JSON Web Key (JWK)
draft-ietf-jose-json-web-key-09

Abstract

A JSON Web Key (JWK) is a JavaScript Object Notation (JSON) data structure that represents a cryptographic key. This specification also defines a JSON Web Key Set (JWK Set) JSON data structure for representing a set of JWKs. Cryptographic algorithms and identifiers for use with this specification are described in the separate JSON Web Algorithms (JWA) specification.

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 October 25, 2013.

Copyright Notice

Copyright (c) 2013 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 (http://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. Notational Conventions
2. Terminology
3. JSON Web Key (JWK) Format
   3.1. "kty" (Key Type) Parameter
   3.2. "use" (Key Use) Parameter
   3.3. "alg" (Algorithm) Parameter
   3.4. "kid" (Key ID) Parameter
4. JSON Web Key Set (JWK Set) Format
   4.1. "keys" (JSON Web Key Set) Parameter
5. String Comparison Rules
6. IANA Considerations
   6.1. JSON Web Key Parameters Registry
       6.1.1. Registration Template
       6.1.2. Initial Registry Contents
1. Introduction

A JSON Web Key (JWK) is a JavaScript Object Notation (JSON) [RFC4627] data structure that represents a cryptographic key. This specification also defines a JSON Web Key Set (JWK Set) JSON data structure for representing a set of JWKs. Cryptographic algorithms and identifiers for use with this specification are described in the separate JSON Web Algorithms (JWA) [JWA] specification.

Goals for this specification do not include representing certificate chains, representing certified keys, and replacing X.509 certificates.

JWKs and JWK Sets are used in the JSON Web Signature (JWS) [JWS] and JSON Web Encryption (JWE) [JWE] specifications.

1.1. Notational Conventions

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

JSON Web Key (JWK)
A JSON object that represents a cryptographic key.

JSON Web Key Set (JWK Set)
A JSON object that contains an array of JWKs as the value of its keys member.

Base64url Encoding
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.)

Collision Resistant Namespace
A namespace that allows names to be allocated in a manner such that they are highly unlikely to collide with other names. For instance, collision resistance can be achieved through administrative delegation of portions of the namespace or through use of collision-resistant name allocation functions. Examples of Collision Resistant Namespaces include: Domain Names, Object Identifiers (OIDs) as defined in the ITU-T X.660 and X.670 Recommendation series, and Universally Unique Identifiers (UUIDs) [RFC4122]. When using an administratively delegated namespace, the definer of a name needs to take reasonable precautions to ensure they are in control of the portion of the namespace they use to define the name.
3. JSON Web Key (JWK) Format

A JSON Web Key (JWK) is a JSON object containing specific members, as specified below. Those members that are common to all key types are defined below.

In addition to the common parameters, each JWK will have members that are specific to the key being represented. These members represent the parameters of the key. Section 5 of the JSON Web Algorithms (JWA) [JWA] specification defines multiple kinds of cryptographic keys and their associated members.

The member names within a JWK MUST be unique; objects with duplicate member names MUST be rejected.

Additional members MAY be present in the JWK. If not understood by implementations encountering them, they MUST be ignored. Member names used for representing key parameters for different kinds of keys need not be distinct. Any new member name SHOULD either be registered in the IANA JSON Web Key Parameters registry Section 6.1 or be a value that contains a Collision Resistant Namespace.

3.1. "kty" (Key Type) Parameter

The kty (key type) member identifies the cryptographic algorithm family used with the key. kty values SHOULD either be registered in the IANA JSON Web Key Types registry [JWA] or be a value that contains a Collision Resistant Namespace. The kty value is a case sensitive string. Use of this member is REQUIRED.

A list of defined kty values can be found in the IANA JSON Web Key Types registry [JWA]; the initial contents of this registry are the values defined in Section 5.1 of the JSON Web Algorithms (JWA) [JWA] specification.

Additional members used with these kty values can be found in the IANA JSON Web Key Parameters registry Section 6.1; the initial contents of this registry are the values defined in Sections 5.2 and 5.3 of the JSON Web Algorithms (JWA) [JWA] specification.

