Internet-Draft
Intended status: Standards Track
Expires: 27 September 2017

R. Housley
Vigil Security

27 March 2017

Use of the SHA3 One-way Hash Functions in the Cryptographic Message Syntax (CMS)

<draft-housley-lamps-cms-sha3-hash-00.txt>

Abstract

This document describes the conventions for using the four one-way hash functions in the SHA3 family with the Cryptographic Message Syntax (CMS).

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 27 September 2017.

Copyright Notice

1. Introduction

The Cryptographic Message Syntax (CMS) [CMS] is used to digitally sign, digest, authenticate, or encrypt arbitrary message contents. This specification describes the use of the four one-way hash functions in the SHA3 family (SHA3-224, SHA3-256, SHA3-384, and SHA3-512) [SHA3] with the CMS. In addition, this specification describes the use of these four one-way hash functions with the RSASSA PKCS#1 version 1.5 signature algorithm [PKCS1] and the Elliptic Curve Digital Signature Algorithm (ECDSA) [DSS] with the CMS signed-data content type.

1.1. ASN.1

CMS values are generated using ASN.1 [ASN1-B], using the Basic Encoding Rules (BER) and the Distinguished Encoding Rules (DER) [ASN1-E].

1.2. Terminology

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 [KEYWORDS].

2. Message Digest Algorithms

One-way hash functions are also referred to as message digest algorithms. This section specifies the conventions employed by CMS implementations that support SHA3-224, SHA3-256, SHA3-384, and SHA3-512 [SHA3].

Digest algorithm identifiers are located in the SignedData digestAlgorithms field, the SignerInfo digestAlgorithm field, the DigestedData digestAlgorithm field, and the AuthenticatedData digestAlgorithm field.

Digest values are located in the DigestedData digest field and the Message Digest authenticated attribute. In addition, digest values are input to signature algorithms.

SHA3-224, SHA3-256, SHA3-384, and SHA3-512 produce output values with 224, 256, 384, and 512 bits, respectively. The object identifiers for these four one-way hash functions are as follows:

```
hashAlgs OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
    us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) 2 }

id-sha3-224 OBJECT IDENTIFIER ::= { hashAlgs 7 }
```

```
id-sha3-256 OBJECT IDENTIFIER ::= { hashAlgs 8 }

id-sha3-384 OBJECT IDENTIFIER ::= { hashAlgs 9 }

id-sha3-512 OBJECT IDENTIFIER ::= { hashAlgs 10 }
```

When using the id-sha3-224, id-sha3-s256, id-sha3-384, or id-sha3-512 algorithm identifiers, the parameters field MUST be absent; not NULL but absent.

3. Signature Algorithms

This section specifies the conventions employed by CMS implementations that support the four SHA3 one-way hash functions with the RSASSA PKCS#1 version 1.5 signature algorithm [PKCS1] and the Elliptic Curve Digital Signature Algorithm (ECDSA) [DSS] with the CMS signed-data content type.

Signature algorithm identifiers are located in the SignerInfo signatureAlgorithm field of SignedData. Also, signature algorithm identifiers are located in the SignerInfo signatureAlgorithm field of countersignature attributes.

Signature values are located in the SignerInfo signature field of SignedData. Also, signature values are located in the SignerInfo signature field of countersignature attributes.

3.1. RSASSA PKCS#1 v1.5 with SHA3

The RSASSA PKCS#1 v1.5 is defined in [PKCS1]. When RSASSA PKCS#1 v1.5 is used in conjunction with one of the SHA3 one-way hash functions, the object identifiers are:

```
sigAlgs OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
    us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) 3 }

id-rsassa-pkcs1-v1_5-with-sha3-224 ::= { sigAlgs 13 }

id-rsassa-pkcs1-v1_5-with-sha3-256 ::= { sigAlgs 14 }

id-rsassa-pkcs1-v1_5-with-sha3-384 ::= { sigAlgs 15 }

id-rsassa-pkcs1-v1_5-with-sha3-512 ::= { sigAlgs 16 }
```

The algorithm identifier for RSASSA PKCS#1 v1.5 subject public keys in certificates is specified in [PKIXALG], and it is repeated here for convenience:

```
rsaEncryption OBJECT IDENTIFIER ::= { iso(1) member-body(2)
    us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 1 }
```

When the rsaEncryption, id-rsassa-pkcs1-v1_5-with-sha3-224, id-rsassa-pkcs1-v1_5-with-sha3-256, id-rsassa-pkcs1-v1_5-with-sha3-384, and id-rsassa-pkcs1-v1_5-with-sha3-512 algorithm identifier is used, AlgorithmIdentifier parameters field MUST contain NULL.

When the rsaEncryption algorithm identifier is used, the RSA public key, which is composed of a modulus and a public exponent, MUST be encoded using the RSAPublicKey type as specified in [PKIXALG]. The output of this encoding is carried in the certificate subject public key. The definition of RSAPublicKey is repeated here for convenience:

```
RSAPublicKey ::= SEQUENCE {
   modulus INTEGER, -- n
   publicExponent INTEGER } -- e
```

When signing, the RSASSA PKCS#1 v1.5 signature algorithm generates a single value, and that value is used directly as the signature value.

