XEP-0448: Encryption for stateless file sharing

Marvin Wißfeld
mailto:xmpp@larma.de
xmpp:jabber@larma.de

2022-07-17
Version 0.2.0

Status	Type	Short Name
Experimental	Standards Track	esfs

This specification provides a protocol for sharing encrypted files using the stateless file sharing protocol (XEP-0447).

Legal

Copyright

This XMPP Extension Protocol is copyright © 1999 – 2024 by the XMPP Standards Foundation (XSF).

Permissions

Permission is hereby granted, free of charge, to any person obtaining a copy of this specification (the ”Specification”), to make use of the Specification without restriction, including without limitation the rights to implement the Specification in a software program, deploy the Specification in a network service, and copy, modify, merge, publish, translate, distribute, sublicense, or sell copies of the Specification, and to permit persons to whom the Specification is furnished to do so, subject to the condition that the foregoing copyright notice and this permission notice shall be included in all copies or substantial portions of the Specification. Unless separate permission is granted, modified works that are redistributed shall not contain misleading information regarding the authors, title, number, or publisher of the Specification, and shall not claim endorsement of the modified works by the authors, any organization or project to which the authors belong, or the XMPP Standards Foundation.

Warranty

NOTE WELL: This Specification is provided on an ”AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE.

Liability

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall the XMPP Standards Foundation or any author of this Specification be liable for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising from, out of, or in connection with the Specification or the implementation, deployment, or other use of the Specification (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if the XMPP Standards Foundation or such author has been advised of the possibility of such damages.

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 Use Cases **1**

3.1 Sharing a file 2

3.2 Receiving a file 3

3.3 Attaching a source 3

4 Ciphers **4**

5 Security Considerations **4**

6 IANA Considerations **4**

7 XMPP Registrar Considerations **5**

7.1 Protocol Namespaces 5

1 Introduction

End-to-end encrypted messaging is a popular feature within the community. Various protocols like OpenPGP for XMPP (XEP-0373) [1] or OMEMO Encryption (XEP-0384) [2] have been proposed to allow sending encrypted messages. Signaling WebRTC datachannels in Jingle (XEP-0343) [3] and Jingle Encrypted Transports (XEP-0391) [4] specify protocols for establishing an encrypted transport using Jingle to share files using Jingle File Transfer (XEP-0234) [5].
Stateless file sharing (XEP-0447) [6] describes a protocol that can be used to share files, previously uploaded using HTTP File Upload (XEP-0363) [7], but lacks means of encrypting files. This leaves files uploaded using HTTP File Upload (XEP-0363) [8] without any standardized means of encrypting them.
This XEP describes a protocol building on top of Stateless file sharing (XEP-0447) [9] to allow encrypting files.

2 Requirements

- Make use of existing protocols for end-to-end encryption (OpenPGP for XMPP (XEP-0373) [10] and Stanza Content Encryption (XEP-0420) [11])
- Reuse existing protocols for the actual transport of the data
- Allow caching and forwarding without being required to decrypt the file
- Backwards compatibility with existing, widely-deployed protocols[12]

3 Use Cases

This protocol is only meaningful for end-to-end encrypted file sharing when transported as end-to-end encrypted XML, like it's possible using Stanza Content Encryption (XEP-0420) [13]. However, usage without such end-to-end encryption still has its usecase, as it allows sharing

[1] XEP-0373: OpenPGP for XMPP <https://xmpp.org/extensions/xep-0373.html>.
[2] XEP-0384: OMEMO Encryption <https://xmpp.org/extensions/xep-0384.html>.
[3] XEP-0343: Signaling WebRTC datachannels in Jingle <https://xmpp.org/extensions/xep-0343.html>.
[4] XEP-0391: Jingle Encrypted Transports <https://xmpp.org/extensions/xep-0391.html>.
[5] XEP-0234: Jingle File Transfer <https://xmpp.org/extensions/xep-0234.html>.
[6] XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.
[7] XEP-0363: HTTP File Upload <https://xmpp.org/extensions/xep-0363.html>.
[8] XEP-0363: HTTP File Upload <https://xmpp.org/extensions/xep-0363.html>.
[9] XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.
[10] XEP-0373: OpenPGP for XMPP <https://xmpp.org/extensions/xep-0373.html>.
[11] XEP-0420: Stanza Content Encryption <https://xmpp.org/extensions/xep-0420.html>.
[12] There is a widely-deployed protocol for encrypted file sharing known as "OMEMO media sharing" or "aesgcm-links" that was never accepted as a XEP. While backwards compatibility with such non-standard is not a maxime of the XSF, it was still considered during the design of this protocol.
[13] XEP-0420: Stanza Content Encryption <https://xmpp.org/extensions/xep-0420.html>.

files through untrusted intermediaries for as long as the intermediary XMPP servers, if any, are trusted.
Note: To make the examples in this document more readable, no end-to-end encryption is used.

3.1 Sharing a file

Before sharing the file, the sending entity MUST create random symmetric private key and initialization vector (IV) as required by the selected encryption cipher (see Ciphers). The file is then encrypted using selected encryption cipher and the generated key and IV. After this it can be uploaded using HTTP File Upload (XEP-0363) [14] or prepared for any other means of file sharing.
The file is then shared using the protocol described in Stateless file sharing (XEP-0447) [15]. The <file/> metadata element still refers to the original file, i.e. it describes the original file name, size and hashes. The <size/> element and one or multiple <hash/> elements are REQUIRED when sending encrypted files.
For the encrypted file, a source is added as an <encrypted/> element to the <sources/>. It carries an attribute cipher with the namespace of the encryption cipher being used. The <encrypted/> element contains a <key/> and an <iv/> element, containing both values as Base64-encoded strings. The <encrypted/> element MAY also include <hash/> elements as described in Use of Cryptographic Hash Functions in XMPP (XEP-0300) [16], referring to the hash of the encrypted file. At last, the <encrypted/> element also includes another <sources/> element as described in Stateless file sharing (XEP-0447) [17], specifying sources to obtain the encrypted file. The outer <sources/> may contain additional sources that directly allow for end-to-end encrypted file transfers, for example Jingle File Transfer (XEP-0234) [18] using Jingle Encrypted Transports (XEP-0391) [19].

