

LAMPS WG
Internet-Draft
Intended status: Standards Track
Expires: December 31, 2018

P. Kampanakis
Cisco Systems
Q. Dang
NIST
June 29, 2018

Internet X.509 Public Key Infrastructure: Additional Algorithm
Identifiers for RSASSA-PSS and ECDSA using SHAKEs as Hash Functions
draft-ietf-lamps-pkix-shake-02

Abstract

Digital signatures are used to sign messages, X.509 certificates and CRLs (Certificate Revocation Lists). This document describes the conventions for using the SHAKE family of hash functions in the Internet X.509 as one-way hash functions with the RSA Probabilistic Signature Scheme and ECDSA signature algorithms. The conventions for the associated subject public keys are also described.

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 <https://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 December 31, 2018.

Copyright Notice

Copyright (c) 2018 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 (<https://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.	Change Log	2
2.	Introduction	3
3.	Identifiers	3
4.	Use in PKIX	4
4.1.	Signatures	4
4.1.1.	RSASSA-PSS Signatures	5
4.1.2.	ECDSA Signatures	5
4.2.	Public Keys	6
4.2.1.	RSASSA-PSS Public Keys	6
4.2.2.	ECDSA Public Keys	7
5.	IANA Considerations	7
6.	Security Considerations	7
7.	Acknowledgements	8
8.	References	8
8.1.	Normative References	8
8.2.	Informative References	9
Appendix A.	ASN.1 module	10
	Authors' Addresses	10

[1.](#) Change Log

[EDNOTE: Remove this section before publication.]

o [draft-ietf-lamps-pkix-shake-02](#):

- * Significant reorganization of the sections to simplify the introduction, the new OIDs and their use in PKIX.
- * Added new OIDs for RSASSA-PSS that hardcode hash, salt and MFG, according the WG consensus.
- * Updated Public Key section to use the new RSASSA-PSS OIDs and clarify the algorithm identifier usage.
- * Removed the no longer used SHAKE OIDs from [section 3.1](#).
- * Consolidated subsection for message digest algorithms.

- * Text fixes.
- o [draft-ietf-lamps-pkix-shake-01](#):

- * Changed titles and section names.
- * Removed DSA after WG discussions.
- * Updated shake OID names and parameters, added MGF1 section.
- * Updated RSASSA-PSS section.
- * Added Public key algorithm OIDs.
- * Populated Introduction and IANA sections.
- o [draft-ietf-lamps-pkix-shake-00](#):
 - * Initial version

[2](#). Introduction

This document describes several cryptographic algorithm identifiers for several cryptographic algorithms which use variable length output SHAKE functions introduced in [[SHA3](#)] which can be used with the Internet X.509 Certificate and CRL profile [[RFC5280](#)].

The SHA-3 family of one-way hash functions is specified in [[SHA3](#)]. In the SHA-3 family, two extendable-output functions, called SHAKE128 and SHAKE256 are defined. Four hash functions, SHA3-224, SHA3-256, SHA3-384, and SHA3-512 are also defined but are out of scope for this document. A SHAKE is a variable length hash function. The output lengths, in bits, of the SHAKE hash functions are defined by the d parameter. The corresponding collision and preimage resistance security levels for SHAKE128 and SHAKE256 are respectively $\min(d/2, 128)$ and $\min(d, 128)$ and $\min(d/2, 256)$ and $\min(d, 256)$ bits.

SHAKEs can be used as the message digest function (to hash the message to be signed) and as the hash function in the mask generating functions in RSASSA-PSS and ECDSA. In this document, we define four

new OIDs for RSASSA-PSS and ECDSA when SHAKE128 and SHAKE256 are used as hash functions. The same algorithm identifiers are used for identifying a public key, and identifying a signature.

[3.](#) Identifiers

The new identifiers for RSASSA-PSS signatures using SHAKEs are below.

```
id-RSASSA-PSS-SHAKE128 OBJECT IDENTIFIER ::= { TBD }
```

Kampanakis & Dang

Expires December 31, 2018

[Page 3]

Internet-Draft

SHAKE identifiers in X.509

June 2018

```
id-RSASSA-PSS-SHAKE256 OBJECT IDENTIFIER ::= { TBD }
```

```
[ EDNOTE: "TBD" will be specified by NIST later. ]
```

The new algorithm identifiers of ECDSA signatures using SHAKEs are below.

```
id-ecdsa-with-shake128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2)
country(16) us(840) organization(1) gov(101) csor(3) al
id-ecdsa-with-shake(3) TBD }
```

```
id-ecdsa-with-shake256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2)
country(16) us(840) organization(1) gov(101) csor(3) al
id-ecdsa-with-shake(3) TBD }
```

```
[ EDNOTE: "TBD" will be specified by NIST later. ]
```

The parameters for these four identifiers above MUST be absent. That is, the identifier SHALL be a SEQUENCE of one component, the OID.

[4.](#) Use in PKIX

[4.1.](#) Signatures

Signatures can be placed in a number of different ASN.1 structures.