3.2. ECDSA with SHA3

The Elliptic Curve Digital Signature Algorithm (ECDSA) is defined in [DSS]. When ECDSA is used in conjunction with one of the SHA3 one-way hash functions, the object identifiers are:

```
sigAlgs OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
    us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) 3 }

id-ecdsa-with-sha3-224 ::= { sigAlgs 9 }

id-ecdsa-with-sha3-256 ::= { sigAlgs 10 }

id-ecdsa-with-sha3-384 ::= { sigAlgs 11 }

id-ecdsa-with-sha3-512 ::= { sigAlgs 12 }
```

When using the id-ecdsa-with-sha3-224, id-ecdsa-with-sha3-256, id-ecdsa-with-sha3-384, and id-ecdsa-with-sha3-512 algorithm identifiers, the parameters field MUST be absent; not NULL but absent.

The conventions for ECDSA public keys is as specified in [PKIXECC]. The ECParameters associated with the ECDSA public key in the signers certificate SHALL apply to the verification of the signature.

When signing, the ECDSA algorithm generates two values. These values are commonly referred to as r and s. To easily transfer these two values as one signature, they MUST be ASN.1 encoded using the ECDSA-Sig-Value defined in [PKIXALG] and repeated here for convenience:

```
ECDSA-Sig-Value ::= SEQUENCE {
    r  INTEGER,
    s  INTEGER }
```

4. Message Authentication Codes

This section specifies the conventions employed by CMS implementations that support the HMAC with SHA3 message authentication code (MAC).

MAC algorithm identifiers are located in the AuthenticatedData macAlgorithm field.

MAC values are located in the AuthenticatedData mac field.

When HMAC is used in conjunction with one of the SHA3 one-way hash functions, the object identifiers are:

```
hashAlgs OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16)
    us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) 2 }

id-hmacWithSHA3-224 OBJECT IDENTIFIER ::= { hashAlgs 13 }

id-hmacWithSHA3-256 OBJECT IDENTIFIER ::= { hashAlgs 14 }

id-hmacWithSHA3-384 OBJECT IDENTIFIER ::= { hashAlgs 15 }

id-hmacWithSHA3-512 OBJECT IDENTIFIER ::= { hashAlgs 16 }
```

When the id-hmacWithSHA3-224, id-hmacWithSHA3-256, id-hmacWithSHA3-384, and id-hmacWithSHA3-512 algorithm identifier is used, the parameters field MUST be absent; not NULL but absent.

5. Security Considerations

Implementations must protect the signer's private key. Compromise of the signer's private key permits masquerade.

When more than two parties share the same message-authentication key, data origin authentication is not provided. Any party that knows the message-authentication key can compute a valid MAC, therefore the content could originate from any one of the parties.

Implementations must randomly generate message-authentication keys and one-time values, such as the k value when generating a ECDSA signature. In addition, the generation of public/private key pairs relies on a random numbers. The use of inadequate pseudo-random number generators (PRNGs) to generate cryptographic such values can result in little or no security. An attacker may find it much easier to reproduce the PRNG environment that produced the keys, searching the resulting small set of possibilities, rather than brute force searching the whole key space. The generation of quality random numbers is difficult. RFC 4086 [RANDOM] offers important guidance in this area, and Appendix 3 of FIPS Pub 186-4 [DSS] provides some PRNG techniques.

Implementers should be aware that cryptographic algorithms become weaker with time. As new cryptanalysis techniques are developed and computing performance improves, the work factor to break a particular cryptographic algorithm will reduce. Therefore, cryptographic algorithm implementations should be modular allowing new algorithms to be readily inserted. That is, implementers should be prepared to regularly update the set of algorithms in their implementations.

6. Normative References

[ASN1-B] ITU-T, "Information technology -- Abstract Syntax Notation One (ASN.1): Specification of basic notation", ITU-T Recommendation X.680, 2015.

[ASN1-E] ITU-T, "Information technology -- ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)", ITU-T Recommendation X.690, 2015.

[CMS] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, September 2009.

[DSS] National Institute of Standards and Technology, U.S. Department of Commerce, "Digital Signature Standard, version 4", NIST FIPS PUB 186-4, 2013.

[HMAC] Krawczyk, H., "HMAC: Keyed-Hashing for Message Authentication", RFC 2104. February 1997.

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[PKCS1] Moriarty, K., Kaliski, B., Jonsson, J., and A. Rusch, "PKCS #1: RSA Cryptography Specifications Version 2.2" RFC 8017, November 2016.

[PKIXALG] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002.

[PKIXECC] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, March 2009.

[SHA3] National Institute of Standards and Technology, U.S. Department of Commerce, "SHA-3 Standard - Permutation-Based Hash and Extendable-Output Functions", FIPS PUB 202, August 2015.

7. Informative References

[RANDOM] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.

Appendix. ASN.1 Module

TBD

Author Address

Russ Housley
Vigil Security, LLC
918 Spring Knoll Drive
Herndon, VA 20170
USA

Email: housley@vigilsec.com