Network Workign Group	A. Jivsov
Internet-Draft	F5 Networks, Inc.
Intended status: Informational	August 19, 2015
Expires: February 20, 2016	

The use of Secure Hash Algorithm 3 in OpenPGP

draft-jivsov-openpgp-sha3-00

Abstract

This document presents the necessary information to implement the SHA-3 hash algorithm with the OpenPGP format.

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 February 20, 2016.

Copyright Notice

Table of Contents

1. Introduction

The OpenPGP format [RFC4880] supports multiple hash algorithms. This document provides the necessary information to use the Secure Hash Algorithm-3 (SHA-3) hash algorithm with the OpenPGP format.

National Institute of Standards and Technology (NIST) selected [Keccak] as the SHA-3 algorithm [SHA-3] for its elegant design, its efficiency on various computing devices, high performance in hardware implementations, and different internal structure that may provide better defense against attacks on its predecessors from the SHA-1 and SHA-2 family.

2. Conventions used in this document

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 [RFC2119].

3. Overview of the hash algorithm use in OpenPGP

Hash algorithm is used in [RFC4880] with digital signatures, with modification detection code, and to calculate key fingerprints. Only digital signatures allow algorithm agility and are most vulnerable to various attacks on hash functions. The focus of this document is on the use of SHA-3 hash with OpenPGP digital signatures.

The use of hash algorithm with digital signatures in OpenPGP falls into two categories. The first one is the digital signatures over messages, and another one is the certifications of the key material. The latter area includes self-signatures, which convey key preference information, among other tasks. The rest of key certifications are third party key signatures. These categories will be considered separately in more details in Section 6 for the impact on interoperability.

4. Supported SHA-3 algorithms

SHA-3 specification [SHA-3] defines a single cryptographic hash function that is parameterized to produce hash output of certain sizes. This document refers to these instantiations of the same hash algorithm as SHA-3 hash algorithms and treats them as different hash algorithms with unique IDs

The SHA-3 algorithms SHA3-256, SHA3-384, and SHA3-512, which have the output size of 256, 384, and 512 bits respectively, MAY be used as a hash algorithm with digital signatures.

The input size to the hash algorithm in OpenPGP MUST be a multiple of 8 bits, in other words, the input is always represented as a sequence of 8-bit octets. In more details, [SHA-3] provides domain separation for

different instantiations of [Keccak] by prefixing a message-to-be-hashed M with two bits 01. M is what gets hashed with other hash algorithms, such as SHA2-256, and MUST be a multiple of 8 bits. The hashing of 01 bits can be integrated into the finalization stage of a SHA-3 implementation. [SHA3IUF] may be useful in verifying the correctness of the SHA-3 implementation.

An application MAY generate signatures with SHA3-256, SHA3-384, and SHA3-512 hash algorithms. An application MUST implement signature verification for 3 SHA-3 algorithms if it implements at least one of the SHA-3 algorithms. The all-or-nothing requirement should be feasible for the vast majority of implementations because it is relatively easy to implement all SHA-3 algorithms with a single unified implementation, such that the only variable that distinguishes these algorithms is an integer representing the hash output size (the capacity of the Keccak). In particular, there are no special tables or magic constants that are specific to the hash output size of the SHA-3 algorithms.

Applications MAY use any digital signature algorithm described in [RFC4880] and Elliptic Curve DSA algorithm described in [RFC6637] with SHA-3.

[SHA3IUF]

5. Use with RSA digital signatures

Section 5.2.2 of [RFC4880] describes the Version 3 Signature Packet Format. One of allowed public key algorithms in that section is the RSA digital signature algorithm. RSA signatures use the PKCS#1 encoding to format (cryptographically "pad") the output of the hash algorithm. The padding includes the DER-encoded prefix that is fixed for the given hash algorithm. While this prefix is a DER encoding of an ASN.1 Object Identifier (OID), the length of the hash output, and the supporting fields, it's possible to define the prefix as a fixed sequence of octets. These prefixes are defined below.

Algorithm	ASN.1 OID
SHA3-256	2.16.840.1.101.3.4.2.8
SHA3-384	2.16.840.1.101.3.4.2.9
SHA3-512	2.16.840.1.101.3.4.2.10

SHA-3 hash OIDs

The full DER-encoded hash prefixes are provided bellow.