The top level structure for an X.509 certificate, to illustrate how signatures are frequently encoded with an algorithm identifier and a location for the signature, is

```
Certificate ::= SEQUENCE {  
    tbsCertificate      TBSCertificate,  
    signatureAlgorithm  AlgorithmIdentifier,  
    signatureValue      BIT STRING }
```

The identifiers defined in [Section 3](#) can be used as the AlgorithmIdentifier in the signatureAlgorithm field in the sequence Certificate and the signature field in the sequence tbsCertificate in X.509 [[RFC3280](#)].

Conforming CA implementations MUST specify the algorithms explicitly by using the OIDs specified in [Section 3](#) when encoding RSASSA-PSS and ECDSA with SHAKE signatures, and public keys in certificates and CRLs. Encoding rules for RSASSA-PSS and ECDSA signature values are specified in [[RFC4055](#)] and [[RFC5480](#)] respectively.

Conforming client implementations that process RSASSA-PSS and ECDSA with SHAKE signatures when processing certificates and CRLs MUST recognize the corresponding OIDs.

[4.1.1](#). RSASSA-PSS Signatures

The RSASSA-PSS algorithm is defined in [[RFC8017](#)]. When id-RSASSA-PSS-SHAKE128 or id-RSASSA-PSS-SHAKE256 specified in [Section 3](#) is used, the encoding MUST omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, id-RSASSA-PSS-SHAKE128 or id-RSASSA-PSS-SHAKE256.

The hash algorithm to hash a message being signed and the hash algorithm in the maskGenAlgorithm used in RSASSA-PSS MUST be the same, SHAKE128 or SHAKE256 respectively. The output-length of the hash algorithm which hashes the message SHALL be 32 or 64 bytes respectively.

The maskGenAlgorithm is the MGF1 specified in Section B.2.1 of [[RFC8017](#)]. The output length for SHAKE128 or SHAKE256 being used as the hash function in MGF1 is $(n - 264)/8$ or $(n - 520)/8$ bytes respectively, where n is the RSA modulus in bits. For example, when

RSA modulus n is 2048, the output length of SHAKE128 or SHAKE256 in the MGF1 will be 223 or 191 when id-RSASSA-PSS-SHAKE128 or id-RSASSA-PSS-SHAKE256 is used respectively.

The RSASSA-PSS saltLength MUST be 32 or 64 bytes respectively. Finally, the trailerField MUST be 1, which represents the trailer field with hexadecimal value 0xBC [RFC8017].

[4.1.2](#). ECDSA Signatures

The Elliptic Curve Digital Signature Algorithm (ECDSA) is defined in [X9.62]. When the id-ecdsa-with-SHAKE128 or id-ecdsa-with-SHAKE256 (specified in [Section 3](#)) algorithm identifier appears, the respective SHAKE function (SHAKE128 or SHAKE256) is used as the hash. The encoding MUST omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-ecdsa-with-SHAKE128 or id-ecdsa-with-SHAKE256.

For simplicity and compliance with the ECDSA standard specification, the output size of the hash function must be explicitly determined. The output size, d , for SHAKE128 or SHAKE256 used in ECDSA MUST be 256 or 512 bits respectively.

Conforming CA implementations that generate ECDSA with SHAKE signatures in certificates or CRLs MUST generate such signatures in accordance with all the requirements specified in [Sections 7.2](#) and

7.3 of [X9.62] or with all the requirements specified in [Section 4.1.3](#) of [SEC1]. They MAY also generate such signatures in accordance with all the recommendations in [X9.62] or [SEC1] if they have a stated policy that requires conformance to these standards. These standards may have not specified SHAKE128 and SHAKE256 as hash algorithm options. However, SHAKE128 and SHAKE256 with output length being 32 and 64 octets respectively are substitutions for 256 and 512-bit output hash algorithms such as SHA256 and SHA512 used in the standards.

[4.2](#). Public Keys

Certificates conforming to [RFC5280] can convey a public key for any public key algorithm. The certificate indicates the algorithm through an algorithm identifier. This algorithm identifier is an OID

and optionally associated parameters.

In the X.509 certificate, the `subjectPublicKeyInfo` field has the `SubjectPublicKeyInfo` type, which has the following ASN.1 syntax:

```
SubjectPublicKeyInfo ::= SEQUENCE {
    algorithm      AlgorithmIdentifier,
    subjectPublicKey BIT STRING
}
```

The fields in `SubjectPublicKeyInfo` have the following meanings:

- o `algorithm` is the algorithm identifier and parameters for the public key.
- o `subjectPublicKey` contains the byte stream of the public key. The algorithms defined in this document always encode the public key as an exact multiple of 8-bits.

The conventions for RSASSA-PSS and ECDSA public keys algorithm identifiers are as specified in [\[RFC3279\]](#), [\[RFC4055\]](#) and [\[RFC5480\]](#) , but we include them below for convenience.