3.2. "use" (Key Use) Parameter

The use (key use) member identifies the intended use of the key. Values defined by this specification are:

- sig (signature or MAC operation)
- enc (encryption)

Other values MAY be used. The use value is a case sensitive string. Use of this member is OPTIONAL.

3.3. "alg" (Algorithm) Parameter

The alg (algorithm) member identifies the algorithm intended for use with the key. The values used in this field are the same as those used in the JWS [JWS] and JWE [JWE] alg and enc header parameters; these values can be found in the JSON Web Signature and Encryption Algorithms registry [JWA]. Use of this member is OPTIONAL.

3.4. "kid" (Key ID) Parameter
The **kid** (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 a JWK Set during key rollover. The interpretation of the **kid** value is unspecified. Key ID values within a JWK Set need not be unique. The **kid** value is a case sensitive string. Use of this member is **OPTIONAL**.

When used with JWS or JWE, the **kid** value can be used to match a JWS or JWE **kid** header parameter value.

In some contexts, different keys using the same Key ID value might be present, with the keys being disambiguated using other information, such as the **kty** or **use** values. For example, imagine **kid** values like "Current", "Upcoming", and "Deprecated", used for key rollover guidance. One could apply a label to all keys where the classification fits. If there are multiple "Current" keys, then in this example, they might be differentiated either by having different "kty" or "use" values, or some combination of both. As one example, there might only be one current RSA signing key and one current Elliptic Curve signing key, but both would be "Current".

### 4. JSON Web Key Set (JWK Set) Format

A JSON Web Key Set (JWK Set) is a JSON object that contains an array of JSON Web Key values as the value of its **keys** member.

The member names within a JWK Set MUST be unique; objects with duplicate member names MUST be rejected.

Additional members MAY be present in the JWK Set. If not understood by implementations encountering them, they MUST be ignored. Parameters for representing additional properties of JWK Sets SHOULD either be registered in the IANA JSON Web Key Set Parameters registry Section 6.2 or be a value that contains a Collision Resistant Namespace.

#### 4.1. "keys" (JSON Web Key Set) Parameter

The value of the **keys** (JSON Web Key Set) member is an array of JSON Web Key (JWK) values. Use of this member is **REQUIRED**.

### 5. String Comparison Rules

Processing a JWK inevitably requires comparing known strings to values in JSON objects. For example, in checking what the key type is, the Unicode string encoding **kty** will be checked against the member names in the JWK to see if there is a matching name.

Comparisons between JSON strings and other Unicode strings MUST be performed by comparing Unicode code points without normalization as specified in the String Comparison Rules in Section 5.3 of [JWS].

### 6. IANA Considerations

The following registration procedure is used for all the registries established by this specification.

Values are registered with a Specification Required [RFC5226] after a two-week review period on the [TBD]@ietf.org mailing list, on the advice of one or more Designated Experts. However, to allow for the allocation of values prior to publication, the Designated Expert(s) may approve registration once they are satisfied that such a specification will be published.

Registration requests must be sent to the [TBD]@ietf.org mailing list for review and comment, with an appropriate subject (e.g., "Request for access token type: example").
Note to RFC-EDITOR: The name of the mailing list should be determined in consultation with the IESG and IANA. Suggested name: jose-reg-review.

Within the review period, the Designated Expert(s) will either approve or deny the registration request, communicating this decision to the review list and IANA. Denials should include an explanation and, if applicable, suggestions as to how to make the request successful.

IANA must only accept registry updates from the Designated Expert(s) and should direct all requests for registration to the review mailing list.

6.1. JSON Web Key Parameters Registry

This specification establishes the IANA JSON Web Key Parameters registry for reserved JWK parameter names. The registry records the reserved parameter name and a reference to the specification that defines it. This specification registers the parameter names defined in Section 3. The same JWK parameter name may be registered multiple times, provided that duplicate parameter registrations are only for algorithm-specific JWK parameters; in this case, the meaning of the duplicate parameter name is disambiguated by the kty value of the JWK containing it.

