Workgroup: HIP
Internet-Draft:
draft-moskowitz-orchid-cshake-01
Updates: 7343 (if approved)
Published: 21 May 2020
Intended Status: Standards Track
Expires: 22 November 2020
Authors: R. Moskowitz HTT Consulting
S. Card AX Enterprize
A. Wiethuechter AX Enterprize

Using cSHAKE in ORCHIDs

Abstract

This document specifies how to use the cSHAKE hash for ORCHID generation and allows for varying sized hashes in the ORCHID along with additional information within the ORCHID. It is an addendum to ORCHIDv2 [RFC7343].

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 22 November 2020.

Table of Contents

1. Introduction

This document adds the [Keccak] based cSHAKE XOF hash function from NIST SP 800-185 [NIST.SP.800-185] to ORCHIDv2 [RFC7343]. cSHAKE is a variable output length hash function. As such it does not use the truncation operation that other hashes need. The invocation of cSHAKE specifies the desired number of bits in the hash output.

cSHAKE is used, rather than SHAKE from NIST FIPS 202 [NIST.FIPS.202], as cSHAKE has a parameter 'S' as a customization bit string.

This parameter will be used for including the ORCHID Context Identifier in a standard fashion.

An additional change to ORCHID construction will allow for shorter hash output lengths to permit inclusion of additional information like Hierarchical HITs [hierarchical-hit] into the ORCHID.

2. Terms, Notation and Definitions

2.1. Requirements Terminology

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2.2. Notation

| Signifies concatenation of information - e.g., X | Y is the concatenation of X and Y.

2.3. Definitions

Keccak (KECCAK Message Authentication Code):
The family of all sponge functions with a KECCAK-f permutation as the underlying function and multi-rate padding as the padding rule.

cSHAKE (The customizable SHAKE function):
Extends the SHAKE scheme to allow users to customize their use of the function.

SHAKE (Secure Hash Algorithm KECCAK):
A secure hash that allows for an arbitrary output length.

XOF (eXtendable-Output Function):
A function on bit strings (also called messages) in which the output can be extended to any desired length.

3. Adding additional information to the ORCHID

ORCHIDv2 [RFC7343] is currently defined as consisting of three components:

```
ORCHID     :=  Prefix | OGA ID | Encode_96( Hash )

where:

Prefix          : A constant 28-bit-long bitstring value
                  (IANA IPv6 assigned).

OGA ID          : A 4-bit long identifier for the Hash_function
                  in use within the specific usage context.

Encode_96( )    : An extraction function in which output is obtained
                  by extracting the middle 96-bit-long bitstring
                  from the argument bitstring.
```

This addendum will be constructed as follows:

```
ORCHID     :=  Prefix | OGA ID | Info (n) | Hash (m)

where:

Prefix          : A constant 28-bit-long bitstring value
                  (IANA IPv6 assigned).

OGA ID          : A 4-bit long identifier for the Hash_function
                  in use within the specific usage context.

Info (n)        : n bits of information that define a use of the
                  ORCHID.  n can be zero, that is no additional
                  information.

Hash (m)        : An extraction function in which output is m bits.

n + m = 96 bits
```

The 96 bits currently allocated to the Encode_96 function can be divided in any manner between the additional information and the hash output. Care must be taken in determining the size of the hash portion, taking into account risks like pre-image attacks. Thus 64 bits as used in Hierarchical HITs may be as small as is acceptable.

4. ORCHID Decoding

With this addendum, the decoding of an ORCHID is determined by the Prefix and OGA ID. ORCHIDv2 [RFC7343] decoding is selected when the Prefix is: 2001:20::/28.

For Heirarchical HITs, the decoding is determined by the presence of the HHIT Prefix as specified in the HHIT document.

5. ORCHID Encoding

ORCHIDv2 has a number of inputs including a Context ID, some header bits, the hash algorithm, and the input bitstream, normally just the public key. The output is a 96 bit value.

This addendum adds a different encoding process to that currently used. The input to the hash function explicitly includes all the fixed header content plus the Context ID. The fixed header content consists of the Prefix, OGA ID, and the Additional Information. Secondly, the length of the resulting hash is set by the rules set by the Prefix/OGA ID. In the case of Hierarchical HITs, this is 64 bits.