Algorithm	Full DER prefix
SHA3-256	30,31,30,0d, 06,09,60,86, 48,01,65,03, 04,02,08,05, 00,04,20 [32-byte hash]
SHA3-384	30,41,30,0d, 06,09,60,86, 48,01,65,03, 04,02,09,05, 00,04,30 [48-byte hash]
SHA3-512	30,51,30,0d, 06,09,60,86, 48,01,65,03, 04,02,0a,05, 00,04,40 [64-byte hash]

SHA-3 hash full DER prefixes

6. Interoperability concerns arising from an introduction of a new hash algorithm

The use of a new hash algorithm in a public key certifications, especially in self-signatures, can make the key unusable in the large extent of the OpenPGP ecosystem. The impact of the use of the new hash algorithm in a digital signature over a message is limited to users who will be verifying this message.

Implementation MUST include SHA-3 algorithms in the Hash Algorithm Preferences of the keys it generates when it updates the algorithm preferences. This preference is described in the Section 13.3.2 of [RFC4880].

In cases when the message is both encrypted and signed, the application knows the keys of the entities who

will be performing signature verification. The application SHOULD rely on Hash Algorithm Preferences of the recipients' public keys to learn about SHA-3 support.

Addition of the SHA-3 support will benefit from planning ahead.

The digital signature validation is dependent on wide support of the selected hash algorithm by deployed OpenPGP implementations that will be verifying the digital signature. In general, there is no method in OpenPGP by which a party that issues a digital signature can be certain about the support of a hash algorithm by other implementations.

The safest method to mitigate these challenges is a phased deployment of the new hash algorithm support in the application, as follows:

- Implement the hash algorithm, test it thoroughly.
- Enable its use with the digital signature algorithms and test signature generation and verification, ideally, with multiple implementations. Then disable the signature generation in the production code (i.e. leave the "read support" only).
- Wait sufficiently long for the deployment of the applications that can verify digital signatures with the new hash algorithm.
- Enable the generation of signatures with the new hash algorithm, starting from the signatures over messages, later extending it to certifications.

7. IANA Considerations

This document asks to allocate the consecutive hash algorithm IDs from the Hash Algorithm ID range, defined in the Section 9.4 of [RFC4880].

The starting ID is not important, but the properties that the IDs are sequential and in the given order are expected to simplify implementation.

ID	Algorithm	Text Name
13	SHA-3 with 256 bit output	"SHA3-256"
14	SHA-3 with 384 bit output	"SHA3-384"
15	SHA-3 with 512 bit output	"SHA3-512"

SHA-3 hash IDs

8. Security Considerations

SHA-3 hash algorithms have different internal structure, informally called "wide pipe" or "sponge" design, that may offer higher security in the future over the strength of its SHA-2 predecessors with identical output size, especially with digital signature applications.

Table 2 from [SP800-57] SHOULD be used to determine the corresponding strength of the public key algorithm for the given hash algorithm. For example, the SHA-3 256 has equivalent security strength to the NIST curve P-256 [RFC6637]. Refer to Table 4 in section A.1 [SHA-3] for security strength of the SHA-3 algorithms.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D. and R. Thayer, "OpenPGP Message Format", RFC 4880, DOI 10.17487/RFC4880, November 2007.

[SHA-3] NIST, "SHA-3 STANDARD: PERMUTATION-BASED HASH AND EXTENDABLE OUTPUT FUNCTIONS", August 2015.

9.2. Informative References

[Keccak] Bertoni, G., Daemen, J., Peeters, M. and G. Van Assche, "The Keccak sponge function family", 2012.

[RFC6637] Jivsov, A., "Elliptic Curve Cryptography (ECC) in OpenPGP", RFC 6637, DOI 10.17487/RFC6637, June 2012.

[SHA3IUF] Jivsov, A., "A single-file C implementation of SHA-3 with IUF API", August 2015.

[SP800-57] NIST, "Recommendation for Key Management -- Part 1: General (Revision 3)", July 2012.

Author's Address

Andrey Jivsov
F5 Networks, Inc.
EMail: openpgp@brainhub.org