6.1.1. Registration Template

Parameter Name:
The name requested (e.g., "example"). This name is case sensitive. Names that match other registered names in a case insensitive manner SHOULD NOT be accepted.
Change Controller:
For Standards Track RFCs, state "IETF". For others, give the name of the responsible party. Other details (e.g., postal address, email address, home page URI) may also be included.
Specification Document(s):
Reference to the document(s) that specify the parameter, preferably including URI(s) that can be used to retrieve copies of the document(s). An indication of the relevant sections may also be included but is not required.

6.1.2. Initial Registry Contents

- Parameter Name: kty
  Change Controller: IETF
  Specification Document(s): Section 3.1 of [[ this document ]]
- Parameter Name: use
  Change Controller: IETF
  Specification Document(s): Section 3.2 of [[ this document ]]
- Parameter Name: alg
  Change Controller: IETF
  Specification Document(s): Section 3.3 of [[ this document ]]
- Parameter Name: kid
  Change Controller: IETF
  Specification Document(s): Section 3.4 of [[ this document ]]

6.2. JSON Web Key Set Parameters Registry

This specification establishes the IANA JSON Web Key Set Parameters registry for reserved JWK Set parameter names. The registry records the reserved parameter name and a
reference to the specification that defines it. This specification registers the parameter names defined in Section 4.

6.2.1. Registration Template

Parameter Name:
The name requested (e.g., "example"). This name is case sensitive. Names that match other registered names in a case insensitive manner SHOULD NOT be accepted.

Change Controller:
For Standards Track RFCs, state "IETF". For others, give the name of the responsible party. Other details (e.g., postal address, email address, home page URI) may also be included.

Specification Document(s):
Reference to the document(s) that specify the parameter, preferably including URI(s) that can be used to retrieve copies of the document(s). An indication of the relevant sections may also be included but is not required.

6.2.2. Initial Registry Contents

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

7. Security Considerations

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, but some significant considerations are listed here.

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

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.

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

The security considerations in XML DSIG 2.0 [W3C.CR-xmldsig-core2-20120124], about key representations also apply to this specification, other than those that are XML specific.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," BCP 14, RFC 2119,
8.2. Informative References


Appendix A. Example JSON Web Key Sets

A.1. Example Public Keys

The following example JWK Set contains two public keys represented as JWKs: one using an Elliptic Curve algorithm and a second one using an RSA algorithm. The first specifies that the key is to be used for encryption. The second specifies that the key is to be used with the RS256 algorithm. Both provide a Key ID for key matching purposes. In both cases, integers are represented using the base64url encoding of their big endian representations. (Long lines are broken are for display purposes only.)

```json
{"keys": [  
  {"kty":"EC",
   "crv":"P-256",
   "x":"MKCCTN1ckUSDIi1ySs3526iDZBAlTo7Tu6KPAqv7D4",
   "y":"4Etl6SRWzYiLurN5vfvVHuHP7x8PxlTMWHbM4IfyM",
   "use":"enc",
   "kid":"1"},

  {"kty":"RSA",
   "n": "0vx7agoebGcQSuuPILJXZptN9ndrQmbXEpSze1Af8w8M78LhwX4cbbfAATV86zu1RK7aPFFxuhDR1LtSoc_BJECPebWKR3xBZCIfVn30knjhmStn64tZ_2w-5JsGY4Hc5n9yBXArw193lqT7_RN5w6Cf0h4QuyQ5v-65YGjQR0_fdw2QvqzY36BQ0MicAaSs8KJZgnYb9c7d8zgdAHzu6qMQvRL5hajrn1n91C0bIPD08qANlyrkt-bFTwhAIVmfH6WeZufM41Fd28Cnrwr3XPKsINHaQ-G_xBnIqbw0Ls1jF44-csFCur-kEgU8awapJzK8qqDKgw",
   "e":"AQAB",
   "alg":"RS256",
   "kid":"2011-04-29"}  
]  
}
```

