XEP-0300: Use of Cryptographic Hash Functions in XMPP

Peter Saint-Andre
mailto:xsf@stpeter.im
xmpp:peter@jabber.org
http://stpeter.im/

Matthew Wild
mailto:mwild1@gmail.com
xmpp:me@matthewwild.co.uk

Kevin Smith
mailto:kevin@kismith.co.uk
xmpp:kevin@doomsong.co.uk

Tobias Markmann
mailto:tobias.markmann@isode.com
xmpp:tm@ayena.de

2019-11-13
Version 1.0.0

Status	Type	Short Name
Draft	Standards Track	hashes

This document provides a common wire format for the transport of cryptographic hash function references and hash function values in XMPP protocol extensions.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

This XMPP Extension Protocol has been contributed in full conformance with the XSF’s Intellectual Property Rights Policy (a copy of which can be found at <https://xmpp.org/about/xsf/ipr-policy> or obtained by writing to XMPP Standards Foundation, P.O. Box 787, Parker, CO 80134 USA).

Contents

1 Introduction 1

2 Requirements 1

3 XML Format 1

4 Hash Functions and Recommendations 2

5 Determining Support 3

6 Recommendations for New XMPP Extensions 3

7 Security Considerations 4

8 IANA Considerations 4

9 XMPP Registrar Considerations 4
- 9.1 Protocol Namespaces 4
- 9.2 Protocol Versioning 4
- 9.3 Service Discovery Features 4

10 XML Schema 6

11 Acknowledgements 7

1 Introduction

Various XMPP extensions make use of cryptographic hash functions, but they do so in different ways (e.g., some define XML elements and some define XML attributes) and often mandate support for different algorithms. The lack of a consistent approach to the use of cryptographic hash functions in XMPP extensions can lead to interoperability problems and security vulnerabilities. Therefore, this document recommends a common approach and XML element that can be re-used in any XMPP protocol extension.

2 Requirements

This extension is designed to meet the following criteria:

Agility It is absolutely necessary to support more secure cryptographic hash functions as they become available, and to stop supporting less secure functions as they are deprecated. The wire format should make it easy to use multiple hash functions at the same time.

Reusability The extension needs to be reusable in any XMPP protocol.

3 XML Format

This document defines a new XML element that can be used in any XMPP protocol extension. An example follows.

```
<hash xmlns='urn:xmpp:hashes:2' algo='sha-256'>2XarmwTlNxDAMkvymloX3S5
    +VbylNrJt/l5QyPa+YoU=</hash>
```

An XMPP protocol can include more than one instance of the <hash/> element, as long as each one has a different value for the 'algo' attribute:

```
<hash xmlns='urn:xmpp:hashes:2' algo='sha-1'>2
    AfMGH8O7UNPTvUVAM9aK13mpCY=</hash>
<hash xmlns='urn:xmpp:hashes:2' algo='sha-256'>2XarmwTlNxDAMkvymloX3S5
    +VbylNrJt/l5QyPa+YoU=</hash>
```

In certain scenarios it makes sense to communicate the hash algorithm that is used prior to the calculation of the hash value.

```
<hash-used xmlns='urn:xmpp:hashes:2' algo='sha-256'/>
```

The value of the 'algo' attribute MUST be one of the values from the IANA Hash Function Textual Names Registry [1] maintained by the Internet Assigned Numbers Authority (IANA) [2], or one of the values defined in the following table.

Hash Function Name	Reference
"sha3-256"	FIPS PUB 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions FIPS PUB 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions <http://dx.doi.org/10.6028/NIST.FIPS.202>.
"sha3-512"	FIPS PUB 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions FIPS PUB 202: SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions <http://dx.doi.org/10.6028/NIST.FIPS.202>.
"blake2b-256"	RFC 7693 RFC 7693: The BLAKE2 Cryptographic Hash and Message Authentication Code (MAC) <http://tools.ietf.org/html/rfc7693>.
"blake2b-512"	RFC 7693 RFC 7693: The BLAKE2 Cryptographic Hash and Message Authentication Code (MAC) <http://tools.ietf.org/html/rfc7693>.

The digest produced by the used hash algorithm is included as the XML character data of the <hash/> element after being encoded using Base64 as specified in Section 4 of RFC 4648 [3]. Thus the character data MUST conform to the base64Binary datatype [4] as defined in XML Schema Part 2 [5]. The Base64 output MUST NOT include whitespace and MUST set padding bits to zero.

4 Hash Functions and Recommendations

Previously, this document made recommendations for specific hash functions. Those documentations have been removed in version 0.6.0 and are now found in Cryptographic Hash Function Recommendations for XMPP (XEP-0414) [6].

[1] IANA registry of Hash Function Textual Names <http://www.iana.org/assignments/hash-function-text-names>.

[2] The Internet Assigned Numbers Authority (IANA) is the central coordinator for the assignment of unique parameter values for Internet protocols, such as port numbers and URI schemes. For further information, see <http://www.iana.org/>.

[3] RFC 4648: The Base16, Base32, and Base64 Data Encodings <http://tools.ietf.org/html/rfc4648>.

[4] See <http://www.w3.org/TR/xmlschema-2/#base64Binary>.

[5] XML Schema Part 2: Datatypes <http://www.w3.org/TR/xmlschema11-2/>.

[6] XEP-0414: Cryptographic Hash Function Recommendations for XMPP <https://xmpp.org/extensions/xep-0414.html>.

