AQAB",
      "d":"X4cTteJY_gn4FYPsXB8rdXix5vwsg1FLN5E3EaG6RJoVH-HLLKD9
M7dx5oo7GURknchnrRweUkC7hT5fJLM0WbFAKNLWY2vv7B6NqXSzUvxT0_YSfqij
wp3RTzlBaCxWp4doFk5N2o8Gy_nHNKroADIkJ46pRUohsXywbReAdYaMwFs9tv8d
_cPVY3i07a3t8MN6TNwm0dSawm9v47UiCl3Sk5ZiG7xojPLu4sbg1U2jx4IBTNBz
nbJSzFHK66jT8bgkuqsk0GjskDJk19Z4qwjwbsnn4j2WBii3RL-Us2lGVkY8fkFz
me1z0HbIkfz0Y6mqnOYtqc0X4jfcKoAC8Q",
      "p":"83i-7IvMGXoMXCskv73TKr8637FiO7Z27zv8oj6pbWUQyLPQBQxtPV
nwD20R-60eTDmD2ujnMt5PoqMrm8RfmNhVWDtjjMmCMjOpSXicFHj7XOuVIYQyqV
WlWEh6dN36GVZYk93N8Bc9vY41xy8B9RzzOGVQzXvNEvn7O0nVbfs",
      "q":"3dfOR9cuYq-0S-mkFLzgItgMEfFzB2q3hWehMuG0oCuqnb3vobLyum
qjVZQO1dIrdwgTnCdpYzBcOfW5r370AFXjiWft_NGEiovonizhKpo9VVS78TzFgx
kIdrecRezsZ-1kYd_s1qDbxtkDEgfAITAG9LUnADun4vIcb6yelxk",
      "dp":"G4sPXkc6Ya9y8oJW9_ILj4xuppu0lzi_H7VTkS8xj5SdX3coE0oim
YwxIi2emTAue0UOa5dpgFGyBJ4c8tQ2VF402XRugKDTP8akYhFo5tAA77Qe_Nmtu
YZc3C3m3I24G2GvR5sSDxUyAN2zq8Lfn9EUms6rY3Ob8YeiKkTiBj0",
      "dq":"s9lAH9fggBsoFR8Oac2R_E2gw282rT2kGOAhvIllETE1efrA6huUU
vMfBcMpn8lqeW6vzznYY5SSQF7pMdC_agI3nG8Ibp1BUb0JUiraRNqUfLhcQb_d9
GF4Dh7e74WbRsobRonujTYN1xCaP6TO61jvWrX-L18txXw494Q_cgk",
      "qi":"GyM_p6JrXySiz1toFgKbWV-JdI3jQ4ypu9rbMWx3rQJBfmt0FoYzg
UIZEVFEcOqwemRN81zoDAaa-Bk0KWNGDjJHZDdDmFhW3AN7lI-puxk_mHZGJ11rx
yR8O55XLSe3SPmRfKwZI6yU24ZxvQKFYItdldUKGzO6Ia6zTKhAVRU",
      "alg":"RS256",
      "kid":"2011-04-29"}
  ]
}
```

---

6. Example Symmetric Keys

TOC

The following example JWK Set contains two symmetric keys represented as JWKs: one designated as being for use with the AES Key Wrap algorithm and a second one that is an HMAC key. (Line breaks are for display purposes only.)

```
{"keys":
  [
    {"kty":"oct",
      "alg":"A128KW",
      "k":"GawgguFyGrWKav7AX4VKUg"},

    {"kty":"oct",
      "k":"AyM1SysPpbyDfgZld3umj1qzKObwVMkoqQ-EstJQLr_T-1qS0gZH75
aKtMN3Yj0iPS4hcgUuTwjAzZr1Z9CAow",
      "kid":"HMAC key used in JWS A.1 example"}
  ]
}
```

---

7. IANA Considerations

TOC

7.1. JSON Web Key Types Registration

TOC

This specification registers the key type defined in **Section 4** in the IANA JSON Web Key Types registry **[JWA]**.