A.2. Example Private Keys

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 the previous section, adding private key...
A.3. Example Symmetric Keys

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.)

```json
{"keys": [
    {
      "kty": "EC",
      "crv": "P-256",
      "x": "MKBCTNicKUSDii1ySs3526iDZ8AiT07Tu6KPAqV7d4",
      "y": "e4Tl6SRWYiUrN5vfvVHup7x8P6ltmWl1bbM4IfyM",
      "d": "876MB6gfUt3J4htUn4VMYy3Pr5eUZN4P84K43bvd3j3eAE",
      "use": "enc",
      "kid": "1",
    },
    {
      "kty": "RSA",
      "n": "0vx7agoebGcQSuuiLiJXZptN9ndrQmbXEpS2aiAFbWhm78LhWx4cbbfAAtVb6zwu1RK7aPPFxfUdDR1L6tSoc_BJECpbWBKRXjBZCiFV4n3oknjMhsn64tZ_2W-5Js6Y4hc5n9yBAXArw19i3q77_RNSw6Cfnh4QyQ5v-65YGyJQR0_FDW2Qvzy368Q0MicAaTszs8KJ2gynY9c792dogc2HzuY66MQvRL5hajrn191cbOpbISD08qNLyrldk-bFTwHAI4vM0Qf6W6ZuTofM41Fd2NiCwr3XPksINHAQ-G_bNl1qbw0LsljF44-csFCur-kEGBawj2zKndKdgw",
      "e": "AQAB",
      "d": "X4cTejY_gn4FYPsXBbrdXiz5vwsq1FLN5E3ea6GRJoVH-LLKD9M7dxs007GURknchnrRweUKahT7F5LMWvBFAKNLWvYv7bB6NqXsZuUvxtV0_YSfqiJwp3RT2laBcXwp4doFksN2o8Gh_nHNKRoADKjJ46pRUohsXywbReAdYaMwFs9tv8dc_CPVY3i67a3t8M6N67Wnm6dSawm9v47u1C135Kz1G7xozPlu4sbgi1U2jx4IBTBNbznbJ5SzHkK66jT8bBkgppskG9jskDj19Z4qvwjbsnn4j2Wbi13RL-U521GvakY8KfKZeIe0Gb1fzY6mnq0Ytqcg6X4jafcKoaAC8Q",
      "p": "33-1IvMGMx0MCSx73TKr6837f072Z7zv8o7j6pbWUyLPQbXqxtPVmeWd2R0-60TDmDuIjMtp6PqmrB6fMnhVWdtjMmCmMoP5SxIcfHj7X0u5yQyvqVILWhEdn3063Vv8YK39N88c9ygY41xyB9Rrz0GVQxvXNeVn700nVbfs",
      "q": "3df0R9cuqy-0S-mkFLoZtTmgw7rZwweaMu6oCuqnb3vobLyumq1VZQ01rdrWgTNcDpyzBc0fW5r370AFXJjwFt_N3EivonizhKpo0V7V8T7zFgxkIYreCresz-1kyd_s1qDbxhKDeqfAIAA9G1UnAdun4Vcb6yelxK",
      "dp": "G4spKxc6Ya9y8oJW_J14xuuph0lzi_H7vTKs8Jx5J5dX3ceOEOimYwXi2emtaUea60uaDpfgfyBj4c8tQ2VF402XRugKDTp8akYHfo5tAA77q_NmuTyz3C33m124G2Vrs5DUXyA2nZq28fln9EUMs6y3Ob8YelK1t1bJ0",
      "dq": "s1aAHfqqggBsw0RF8ooac2R_Ezgw28zr2TkG0ahvU11TE1eFaf6hUuVMf8Cmnp81qevWvzzyY55SQF7pMc_addi3nG8bpi1BuB0TJiurarNQULhcbq_d9G4Wdhef754wRsoBunjUTyN1xcP6TO6ijwR-x-L1htxXw494Q_cqK",
      "qi": "Gym_p6jrXySiz1toFgKbWV-6D3j3q4yupu9rB0MwXs3QbftmF0YzgUIZEVFe0cQwemRNN81zoOAa-Bk0KwNqDjJHzdDmFw3AN71-I-puxK_mHZGJllrxyR0855XLSe3SmPmRfkwZI6yU24ZxvQKFYItd1UKGp576IA6zTkhAVRU",
      "alg": "RS256",
      "kid": "2011-04-29"
    },
    {
      "kty": "oct",
      "alg": "A128KW",
      "k": "GawgguyFaKrWav7AX4VKug",
    },
    {
      "kty": "oct",
      "k": "AyM1SysPpbvDgefZld3umJ1qzKObwVMKoqQ-EstJQLr_T-1qS0gZH75akTm3N3YjioP5A4hcgUUtWjAzZr9CAow",
      "kid": "HMAC key used in JWS A.1 example"
    }
  ]
}
Appendix B. Acknowledgements

A JSON representation for RSA public keys was previously introduced by John Panzer, Ben Laurie, and Dirk Balfanz in Magic Signatures [MagicSignatures].

This specification is the work of the JOSE Working Group, which includes dozens of active and dedicated participants. In particular, the following individuals contributed ideas, feedback, and wording that influenced this specification:


Jim Schaad and Karen O'Donoghue chaired the JOSE working group and Sean Turner and Stephen Farrell served as Security area directors during the creation of this specification.

Appendix C. Document History

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