[4.2.1](#). RSASSA-PSS Public Keys

[\[RFC3279\]](#) defines the following OID for RSA `AlgorithmIdentifier` in the `SubjectPublicKeyInfo` with NULL parameters.

```
rsaEncryption OBJECT IDENTIFIER ::= { pkcs-1 1 }
```

Additionally, when the RSA private key owner wishes to limit the use of the public key exclusively to RSASSA-PSS, the `AlgorithmIdentifiers` for RSASSA-PSS defined in [Section 3](#) can be used as the algorithm

field in the `SubjectPublicKeyInfo` sequence [\[RFC3280\]](#). The identifier parameters, as explained in section [Section 3](#), MUST be absent.

Regardless of what public key 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 [\[RFC4055\]](#). The output of this encoding is carried in the certificate `subjectPublicKey`.

```

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

```

[4.2.2.](#) ECDSA Public Keys

For ECDSA, when `id-ecdsa-with-shake128` or `id-ecdsa-with-shake256` is used as the `AlgorithmIdentifier` in the `algorithm` field of `SubjectPublicKeyInfo`, the parameters, as explained in [section 3](#), MUST be absent.

Additionally, the mandatory EC `SubjectPublicKey` is defined in [Section 2.1.1](#) and its syntax is in [Section 2.2 of \[RFC5480\]](#). We also include them here for convenience:

```

id-ecPublicKey OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) ansi-X9-62(10045) keyType(2) 1 }

```

The `id-ecPublicKey` parameters MUST be present and are defined as

```

ECParameters ::= CHOICE {
    namedCurve      OBJECT IDENTIFIER
    -- implicitCurve NULL
    -- specifiedCurve SpecifiedECDomain
}

```

The `ECParameters` associated with the ECDSA public key in the signer's certificate SHALL apply to the verification of the signature.

[5.](#) IANA Considerations

This document uses several new registries [EDNOTE: Update here.]

[6.](#) Security Considerations

The SHAKEs are deterministic functions. Like any other deterministic functions, executing each function with the same input multiple times will produce the same output. Therefore, users should not expect

unrelated outputs (with the same or different output lengths) from

excuting a SHAKE function with the same input multiple times.

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

Implementations must randomly generate one-time values, such as the k value when generating a ECDSA signature. In addition, the generation of public/private key pairs relies on random numbers. The use of inadequate pseudo-random number generators (PRNGs) to generate such cryptographic values can result in little or no security. The generation of quality random numbers is difficult. [RFC4086] offers important guidance in this area, and [SP800-90A] series provide acceptable PRNGs.

Implementers should be aware that cryptographic algorithms may become weaker with time. As new cryptanalysis techniques are developed and computing power increases, the work factor or time required to break a particular cryptographic algorithm may decrease. 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.

7. Acknowledgements

We would like to thank Sean Turner for his valuable contributions to this document.

8. References

8.1. Normative References

- [RFC3280] Housley, R., Polk, W., Ford, W., and D. Solo, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", [RFC 3280](#), DOI 10.17487/RFC3280, April 2002, <<https://www.rfc-editor.org/info/rfc3280>>.
- [RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", [RFC 4055](#), DOI 10.17487/RFC4055, June 2005, <<https://www.rfc-editor.org/info/rfc4055>>.

- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", [RFC 5280](#), DOI 10.17487/RFC5280, May 2008, <<https://www.rfc-editor.org/info/rfc5280>>.
- [RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", [RFC 5480](#), DOI 10.17487/RFC5480, March 2009, <<https://www.rfc-editor.org/info/rfc5480>>.
- [RFC8017] Moriarty, K., Ed., Kaliski, B., Jonsson, J., and A. Rusch, "PKCS #1: RSA Cryptography Specifications Version 2.2", [RFC 8017](#), DOI 10.17487/RFC8017, November 2016, <<https://www.rfc-editor.org/info/rfc8017>>.
- [SHA3] National Institute of Standards and Technology, "SHA-3 Standard – Permutation-Based Hash and Extendable-Output Functions FIPS PUB 202", August 2015, <<https://www.nist.gov/publications/sha-3-standard-permutation-based-hash-and-extendable-output-functions>>.

8.2. Informative References

- [RFC3279] 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](#), DOI 10.17487/RFC3279, April 2002, <<https://www.rfc-editor.org/info/rfc3279>>.
- [RFC4086] Eastlake 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", [BCP 106](#), [RFC 4086](#), DOI 10.17487/RFC4086, June 2005, <<https://www.rfc-editor.org/info/rfc4086>>.
- [SEC1] Standards for Efficient Cryptography Group, "SEC 1: Elliptic Curve Cryptography", May 2009, <<http://www.secg.org/sec1-v2.pdf>>.
- [SP800-90A] National Institute of Standards and Technology, "Recommendation for Random Number Generation Using Deterministic Random Bit Generators. NIST SP 800-90A", June 2015, <<http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf>>.

[X9.62] American National Standard for Financial Services (ANSI), "X9.62-2005 Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Standard (ECDSA)", November 2005.

[Appendix A](#). ASN.1 module

[EDNOTE: More here.]

Authors' Addresses

Panos Kampanakis
Cisco Systems

Email: pkampana@cisco.com

Quynh Dang
NIST
100 Bureau Drive, Stop 8930
Gaithersburg, MD 20899-8930
USA

Email: quynh.dang@nist.gov