5 Determining Support

If an entity supports the protocol defined herein, it MUST report that by including a Service Discovery (XEP-0030) [7] feature of "urn:xmpp:hashes:2" in response to disco#info requests, along with one service discovery feature for each algorithm it supports:

Listing 1: Service discovery information request

```
<iq from='romeo@montague.lit/orchard'
    id='uw72g176'
    to='juliet@capulet.lit/balcony'
    type='get'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 2: Service discovery information response

```
<iq from='juliet@capulet.lit/balcony'
    id='uw72g176'
    to='romeo@montague.lit/orchard'
    type='result'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <feature var='urn:xmpp:hashes:2'/>
    <feature var='urn:xmpp:hash-function-text-names:sha-256'/>
    <feature var='urn:xmpp:hash-function-text-names:sha3-256'/>
  </query>
</iq>
```

In order for an application to determine whether an entity supports this protocol, where possible it SHOULD use the dynamic, presence-based profile of service discovery defined in Entity Capabilities (XEP-0115) [8]. However, if an application has not received entity capabilities information from an entity, it SHOULD use explicit service discovery instead.

6 Recommendations for New XMPP Extensions

The XSF is strongly encouraged to incorporate hash agility into new XMPP extensions that it develops by mandating re-use of the protocol defined in this specification (instead of hash elements or attributes specific to each extension).
Specifications should take the considerations in Cryptographic Hash Function Recommendations for XMPP (XEP-0414) [9] into account.

[7] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.
[8] XEP-0115: Entity Capabilities <https://xmpp.org/extensions/xep-0115.html>.
[9] XEP-0414: Cryptographic Hash Function Recommendations for XMPP <https://xmpp.org/extensions/xep-0414.html>.

7 Security Considerations

This entire document discusses security.

8 IANA Considerations

This document requires no interaction with the IANA. However, it reuses entries from the relevant IANA registry.

9 XMPP Registrar Considerations

9.1 Protocol Namespaces

This specification defines the following XML namespace:

- urn:xmpp:hashes:2

The XMPP Registrar [10] shall include the foregoing namespace in its registry at <https://xmpp.org/registrar/namespaces.html>, as governed by XMPP Registrar Function (XEP-0053) [11].

9.2 Protocol Versioning

If the protocol defined in this specification undergoes a revision that is not fully backwards-compatible with an older version, the XMPP Registrar shall increment the protocol version number found at the end of the XML namespaces defined herein, as described in Section 4 of XEP-0053.

9.3 Service Discovery Features

An entity SHOULD provide one service discovery feature for each algorithm it supports. Ideally these features would be of the form "urn:iana:hash-function-text-names:foo" (where "foo" is the name of an algorithm registered with the IANA); however there is no urn:iana namespace at present. Until there is, we use features of the form "urn:xmpp:hash-function-text-names:foo" instead. Therefore the registry submission is as follows.

[10] The XMPP Registrar maintains a list of reserved protocol namespaces as well as registries of parameters used in the context of XMPP extension protocols approved by the XMPP Standards Foundation. For further information, see <https://xmpp.org/registrar/>.

[11] XEP-0053: XMPP Registrar Function <https://xmpp.org/extensions/xep-0053.html>.

```
<var>
  <name>urn:xmpp:hash-function-text-names:md5</name>
  <desc>Support for the MD5 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>

<var>
  <name>urn:xmpp:hash-function-text-names:sha-1</name>
  <desc>Support for the SHA-1 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>

<var>
  <name>urn:xmpp:hash-function-text-names:sha-224</name>
  <desc>Support for the SHA-224 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
  <name>urn:xmpp:hash-function-text-names:sha-256</name>
  <desc>Support for the SHA-256 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
  <name>urn:xmpp:hash-function-text-names:sha-384</name>
  <desc>Support for the SHA-384 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
  <name>urn:xmpp:hash-function-text-names:sha-512</name>
  <desc>Support for the SHA-512 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>

<var>
  <name>urn:xmpp:hash-function-text-names:sha3-224</name>
  <desc>Support for the SHA3-224 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
  <name>urn:xmpp:hash-function-text-names:sha3-256</name>
  <desc>Support for the SHA3-256 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
  <name>urn:xmpp:hash-function-text-names:sha3-384</name>
  <desc>Support for the SHA3-384 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
```

```
  <name>urn:xmpp:hash-function-text-names:sha3-512</name>
  <desc>Support for the SHA3-512 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>

<var>
  <name>urn:xmpp:hash-function-text-names:id-blake2b160</name>
  <desc>Support for the BLAKE2b-160 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
  <name>urn:xmpp:hash-function-text-names:id-blake2b256</name>
  <desc>Support for the BLAKE2b-256 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
  <name>urn:xmpp:hash-function-text-names:id-blake2b384</name>
  <desc>Support for the BLAKE2b-384 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
<var>
  <name>urn:xmpp:hash-function-text-names:id-blake2b512</name>
  <desc>Support for the BLAKE2b-512 hashing algorithm</desc>
  <doc>XEP-0300</doc>
</var>
```

10 XML Schema

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:xmpp:hashes:2'
    xmlns='urn:xmpp:hashes:2'
    elementFormDefault='qualified'>

  <xs:element name='hash'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:base64Binary'>
          <xs:attribute name='algo' type='xs:NCName' use='required'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name='hash-used'>
```

```
    <xs:complexType>
      <xs:extension base='empty'>
        <xs:attribute name='algo' type='xs:NCName' use='required'/>
      </xs:extension>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

11 Acknowledgements

Thanks to Dave Cridland, Waqas Hussain, Glenn Maynard, Remko Tronçon, Paul Schaub, Christian Schudt, and Florian Schmaus for their input.