

JOSE Working Group	M. Jones
Internet-Draft	Microsoft
Intended status: Standards Track	J. Bradley
Expires: September 6, 2012	independent
	N. Sakimura
	Nomura Research Institute
	March 5, 2012

JSON Web Signature JSON Serialization (JWS-JS) draft-jones-json-web-signature-json-serialization-00

Abstract

The JSON Web Signature JSON Serialization (JWS-JS) is a means of representing content secured with digital signatures or Hash-based Message Authentication Codes (HMACs) using JSON data structures. This specification describes a means of representing secured content as a JSON data object (as opposed to the JWS specification, which uses a compact serialization with a URL-safe representation). It enables multiple digital signatures and/or HMACs to be applied to the same content (unlike JWS). Cryptographic algorithms and identifiers used with this specification are enumerated in the separate JSON Web Algorithms (JWA) specification. The JSON Serialization for related encryption functionality is described in the separate JSON Web Encryption JSON Serialization (JWE-JS) specification.

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

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 September 6, 2012.

Copyright Notice

Copyright (c) 2012 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.

- [1. Introduction](#)
- [2. Terminology](#)
- [3. JSON Serialization](#)
- [4. Example JWS-JS](#)
- [5. IANA Considerations](#)
- [6. Security Considerations](#)
- [7. References](#)
 - [7.1. Normative References](#)
 - [7.2. Informative References](#)
- [Appendix A. Acknowledgements](#)
- [Appendix B. Document History](#)
- [§ Authors' Addresses](#)

1. Introduction

TOC

The JSON Web Signature JSON Serialization (JWS-JS) is a format for representing content secured with digital signatures or Hash-based Message Authentication Codes (HMACs) as a JSON [\[RFC4627\]](#) object. It enables multiple digital signatures and/or HMACs to be applied to the same content (unlike JWS [\[JWS\]](#)). The digital signature and HMAC mechanisms used are independent of the type of content being secured, allowing arbitrary content to be secured. Cryptographic algorithms and identifiers used with this specification are enumerated in the separate JSON Web Algorithms (JWA) [\[JWA\]](#) specification. The JSON Serialization for related encryption functionality is described in the separate JSON Web Encryption JSON Serialization [\[JWE-JS\]](#) specification.

2. Terminology

TOC

This specification uses the same terminology as the JSON Web Signature (JWS) [\[JWS\]](#) specification.

3. JSON Serialization

TOC

The JSON Serialization represents secured content as a JSON object with members for each of three constituent parts: a `headers` member whose value is a non-empty array of Encoded JWS Header values, a `payload` member whose value is an Encoded JWS Payload value, and a `signatures` member whose value is a non-empty array of Encoded JWS Signature values, where the number of elements in both arrays is the same.

Unlike the compact serialization used by JWSs, content using the JSON Serialization MAY be secured with more than one digital signature and/or HMAC value. Each is represented as an Encoded JWS Signature in the `signatures` member array. For each, there is a corresponding `headers` member array element that is an Encoded JWS Header specifying the digital signature or HMAC applied to the Encoded JWS Header value and the Encoded JWS Payload value to create the JWS Signature value. Therefore, the syntax is:

```
{ "headers": [ "<header 1 contents>", ..., "<header N contents>" ],  
  "payload": "<payload contents>",  
  "signatures": [ "<signature 1 contents>", ..., "<signature N contents>" ]  
}
```

The contents of the Encoded JWS Header, Encoded JWS Payload, and Encoded JWS Signature values are exactly as specified in JSON Web Signature (JWS) [\[JWS\]](#). They are interpreted and validated in the same manner, with each corresponding `headers` and `signatures` value being created or validated together. The arrays MUST have the same number of elements.

The *i*'th JWS Signature value is computed on the JWS Secured Input corresponding to the

concatenation of the *i*'th Encoded JWS Header, a period ('.') character, and the Encoded JWS Payload in the same manner described in the JWS specification. This has the desirable result that each Encoded JWS signature value in the `signatures` array is identical to the value that would be used for the same header and payload in a JWS.

4. Example JWS-JS

TOC

This section contains an example using the JWS JSON Serialization. This example demonstrates the capability for conveying multiple digital signatures and/or HMACs for the same payload.