Listing 1: Sharing summit.jpg with juliet@shakespeare.lit using encryption

```
<message to='juliet@shakespeare.lit' from='romeo@montague.lit/resource
    ' id='sharing-a-file'>
  <file-sharing xmlns='urn:xmpp:sfs:0'>
    <file xmlns='urn:xmpp:file:metadata:0'>
      <media-type>image/jpeg</media-type>
      <name>summit.jpg</name>
      <size>3032449</size>
      <dimension>4096x2160</dimension>
      <hash xmlns='urn:xmpp:hashes:2' algo='sha3-256'>2
          XarmwTlNxDAMkvymloX3S5+VbylNrJt/l5QyPa+YoU=</hash>
```

[14] XEP-0363: HTTP File Upload <https://xmpp.org/extensions/xep-0363.html>.
[15] XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.
[16] XEP-0300: Use of Cryptographic Hash Functions in XMPP <https://xmpp.org/extensions/xep-0300.html>.
[17] XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.
[18] XEP-0234: Jingle File Transfer <https://xmpp.org/extensions/xep-0234.html>.
[19] XEP-0391: Jingle Encrypted Transports <https://xmpp.org/extensions/xep-0391.html>.

```
      <hash xmlns='urn:xmpp:hashes:2' algo='id-blake2b256'>2
          AfMGH8O7UNPTvUVAM9aK13mpCY=</hash>
      <desc>Photo from the summit.</desc>
      <thumbnail xmlns='urn:xmpp:thumbs:1' uri='cid:sha1+
          ffd7c8d28e9c5e82afea41f97108c6b4@bob.xmpp.org' media-type='
          image/png' width='128' height='96'/>
    </file>
    <sources>
      <encrypted xmlns='urn:xmpp:esfs:0' cipher='urn:xmpp:ciphers:aes
          -256-gcm-nopadding:0'>
        <key>SuRJ2agVm/pQbJQlPq/B23Xt1YOOJCcEGJA5HrcYOGQ=</key>
        <iv>T8RDMBaiqn6Ci4Nw</iv>
        <hash xmlns='urn:xmpp:hashes:2' algo='sha3-256'>
            BgKI2gp2kNCRsARNvhFmw5kFf9BBo2pTbV2D8XHTMWI=</hash>
        <hash xmlns='urn:xmpp:hashes:2' algo='id-blake2b256'>id4cnqqy9
            /ssfCkM4vYSkiXXrlE=</hash>
        <sources xmlns='urn:xmpp:sfs:0'>
          <url-data xmlns='http://jabber.org/protocol/url-data' target
              ='https://download.montague.lit/4a771ac1-f0b2-4a4a-9700-
              f2a26fa2bb67/encrypted.jpg' />
        </sources>
      </encrypted>
      <jinglepub xmlns='urn:xmpp:jinglepub:1' from='romeo@montague.lit
          /resource' id='9559976B-3FBF-4E7E-B457-2DAA225972BB'>
        <description xmlns='urn:xmpp:jingle:apps:file-transfer:5' />
      </jinglepub>
    </sources>
  </file-sharing>
</message>
```

3.2 Receiving a file

On receive of a message including a <file-sharing/> element, that has an <encrypted/> element in its sources, normal processing as described in Stateless file sharing (XEP-0447) [20] applies. When the receiving entity tries to obtain the file from the source described by the <encrypted/> element, it will try to obtain any of its inner sources instead. On success, it decrypts the obtained file using the encryption cipher, private key and IV provided. If the resulting file is larger than the number of bytes specified in the <size/> metadata element, the additional bytes are cut off.

3.3 Attaching a source

The protocol to attach a source described in Stateless file sharing (XEP-0447) [21] can also be used to attach encrypted sources. After receiving a file using encrypted means, it is

[20] XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.
[21] XEP-0447: Stateless file sharing <https://xmpp.org/extensions/xep-0447.html>.

RECOMMENDED to only attach additional sources that support encryption.

4 Ciphers

Note The following table was copied from Jingle Encrypted Transports (XEP-0391) [22].
In order to encrypt the file, the sending entity must transmit a cipher key to the responder. There are multiple options available:

Namespace	Type	Length (bits)	Parameters
urn:xmpp:ciphers:aes-128-gcm-nopadding:0	AES	Key: 128, IV: 96	GCM/NoPadding
urn:xmpp:ciphers:aes-256-gcm-nopadding:0	AES	Key: 256, IV: 96	GCM/NoPadding
urn:xmpp:ciphers:aes-256-cbc-pkcs7:0	AES	Key: 256, IV: 128	CBC/PKCS#7

For compatibility reasons, it is RECOMMENDED to append the GCM authentication tag to the uploaded file when using any AES cipher with GCM. The GCM authentication tag is not needed when using the protocol described in this document as a hash of the resulting file is transported independently.

5 Security Considerations

Yes.

6 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [23].

[22] XEP-0391: Jingle Encrypted Transports <https://xmpp.org/extensions/xep-0391.html>.
[23] 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/>.

7 XMPP Registrar Considerations

7.1 Protocol Namespaces

The XMPP Registrar [24] includes 'urn:xmpp:esfs:0' in its registry of protocol namespaces (see <https://xmpp.org/registrar/namespaces.html>).

- urn:xmpp:esfs:0

[24] 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/>.