XEP-0052: File Transfer

Thomas Muldowney
mailto:temas@box5.net
xmpp:temas@jabber.org

Matthew Miller
mailto:linuxwolf@outer-planes.net
xmpp:linuxwolf@outer-planes.net

Justin Karneges
mailto:justin@affinix.com
xmpp:justin@andbit.net

2018-11-03
Version 0.2.1

Status	Type	Short Name
Retracted	Standards Track	N/A

A protocol for transferring a file between two Jabber IDs.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction **1**

2 Use Cases **1**

2.1 Send File Use Case 1

3 Basic Usage **2**

4 Stream Relation **5**

4.1 ”iq:oob” Relation 5

5 Formal Description **5**

5.1 DTD 5

5.2 <file/> Element 6

5.3 <desc/> Element 7

5.4 <range/> Element 7

5.5 Error Descriptions 7

6 Security Considerations **7**

7 IANA Considerations **8**

8 JANA Considerations **8**

1 Introduction

This document describes the http://www.jabber.org/protocol/filexfer namespace, which is used for offering and transferring files from one Jabber ID to another. It tries to expand the basic method (iq:oob) that currently exists to allow for numerous stream methods, and more detailed file information before accepting an offer. This document only describes the negotiation method and suggests how streams could link back to the negotiated information.

2 Use Cases

This document covers one use case of sending a file to another user. Future specifications may enhance this to include searching and offering.

2.1 Send File Use Case

Primary Flow:

1. Determine if the receiver supports FT through disco/browse. [E1]
2. Sender sends metadata and available methods to receiver
3. Receiver sends the accepted method to Sender and any range/offset information. [E2],[E3]
4. Sender and Receiver establish the negotiated method[E4]
5. Sender sends data as described by method
6. After the stream closes the Receiver notifies the Sender of completion. [E5]
7. END

Errors Conditions:

1. User does not support filetransfer. END
2. Receiver rejects send. END
3. Receiver does not have any methods shared with the sender. END
4. The stream is unable to be started. END
5. The Receiver notifies sender of an error transferring. END

3 Basic Usage

In order to send a file, the sender must first tell the receiver a little bit about the file to make sure they will accept it. At the same time they list the stream methods they support in the order they wish to use them. This is done by sending the information in the http://www.jabber.org/protocol/filexfer namespace.

Listing 1: Generic Offer

```
<iq type='set' id='ft_1' to='recvr@jabber.org/Home'>
      <file xmlns='http://www.jabber.org/protocol/filexfer'
              action='offer'
              id='a0'
              name='myfile.txt'
              size='1024'
              mime-type='text/plain'>
        <feature xmlns='http://jabber.org/protocol/feature-neg'>
          <x xmlns='jabber:x:data'>
            <field var='file-transfer-method' type='list-single'>
              <option><value>jabber:iq:oob</value></option>
            </field>
          </x>
        </feature>
      </file>
    </iq>
```

That is the basic request, a more complete requeset with range support is shown below.

Listing 2: Complete File Offer

```
<iq type='set' id='ft_1' to='recvr@jabber.org/Home'>
      <file xmlns='http://www.jabber.org/protocol/filexfer'
            action='offer'
            id='a0'
            name='myfile.txt'
            size='1024'
            mime-type='text/plain'
            date='20020412T00:00:00'
            hash='23e4ad6b63343b33a333c334'>
        <desc>A cool file</desc>
        <range/>
        <feature xmlns='http://jabber.org/protocol/feature-neg'>
          <x xmlns='jabber:x:data'>
            <field var='file-transfer-method' type='list-single'>
              <option><value>jobs</value></option>
              <option><value>dtcp</value></option>
              <option><value>jabber:iq:oob</value></option>
              <option><value>ibb</value></option>
            </field>
```

```
          </x>
        </feature>
      </file>
    </iq>
```

If a receiver decides to accept an offered file they request it from the sending with an <iq/> type result. The receiver sends back the id of the file being sent, the method they wish to use, and the range they wish to download (if the sender announced support). When range support is being used the receiver MUST specify the length and MAY specify a beginning offset with the acceptance.