-09

- Expanded the scope of the JWK specification to include private and symmetric key representations, as specified by draft-jones-jose-json-private-and-symmetric-key-00.
- Defined that members that are not understood must be ignored.

-08

- Changed the name of the JWK key type parameter from alg to kty to enable use of alg to indicate the particular algorithm that the key is intended to be used with.
- Clarified statements of the form "This member is OPTIONAL" to "Use of this member is OPTIONAL".
- Referenced String Comparison Rules in JWS.
- Added seriesInfo information to Internet Draft references.

-07

- Changed the name of the JWK RSA modulus parameter from mod to n and the name of the JWK RSA exponent parameter from xpo to e, so that the identifiers are the same as those used in RFC 3447.

-06

- Changed the name of the JWK RSA exponent parameter from exp to xpo so as to allow the potential use of the name exp for a future extension that might define an expiration parameter for keys. (The exp name is already used for this purpose in the JWT specification.)
- Clarify that the alg (algorithm family) member is REQUIRED.
- Correct an instance of "JWK" that should have been "JWK Set".
- Applied changes made by the RFC Editor to RFC 6749's registry language to this specification.

-05

- Indented artwork elements to better distinguish them from the body text.

-04

- Refer to the registries as the primary sources of defined values and then
secondarily reference the sections defining the initial contents of the registries.

- Added this language to Registration Templates: "This name is case sensitive. Names that match other registered names in a case insensitive manner SHOULD NOT be accepted."
- Described additional open issues.
- Applied editorial suggestions.

-03

- Clarified that kid values need not be unique within a JWK Set.
- Moved JSON Web Key Parameters registry to the JWK specification.
- Added "Collision Resistant Namespace" to the terminology section.
- Changed registration requirements from RFC Required to Specification Required with Expert Review.
- Added Registration Template sections for defined registries.
- Added Registry Contents sections to populate registry values.
- Numerous editorial improvements.

-02

- Simplified JWK terminology to get replace the "JWK Key Object" and "JWK Container Object" terms with simply "JSON Web Key (JWK)" and "JSON Web Key Set (JWK Set)" and to eliminate potential confusion between single keys and sets of keys. As part of this change, the top-level member name for a set of keys was changed from jwk to keys.
- Clarified that values with duplicate member names MUST be rejected.
- Established JSON Web Key Set Parameters registry.
- Explicitly listed non-goals in the introduction.
- Moved algorithm-specific definitions from JWK to JWA.
- Reformatted to give each member definition its own section heading.

-01

- Corrected the Magic Signatures reference.

-00

- Created the initial IETF draft based upon draft-jones-json-web-key-03 with no normative changes.

Author's Address

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