COSE Working Group
Internet-Draft
Intended status: Standards Track
Expires: November 19, 2017

M. Jones
Microsoft

May 18, 2017

Using RSA Algorithms with COSE Messages
draft-jones-cose-rsa-03

Abstract

The CBOR Object Signing and Encryption (COSE) specification defines cryptographic message encodings using Concise Binary Object Representation (CBOR). This specification defines algorithm encodings and representations enabling RSA algorithms to be used for COSE messages. Encodings for the use of RSASSA-PSS signatures, RSAES-OAEP encryption, and RSA keys are specified.

Table of Contents

1. Introduction

The CBOR Object Signing and Encryption (COSE) [I-D.ietf-cose-msg] specification defines cryptographic message encodings using Concise Binary Object Representation (CBOR) [RFC7049]. This specification defines algorithm encodings and representations enabling RSA algorithms to be used for COSE messages.

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 RFC 2119 [RFC2119].

2. RSASSA-PSS Signature Algorithm

The RSASSA-PSS signature algorithm is defined in [RFC3447].

The RSASSA-PSS signature algorithm is parameterized with a hash function (h), a mask generation function (mgf) and a salt length (sLen). For this specification, the mask generation function is fixed to be MGF1 as defined in [RFC3447]. It has been recommended

that the same hash function be used for hashing the data as well as in the mask generation function. This specification follows this recommendation. The salt length is the same length as the hash function output.

Implementations need to check that the key type is 'RSA' when creating or verifying a signature.

The algorithms defined in this document can be found in Table 1.

Name	Value	Hash	Salt Length	Description
PS256	-37	SHA-256	32	RSASSA-PSS w/ SHA-256
PS384	-38	SHA-384	48	RSASSA-PSS w/ SHA-384
PS512	-39	SHA-512	64	RSASSA-PSS w/ SHA-512

Table 1: RSASSA-PSS Algorithm Values

3. RSAES-OAEP Key Encryption Algorithm

RSAES-OAEP is an asymmetric key encryption algorithm. The definition of RSAEA-OAEP can be find in Section 7.1 of [RFC3447]. The algorithm is parameterized using a masking generation function (mgf), a hash function (h) and encoding parameters (P). For the algorithm identifiers defined in this section:

- mgf is always set to MFG1 from [RFC3447] and uses the same hash function as h.

- P is always set to the empty octet string.

Table 2 summarizes the rest of the values.

Name	Value	Hash	Description
RSAES-OAEP w/ RFC 3447 default parameters	-40	SHA-1	RSAES OAEP w/ SHA-1
RSAES-OAEP w/ SHA-256	-41	SHA-256	RSAES OAEP w/ SHA-256
RSAES-OAEP w/ SHA-512	-42	SHA-512	RSAES OAEP w/ SHA-512

Table 2: RSAES-OAEP Algorithm Values

The key type MUST be 'RSA'.

4. RSA Keys

Key types are identified by the 'kty' member of the COSE_Key object. This specification defines one value for this member.

Name	Value	Description
RSA	3	RSA Key

Table 3: Key Type Values

This document defines a key structure for both the public and private parts of RSA keys. Together, an RSA public key and an RSA private key form an RSA key pair.

The document also provides support for the so-called "multi-prime" RSA keys, in which the modulus may have more than two prime factors. The benefit of multi-prime RSA is lower computational cost for the decryption and signature primitives. For a discussion on how multi-prime affects the security of RSA crypto-systems, the reader is referred to [MultiPrimeRSA].

This document follows the naming convention of [RFC3447] for the naming of the fields of an RSA public or private key. Table 4 provides a summary of the label values and the types associated with each of those labels. The requirements for fields for RSA keys are as follows:

- For all keys, 'kty' MUST be present and MUST have a value of 3.
- For public keys, the fields 'n' and 'e' MUST be present. All other fields defined in Table 4 MUST be absent.
- For private keys with two primes, the fields 'other', 'r_i', 'd_i' and 't_i' MUST be absent; all other fields MUST be present.
- For private keys with more than two primes, all fields MUST be present. For the third to nth primes, each of the primes is represented as a map containing the fields 'r_i', 'd_i' and 't_i'. The field 'other' is an array of those maps.
- All numeric key parameters are encoded in an unsigned big-endian representation as an octet sequence using the CBOR byte string type (major type 2). The octet sequence MUST utilize the minimum

number of octets needed to represent the value. For instance, the value 32,768 is represented as the CBOR byte sequence 0b010_00010 (major type 2, additional information 2 for the length), 0x80 0x00.