Listing 3: Request the Offered File

```
<iq type='result' id='ft_req_1' to='sender@jabber.org/res'>
      <file xmlns='http://www.jabber.org/protocol/filexfer' id='a0'
          action='get'>
        <feature xmlns='http://jabber.org/protocol/feature-neg'>
          <x xmlns='jabber:x:data'>
            <field var='file-transfer-method'>
              <option><value>jabber:iq:oob</value></option>
            </field'>
          ␣␣␣␣␣␣␣␣␣␣</x>
        ␣␣␣␣␣␣␣␣</feature>
      ␣␣␣␣␣␣</file>
    ␣␣␣␣</iq>
```

Listing 4: Accept the Offered File with a Range and Offset

```
<iq type='result' id='ft_req_q' to='sender@jabber.org/res'>
      <file xmlns='http://www.jabber.org/protocol/filexfer' id='a0'
          action='get'>
        <range offset='100' length='500' />
          <x xmlns='jabber:x:data'>
            <field var='file-transfer-method'>
              <option><value>jobs</value></option>
            </field>
          </x>
        </feature>
      </file>
    </iq>
```

If the receiver decides to not accept the file they SHOULD send back an error 403 to the sender.

Listing 5: Declining the Offered File

```
<iq type='error' id='ft_1' to='sender@jabber.org/res'>
      <error code='403'>Offer Declined</error>
    </iq>
```

If the receiver does not support any of the offered stream methods they SHOULD send back an error 406 to the sender.

Listing 6: No Acceptable Methods

```
<iq type='error' id='ft_1' to='sender@jabber.org/res'>
      <error code='406'>No Acceptable Methods</error>
    </iq>
```

At this point the sender will setup the stream method and begin to transfer data. The stream itself can use the file transfer namespace to tie the metadata to the actual data sent, this is illustrated below using iq:oob.

Listing 7: Starting an iq:oob transfer

```
<iq type='set' id='ft_oob_1' to='recvr@jabber.org/Home'>
      <file xmlns='http://www.jabber.org/protocol/filexfer' id='a0'
          action='start'/>
      <query xmlns='jabber:iq:oob'>
        <url>http://www.jabber.org/file.txt</url>
        <desc>Here is the file</desc>
      </query>
    </iq>
```

If the receiver is unable to start the negotiated stream for any reason they should send an <error/> with a 502 code to the sender.

Listing 8: Unable to Start Stream

```
<iq type='error' id='ft_oob_e_1' to='sender@jabber.org/res'>
      <file xmlns='http://www.jabber.org/protocol/filexfer' id='a0'
          action='error'/>
      <error code='502'>Unable to Start Stream</error>
    </iq>
```

Once the data has been transferred the receiver SHOULD send the sender a notification that the transfer completed. This is done by sending an <iq/> type set with the file id and a completed action.

Listing 9: Completed Transfer Notification

```
<iq type='set' id='ft_c_1' to='sender@jabber.org/res'>
      <file xmlns='http://www.jabber.org/protocol/filexfer' id='a0'
          action='complete' />
    </iq>
```

If the transfer does not complete, for any reason after the metadata negotiation, the party that has the error SHOULD send an error 500 and the file id to the other party.

Listing 10: Failed Transfer Error

```
<iq type='error' id='ft_1' to='sender@jabber.org/res'>
      <error code='500'>File Transfer Failed</error>
      <file xmlns='http://www.jabber.org/protocol/filexfer' id='a0'
         action='error'/>
    </iq>
```

4 Stream Relation

By staying in just the realm of negotiating the metadata to a file, we allow for multiple transport layers, or streams, to be used. Some streams will need to tie the metadata to the actual data transfer, to help accomodate this the stream may use the <file/> with an action of start and the correct id. The <file/> could be transported in the stream negotiations, or along side it. Although this spec does not mandate any specific methods to new stream authors, it does provide the syntax for the currently existing "iq:oob" system.

4.1 "iq:oob" Relation

For an "iq:oob" transfer to be related to its metadata, a <file/> is transported along side the <query/>. The id used on the <file/> is the id for the metadata of the actual data that is being sent. The action on the <file/> is "start". An example of this can be found in the Basic Usage section.