To achieve the variable length output in a consistent manner, the cSHAKE hash is used. For this purpose, cSHAKE128 is appropriate. The the cSHAKE function call for this addendum is:

```
cSHAKE128(Input, L, "", Context ID)

Input      :=  Prefix | OGA ID | Additional Information | HOST_ID
L          :=  Length in bits of hash portion of ORCHID
```

Hierarchical HIT uses the same context as all other HIPv2 HIT Suites as they are clearly separated by the distinct HIT Suite ID.

6. Initial use case for cSHAKE

The EdDSA/cSHAKE based HITs in New Cryptographic Algorithms for HIP [new-hip-crypto] is the first HIP Suite to use cSHAKE, thus using this addendum.

7. Initial use case for Additional Information

Hierarchical HITs [hierarchical-hit] (HHITs) is the first HIT construct that specifies the need of dividing the 96 bits available to ORCHID into its Hierarchy ID (HID) and HI Hash, thus using this addendum.

HHITs use a unique Context ID as well as a Prefix different from HIPv2 [RFC7401]. The different Prefix enables receivers to properly decode the HID out of the HIT and validate the HIT, given the HI.

8. Collision risks with Hierarchical HITs

The 64 bit hash size does have an increased risk of collisions over the 96 bit hash size used for the other HIT Suites. There is a 0.01% probability of a collision in a population of 66 million. The probability goes up to 1% for a population of 663 million. See Appendix A for the collision probability formula.

However, this risk of collision is within a single "Additional Information" value. Some registration process should be used to reject a collision, forcing the client to generate a new HI and thus HIT and reapplying to the registration process.

9. IANA Considerations

TBD.

10. Security Considerations

A 64 bit hash space presents a real risk of second pre-image attacks. A Registry service effectively block attempts to "take over" such a HIT. It does not stop a rogue attempting to impersonate a known HIT. This attack can be mitigated by the Responder using DNS to find the HI for the HIT or the RVS for the HIT that then provides the registered HI.

11. References

11.1. Normative References

[NIST.FIPS.202] Dworkin, M., "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions", National Institute of Standards and Technology report, DOI 10.6028/nist.fips.202, July 2015, <https://doi.org/10.6028/nist.fips.202>.

[NIST.SP.800-185] Kelsey, J., Change, S., and R. Perlner, "SHA-3 derived functions: cSHAKE, KMAC, TupleHash and ParallelHash", National Institute of Standards and Technology report, DOI 10.6028/nist.sp.800-185, December 2016, <https://doi.org/10.6028/nist.sp.800-185>.

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

[RFC7343] Laganier, J. and F. Dupont, "An IPv6 Prefix for Overlay Routable Cryptographic Hash Identifiers Version 2 (ORCHIDv2)", RFC 7343, DOI 10.17487/RFC7343, September 2014, <https://www.rfc-editor.org/info/rfc7343>.

[RFC8174]
Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

11.2. Informative References

[hierarchical-hit]
Moskowitz, R., Card, S., and A. Wiethuechter, "Hierarchical HITs for HIPv2", Work in Progress, Internet-Draft, draft-moskowitz-hip-hierarchical-hit-05, 13 May 2020, <https://tools.ietf.org/html/draft-moskowitz-hip-hierarchical-hit-05>.

[Keccak] Bertoni, G., Daemen, J., Peeters, M., Van Assche, G., and R. Van Keer, "The Keccak Function", , <https://keccak.team/index.html>.

[new-hip-crypto] Moskowitz, R., Card, S., and A. Wiethuechter, "New Cryptographic Algorithms for HIP", Work in Progress, Internet-Draft, draft-moskowitz-hip-new-crypto-04, 23 January 2020, <https://tools.ietf.org/html/draft-moskowitz-hip-new-crypto-04>.

[RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <https://www.rfc-editor.org/info/rfc7401>.

Appendix A. Calculating Collision Probabilities

The accepted formula for calculating the probability of a collision is:

```
p = 1 - e^{-k^2/(2n)}
```

```
P   Collision Probability
n   Total possible population
k   Actual population
```

Acknowledgments

Quynh Dang of NIST gave considerable guidance on using Keccak and the NIST supporting documents. Joan Deamen of the Keccak team was especially helpful in many aspects of using Keccak.

Authors' Addresses

Robert Moskowitz
HTT Consulting
Oak Park, MI 48237
United States of America

Email: rgm@labs.htt-consult.com

Stuart W. Card
AX Enterprize
4947 Commercial Drive
Yorkville, NY 13495
United States of America

Email: stu.card@axenterprize.com

Adam Wiethuechter
AX Enterprize
4947 Commercial Drive
Yorkville, NY 13495
United States of America

Email: adam.wiethuechter@axenterprize.com