Network Working Group	M.B. Jones
Internet-Draft	Microsoft
Intended status: Standards Track	October 31, 2011
Expires: May 03, 2012	

JSON Web Key (JWK)
draft-jones-json-web-key-02

Abstract

A JSON Web Key (JWK) is a JSON data structure that represents a set of public keys.

Requirements Language

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 RFC 2119 *[RFC2119]*.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

A JSON Web Key (JWK) is a JSON data structure that represents a set of public keys as a JSON object [RFC4627]. The JWK format is used to represent bare keys; representing certificate chains is an explicit non-goal of this specification. JSON Web Keys are referenced in JSON Web Signatures (JWSs) [JWS] using the jku (JSON Key URL) header parameter.

2. Terminology

JSON Web Key (JWK) A JSON data structure that represents a set of public keys. A JWK consists of a single JWK Container Object that contains an array of JWK Key Objects.

JWK Container Object
: A JSON object that contains an array of JWK Key Objects as a member.

JWK Key Object
: A JSON object that represents a single public key.

Base64url Encoding
: For the purposes of this specification, this term always refers to the URL- and filename-safe Base64 encoding described in RFC 4648 *[RFC4648]*, Section 5, with the (non URL-safe) '=' padding characters omitted, as permitted by Section 3.2. (See Appendix C of [JWS] for notes on implementing base64url encoding without padding.)

3. JSON Web Key (JWK) Overview

It is sometimes useful to be able to reference public key representations, for instance, in order to verify the signature on content signed with the corresponding private key. The JSON Web Key (JWK) data structure provides a convenient JSON representation for sets of public keys utilizing either the Elliptic Curve or RSA families of algorithms.

3.1. Example JWK

The following example JWK contains two public keys: one using an Elliptic Curve algorithm and a second one using an RSA algorithm. In both cases, integers are represented using the base64url encoding of their big endian representations.

```
{"keyvalues":
   [
     {"algorithm":"EC",
      "curve":"P-256",
      "x":"MKBCTNIcKUSDii11ySs3526iDZ8AiTo7Tu6KPAqv7D4",
      "y":"4Etl6SRW2YiLUrN5vfvVHuhp7x8PxltmWWlbbM4IFyM",
      "use":"encryption",
      "keyid":"1"},

    {"algorithm":"RSA",
      "modulus": "0vx7agoebGcQSuuPiLJXZptN9nndrQmbXEps2aiAFbWhM78LhWx4cbbfAAtVT86zwu1RK7aPFx
      "exponent":"AQAB",
      "keyid":"2011-04-29"}
   ]
 }
```

4. JWK Format

A JWK consists of a JWK Container Object, which is a JSON object that contains an array of JWK Key Objects as a member. This section specifies the format of these objects.

4.1. JWK Container Object Format

A JWK Container Object is a JSON object containing a specific member. This member is:

Member Name	JSON Value Type	Container Object Member Semantics
keyvalues	array	The keyvalues member value contains an array of JWK Key Objects. This member is REQUIRED.

JWK Container Object Member

Additional members MAY be present in the JWK Container Object. If present, they MUST be understood by implementations using that JWK.

4.2. JWK Key Object Format

A JWK Key Object is a JSON object containing specific members. Those members that are common to all key types are as follows:

Member Name	JSON Value Type	Key Object Member Semantics
algorithm	string	The algorithm member identifies the cryptographic algorithm family used with the key. Values defined by this specification are EC and RSA. Specific additional members are required to represent the key, depending upon the algorithm value. The algorithm value is case sensitive. This member is REQUIRED.
use	string	The use member identifies the intended use of the key. Values defined by this specification are signature and encryption. Other values MAY be used. The use value is case sensitive. This member is OPTIONAL.
keyid	string	The keyid (Key ID) member can be used to match a specific key. This can be used, for instance, to choose among a set of keys within the JWK during key rollover. The keyid value MAY correspond to a JWS kid value. The interpretation of the keyid value is unspecified. This member is OPTIONAL.

JWK Key Object Members

Additional members MAY be present in the JWK Key Object. If present, they MUST be understood by implementations using that key.

4.2.1. JWK Key Object Members for Elliptic Curve Keys

JWKs can represent Elliptic Curve [FIPS.186-3] keys. In this case, the algorithm member value MUST be EC. Furthermore, these additional members MUST be present:

Member Name	JSON Value Type	Key Object Member Semantics
curve	string	The curve member identifies the cryptographic curve used with the key. Values defined by this specification are P-256, P-384 and P-521. Additional curve values MAY be used, provided they are understood by implementations using that Elliptic Curve key. The curve value is case sensitive.
x	string	The x member contains the x coordinate for the elliptic curve point. It is represented as the base64url encoding of the coordinate's big endian representation.
y	string	The y member contains the y coordinate for the elliptic curve point. It is represented as the base64url encoding of the coordinate's big endian representation.

Members for Elliptic Curve Keys

4.2.2. JWK Key Object Members for RSA Keys

JWKs can represent RSA [RFC3447] keys. In this case, the algorithm member value MUST be RSA. Furthermore, these additional members MUST be present:

Member Name	JSON Value Type	Key Object Member Semantics
modulus	string	The modulus member contains the modulus value for the RSA public key. It is represented as the base64url encoding of the value's big endian representation.
exponent	string	The exponent member contains the exponent value for the RSA public key. It is represented as the base64url encoding of the value's big endian representation.

Members for RSA Keys

5. Base64url encoding as used by JWKs

JWKs make use of the base64url encoding as defined in RFC 4648 *[RFC4648]*. As allowed by Section 3.2 of the RFC, this specification mandates that base64url encoding when used with JWKs MUST NOT use

padding. Notes on implementing base64url encoding can be found in the JWS [JWS] specification.

6. IANA Considerations

No IANA actions are required by this specification.

7. Security Considerations

TBD

8. Open Issues and Things To Be Done (TBD)

The following items remain to be done in this draft:

- *Write the Security Considerations section.

9. References

9.1. Normative References

[RFC2119]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
[RFC3447]	Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, February 2003.
[RFC4627]	Crockford, D., "The application/json Media Type for JavaScript Object Notation (JSON)", RFC 4627, July 2006.
[RFC4648]	Josefsson, S., "The Base16, Base32, and Base64 Data Encodings", RFC 4648, October 2006.
[FIPS. 186-3]	National Institute of Standards and Technology, "Digital Signature Standard (DSS)", FIPS PUB 186-3, June 2009.

9.2. Informative References

[MagicSignatures]	Panzer (editor), J., Laurie, B. and D. Balfanz, "Magic Signatures", August 2010.
[JWS]	Jones, M.B., Balfanz, D., Bradley, J., Goland, Y.Y., Panzer, J., Sakimura, N. and P. Tarjan, "JSON Web Signature (JWS)", October 2011.

Appendix A. Acknowledgements

A JSON representation for RSA public keys was previously introduced in Magic Signatures *[MagicSignatures]*.

Appendix B. Document History

-02

*Editorial changes to have this spec better match the JWT, JWS, and JWE specs. No normative changes.

-01

*Changed algorithm member value for Elliptic Curve keys from ECDSA to EC, since Elliptic Curve keys can be used with more algorithms than just the Elliptic Curve Digital Signature Algorithm (ECDSA).

*Added OPTIONAL use member to identify intended key usage, especially since the same Elliptic Curve key should not be used for both signing and encryption operations.

-00

*Created first version based upon decisions made at the Internet Identity Workshop (IIW), as documented at http://self-issued.info/?p=390.

Author's Address

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