Name	Key Type	Value	Type	Description
n	3	-1	bstr	Modulus Parameter
e	3	-2	bstr	Exponent Parameter
d	3	-3	bstr	Private Exponent Parameter
p	3	-4	bstr	First Prime Factor
q	3	-5	bstr	Second Prime Factor
dP	3	-6	bstr	First Factor CRT Exponent
dQ	3	-7	bstr	Second Factor CRT Exponent
qInv	3	-8	bstr	First CRT Coefficient
other	3	-9	array	Other Primes Info
r_i	3	-10	bstr	i-th factor, Prime Factor
d_i	3	-11	bstr	i-th factor, Factor CRT Exponent
t_i	3	-12	bstr	i-th factor, Factor CRT Coefficient

Table 4: RSA Key Parameters

5. IANA Considerations

5.1. COSE Algorithms Registry

This section registers values in the IANA "COSE Algorithms" registry [IANA.COSE].

The values in Table 1 and Table 2 are to be added to the registry.

5.2. COSE Key Type Registry

This section registers values in the IANA "COSE Key Type" registry [IANA.COSE].

The values in Table 3 are to be added to the registry.

5.3. COSE Key Type Parameters Registry

This section registers values in the IANA "COSE Key Type Parameters" registry [IANA.COSE].

The values in Table 4 are to be added to the registry.

6. Security Considerations

6.1. Key Size Security Considerations

A key size of 2048 bits or larger MUST be used with these algorithms. This key size corresponds roughly to the same strength as provided by a 128-bit symmetric encryption algorithm. Implementations SHOULD be able to encrypt and decrypt with modulus between 2048 and 16K bits in length. Applications can impose additional restrictions on the length of the modulus.

In addition to needing to worry about keys that are too small to provide the required security, there are issues with keys that are too large. Denial of service attacks have been mounted with overly large keys or oddly sized keys. This has the potential to consume resources with these keys. It is highly recommended that checks on the key length be done before starting a cryptographic operation.

There are two reasonable ways to address this attack. First, a key should not be used for a cryptographic operation until it has been verified that it is controlled by a legitimate participant. This approach means that no cryptography would be done except with keys of legitimate parties. Second, applications can impose maximum as well as minimum length requirements on keys. This limits the resources that would otherwise be consumed by the use of overly large keys.

6.2. RSASSA-PSS Security Considerations

There is a theoretical hash substitution attack that can be mounted against RSASSA-PSS. However, the requirement that the same hash function be used consistently for all operations is an effective mitigation against it. Unlike ECDSA, hash function outputs are not truncated so that the full hash value is always signed. The internal padding structure of RSASSA-PSS means that one needs to have multiple collisions between the two hash functions to be successful in producing a forgery based on changing the hash function. This is highly unlikely.

6.3. RSAES-OAEP Security Considerations

A version of RSAES-OAEP using the default parameters specified in Appendix A.2.1 of RFC 3447 is included because this is the most widely implemented set of OAEP parameter choices. (Those default parameters are the SHA-1 hash function and the MGF1 with SHA-1 mask generation function.) While SHA-1 is deprecated as a general-purpose hash function, no known practical attacks are enabled by its use in this context.

7. References

7.1. Normative References

[I-D.ietf-cose-msg]
Schaad, J., "CBOR Object Signing and Encryption (COSE)", draft-ietf-cose-msg-24 (work in progress), November 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, DOI 10.17487/RFC3447, February 2003, <http://www.rfc-editor.org/info/rfc3447>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <http://www.rfc-editor.org/info/rfc7049>.

7.2. Informative References

[IANA.COSE]
IANA, "CBOR Object Signing and Encryption (COSE)", <http://www.iana.org/assignments/cose>.

[MultiPrimeRSA]
Hinek, M. and D. Cheriton, "On the Security of Multi-prime RSA", June 2006.

Appendix A. Acknowledgements

This specification incorporates text from draft-ietf-cose-msg-05 by Jim Schaad. Thanks are due to Kathleen Moriarty, Rich Salz, and Jim Schaad for their reviews of the specification.

Appendix B. Document History

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

-03

- Clarified the Security Considerations in ways suggested by Kathleen Moriarty.

- Acknowledged reviewers.

-02

- Reorganized the security considerations.
- Flattened the section structure.
- Applied wording improvements suggested by Jim Schaad.

-01

- Completed the sets of IANA registration requests.
- Revised the algorithm assignments based on those in draft-ietf-cose-msg-24.

-00

- This specification addresses COSE issue #21: Restore RSA-PSS and the "RSA" key type. The initial version of this specification incorporates text from draft-ietf-cose-msg-05 -- the last COSE message specification version before the RSA algorithms were removed.

Author's Address

Michael B. Jones
Microsoft

Email: mbj@microsoft.com
URI: http://self-issued.info/