5 Formal Description

5.1 DTD

```
<!ELEMENT file  ( ( desc )? | ( range )? | ( PCDATA )* ) >
<!ELEMENT desc    ( #PCDATA )*  >
<!ELEMENT range   EMPTY         >

<!ATTLIST file
    id        CDATA   #REQUIRED
    action    "offer" | "get" | "complete" | "start" | "error"  #
       IMPLIED "offer"
    name      CDATA   #OPTIONAL
    size      CDATA   #OPTIONAL >
    mime-type CDATA   #OPTIONAL
    date      CDATA   #OPTIONAL
    hash      CDATA   #OPTIONAL
<!ATTLIST range
    length    CDATA   #OPTIONAL
```

```
    offset     CDATA    #OPTIONAL >
```

5.2 <file/> Element

The <file/> element is the ”workhorse” element. This element is used to convey metadata and report file transfer actions. This elemnt contains attributes for file metadata and actions, and MAY contain a <desc/>, a <range/>, and zero or more <feature xmlns='jabber:iq:negotiate'/> (Feature Negotiation (XEP-0020) [1]) elements.
The ”id” attribute specifies the identifier for this particular file transfer. This attribute MUST be present at all times. There are no value requirements other than it MUST be unique between the sender and receiver.
The ”action” attribute specifies the action to undertake with the given file. This attribute SHOULD be present in most cases. If not present, the value ”offer” is implied. The value of ”action” MUST be one of the following:

Value	Description
complete	The file transfer is complete.
get	The file transfer should start.
offer	The file transfer is offered (metadata MUST be present)
start	The file transfer is starting.
error	The file transfer has failed. The outlying error tag has more information.

The ”name” attribute specifies the file name. This attribute MUST be present if the action is ”offer”, otherwise it SHOULD NOT be present.
The ”size” attribute specifies the file size, in bytes. This attribute MUST be present if the action is ”offer”, otherwise it SHOULD NOT be present.
The ”mime-type” attribute specifies the MIME-type for the file. This attribute SHOULD be present if the action is ”offer”, otherwise it SHOULD NOT be present. The value of this attribute MUST follow the specification for MIME-types from RFC-2046[2].
The ”date” attribute specifies the file date. This attribute MAY be present if the action is ”offer”, otherwise it SHOULD NOT be present. The value MUST follow the specification for ISO 8601 date/time formats[3].
The ”hash” attribute specifies the hash of the file contents. This attribute MAY be present if the action is ”offer”, otherwise it SHOULD NOT be present. The value MUST be an SHA1 hash of the file contents.

[1] XEP-0020: Feature Negotiation <https://xmpp.org/extensions/xep-0020.html>.
[2] RFC 2046: ”Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types” -- http://www.ietf.org/rfc/rfc2046.txt
[3] ISO 8601: ”[Summary of the] International Standard Date and Time Notation” -- http://www.cl.cam.ac.uk/~{}mgk25/iso-time.html

5.3 <desc/> Element

The <desc/> element contains a human-readable description of the file. This element has no attributes, and contains character data content.

5.4 <range/> Element

The <range/> element describes range information for a partial transfer. This element has attributes to define the range length and range offset. This element contains no content.
The "length" attribute defines the range length, in bytes. This attribute MUST be present if the containing <file/> has an action value of "get", otherwise it SHOULD NOT be present. The value of this attribute MUST be an integer value and MUST be less than or equal to the (size + offset) of the file.
The "offset" attribute defines the range offset, in bytes. This attribute MAY be present if the containing <file/> has an action value of "get", otherwise it SHOULD NOT be present. If this attribute is not present, a value of 0 is implied. The value of this attribute MUST be an integer, MUST NOT be less than 0, and MUST be less than (size - length).

5.5 Error Descriptions

There are three main error conditions in file transfer. Following are the conditions, error codes and descriptions:

- *Declining Transfer (403)*: During the metadata negotiation the receiver may decline the transfer by sending the 403 error. The <error/> CDATA MAY contain a descriptive reason why, but is not necessary.
- *No Available Methods (406)*: When the sender presents the available stream methods, and the receiver can not use any of them, they send a 406 error. The <error/> CDATA is not important.
- *Transfer Failed (500)*: If the file transfer fails for any reason after negotiation, the error generator SHOULD send a 500 error to the other party. This is the only error message that both the sender and reciever may send. The <error/> CDATA MAY contain information about the failure.
- *Unable to Start Stream (502)*: When the receiver is unable to start the negotiated stream method they send a 502 error to the sender. The <error/> CDATA is not important.

6 Security Considerations

Data integrity can be checked with the sha1 of the file that is sent. This could be attacked via a man in the middle attack, but much more embarrasing things could result from that than a

bad file. The wire integrity is left to the stream method.

7 IANA Considerations

The mime-type attribute on <file/> is a valid MIME type as controlled by the IANA.

8 JANA Considerations

The ”http://jabber.org/protocol/filexfer” is the only namespace that needs to be registered with the JANA.