7.1.1. Registry Contents

TOC

- "kty" Parameter Value: `oct`
- Implementation Requirements: RECOMMENDED+
- Change Controller: IETF
- Specification Document(s): **Section 4** of [[this document]]

7.2. JSON Web Key Parameters Registration

TOC

This specification registers the parameter names defined in **Section 3.1**, **Section 3.2**, and **Section 4** in the IANA JSON Web Key Parameters registry **[JWK]**.

7.2.1. Registry Contents

TOC

- Parameter Name: `d`
- Change Controller: IETF
- Specification Document(s): **Section 3.1.1** of [[this document]]

- Parameter Name: `d`
- Change Controller: IETF
- Specification Document(s): **Section 3.2.1** of [[this document]]

- Parameter Name: `p`
- Change Controller: IETF
- Specification Document(s): **Section 3.2.2** of [[this document]]

- Parameter Name: `q`
- Change Controller: IETF
- Specification Document(s): **Section 3.2.3** of [[this document]]

- Parameter Name: `dp`
- Change Controller: IETF
- Specification Document(s): **Section 3.2.4** of [[this document]]

- Parameter Name: `dq`
- Change Controller: IETF
- Specification Document(s): **Section 3.2.5** of [[this document]]

- Parameter Name: `qi`
- Change Controller: IETF
- Specification Document(s): **Section 3.2.6** of [[this document]]

- Parameter Name: `oth`
- Change Controller: IETF
- Specification Document(s): **Section 3.2.7** of [[this document]]

- Parameter Name: `k`
- Change Controller: IETF
- Specification Document(s): **Section 4.1** of [[this document]]

---

8. Security Considerations

TOC

All of the security issues faced by any cryptographic application must be faced by a JWS/JWE/JWK agent. Among these issues are protecting the user's private and symmetric keys, preventing various attacks, and helping the user avoid mistakes such as inadvertently encrypting a message for the wrong recipient. The entire list of security considerations is beyond the scope of this document.

Private and symmetric keys must be protected from disclosure to unintended parties. One recommended means of doing so is to encrypt JWKs or JWK Sets containing them by using the JWK or JWK Set value as the plaintext of a JWE.

A key is no more trustworthy than the method by which it was received.

The security considerations in **RFC 3447** [RFC3447] and **RFC 6030** [RFC6030] about protecting private and symmetric keys also apply to this specification.

TOC

9. Normative References

[JWA] Jones, M., "JSON Web Algorithms (JWA)," draft-ietf-jose-json-web-algorithms (work in progress), December 2012 (HTML).

[JWK] Jones, M., "JSON Web Key (JWK)," draft-ietf-jose-json-web-key (work in progress), December 2012 (HTML).

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119, March 1997 (TXT, HTML, XML).

[RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1," RFC 3447, February 2003 (TXT).

[RFC4627] Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)," RFC 4627, July 2006 (TXT).

[RFC6030] Hoyer, P., Pei, M., and S. Machani, "Portable Symmetric Key Container (PSKC)," RFC 6030, October 2010 (TXT).

TOC

Appendix A. Acknowledgements

John Bradley and James Manger contributed to the contents of this specification.

TOC

Appendix B. Document History

[[to be removed by the RFC editor before publication as an RFC]]

-00

- Created draft-jones-jose-json-private-and-symmetric-key from draft-jones-jose-json-private-key, adding a representation for symmetric keys.
- Tracked parameter changes and additions in the JWK spec.
- Recommend encryption of JWKs and JWK Sets containing private or symmetric keys as JWEs.
- Added seriesInfo information to Internet Draft references.

draft-jones-jose-json-private-key-01

- Changed the names of the RSA key parameters so that the identifiers are the same as those used in RFC 3447.
- Added the RSA private key fields enabling Chinese Remainder Theorem (CRT) calculations, based upon their use in RFC 3447.

draft-jones-jose-json-private-key-00

- Created draft-jones-jose-json-private-key to facilitate discussion of the question from the W3C WebCrypto WG to the IETF JOSE WG of whether JOSE plans to support a format for representing private keys.

TOC

Author's Address

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