The Encoded JWS Payload used in this example is the same as used in the examples in Appendix A of JWS (with line breaks for display purposes only):

```
eyJpc3MiOiJqb2UiLA0KICJleHAiOjEzMDA4MTkzODAsDQogImh0dHA6Ly9leGFtcGxlLmNvbS9pc19yb290Ijpb0cnV1fQ
```

Two digital signatures are used in this example: an RSA SHA-256 signature, for which the header and signature values are the same as in Appendix A.2 of JWS, and an ECDSA P-256 SHA-256 signature, for which the header and signature values are the same as in Appendix A.3 of JWS. The two Decoded JWS Header Segments used are:

```
{"alg": "RS256"}
```

and:

```
{"alg": "ES256"}
```

Since the computations of the JWS Header and JWS Signature values are the same as in Appendix A.2 and Appendix A.3 of JWS, they are not repeated here.

The complete JSON Web Signature JSON Serialization (JWS-JS) for these values is as follows (with line breaks for display purposes only):

```
{"headers": [
  "eyJhbGciOiJSUzI1NiJ9",
  "eyJhbGciOiJFUzI1NiJ9"},
  "payload": "eyJpc3MiOiJqb2UiLA0KICJleHAiOjEzMDA4MTkzODAsDQogImh0dHA6Ly9leGFtcGxlLmNvbS9pc19yb290Ijpb0cnV1fQ",
  "signatures": [
    "cC4hiUPoj9Eetdgtv3hF80EGrhuB__dzERat0XF9g2VtQgr9PJbu3X0izj5RZmh7AAuHI4Bh-0Qc_lF5YKt_08W2Fp5jujGbds9uJdbF9CUAr7t1dnZcAcQjbKBYNX4BAynRFdiuB--f_nZLgrnbyTyWz075vRK5h6xBARLIARNPvkSjtQBMH1b1L07Qe7K0GarZRmB_eSN9383Lc0Ln6_d0--xi12jzDwusC-e0kHWESqtFZESc6BfI7no0PqvhJ1phCnvWh6IeYI2w9Q0YEUipUTI8np6LbgGY9Fs98rqVt5AXLIhWkWyw1VmtVrBp0igcN_IoypG1UPQGe77Rw",
    "DtEhU31jbEg8L38VwAFUAq0yKAM6-Xx-F4GawxaepmXFCgftJdxw5djxLa8IS1SApmWQxfKTUJqPP3-Kg6NU1Q"]
}
```

5. IANA Considerations

TOC

This specification makes no requests of IANA.

6. Security Considerations

TOC

The security considerations for this specification are the same as those for the JSON Web Signature (JWS) **[JWS]** specification.

7. References

TOC

7.1. Normative References

TOC

- [JWA]** [Jones, M.](#), "[JSON Web Algorithms \(JWA\)](#)," January 2012.
- [JWS]** [Jones, M.](#), [Bradley, J.](#), and [N. Sakimura](#), "[JSON Web Signature \(JWS\)](#)," January 2012.
- [RFC2119]** [Bradner, S.](#), "[Key words for use in RFCs to Indicate Requirement Levels](#)," BCP 14, RFC 2119, March 1997 ([TXT](#), [HTML](#), [XML](#)).
- [RFC4627]** Crockford, D., "[The application/json Media Type for JavaScript Object Notation \(JSON\)](#)," RFC 4627, July 2006 ([TXT](#)).

7.2. Informative References

TOC

- [JSS]** Bradley, J. and N. Sakimura (editor), "[JSON Simple Sign](#)," September 2010.
- [JWE-JS]** [Jones, M.](#), "[JSON Web Encryption JSON Serialization \(JWE-JS\)](#)," March 2012.
- [MagicSignatures]** Panzer (editor), J., Laurie, B., and D. Balfanz, "[Magic Signatures](#)," August 2010.

Appendix A. Acknowledgements

TOC

JSON serializations for secured content were previously explored by **Magic Signatures** [MagicSignatures] and **JSON Simple Sign** [JSS].

Appendix B. Document History

TOC

-00

- Created the initial version incorporating JOSE working group input and drawing from the JSON Serialization previously proposed in draft-jones-json-web-token-01.

Authors' Addresses

TOC

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

John Bradley
independent
Email: ve7jtb@ve7jtb.com

Nat Sakimura
Nomura Research Institute
Email: n-sakimura@nri.co.jp