XEP-0135: File Sharing

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

2004-06-04
Version 0.1

Status	Type	Short Name
Deferred	Standards Track	files

This document specifies a simple extension to existing protocols for file sharing over Jabber/XMPP.

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

Existing Jabber protocols provide a strong foundation for the controlled, permissions-based sharing of files between Jabber entities, e.g., to enable shared workspaces among ad-hoc workgroups and the attachment of files to Multi-User Chat (XEP-0045) [1] rooms.
This document defines several additional building blocks (a simple request protocol along with well-known service discovery nodes) that tie together existing protocols to enable the sharing of files between Jabber entities.

2 Requirements

1. Enable a requesting entity to find files shared by an offering entity without depending on third parties (such as Publish-Subscribe (XEP-0060) [2] services or dedicated file-hosting services) (REQUIRED).
2. Enable a requesting entity to discover detailed file information (OPTIONAL).
3. Enable a requesting entity to ask for files shared by an offering entity (REQUIRED).
4. Re-use existing Jabber/XMPP protocols wherever possible.

3 Assumptions

The protocol defined herein assumes that the offering entity is the file owner. The primary reason for this approach is that most likely the file owner (e.g., a chatroom or end user) knows best which files are available and which requesting entities have permission to view which files. Furthermore, for the purposes of this protocol it is unnecessary to assume that dedicated file-hosting services will exist on the Jabber network or that existing Jabber servers will offer file-hosting functionality (though see the File-Hosting Services section of this document).

4 Discovering Support and Address of Offering Entity

4.1 Offering Entity is File Owner

If an entity directly supports the protocol defined herein, it SHOULD include a feature of "http://jabber.org/protocol/files" in its response to a Service Discovery (XEP-0030) [3] information request. The protocol flow is shown in the following example (an end user querying a chatroom):

[1] XEP-0045: Multi-User Chat <https://xmpp.org/extensions/xep-0045.html>.
[2] XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.
[3] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.

Listing 1: User Queries a Room Regarding Support

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='disco1'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 2: Room Returns Info

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='disco3'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    ...
    <feature var='http://jabber.org/protocol/files'/>
    ...
  </query>
</iq>
```

This document stipulates that communications regarding files MUST occur by sending stanzas to the well-known service discovery node "files" (or sub-nodes thereof as defined below). Therefore, even if (as in the foregoing example) the file owner directly supports the protocol defined herein, the requesting entity MUST send subsequent file-related service discovery requests to the node "files" (or sub-nodes thereof). The file owner also SHOULD list that node in its response to a service discovery items request, as shown in the following example:

Listing 3: User Queries a Room Regarding Items

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='disco2'>
  <query xmlns='http://jabber.org/protocol/disco#items'/>
</iq>
```

Listing 4: Room Returns Items

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='disco2'>
  <query xmlns='http://jabber.org/protocol/disco#items'>
    ...
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='files'
          name='The␣files␣attached␣to␣this␣room'/>
```

```
    ...
  </query>
</iq>
```

4.2 Offering Entity is not File Owner

It is possible that the file owner does not directly support the protocol defined herein and therefore that the offering entity has a JID different from that of the file owner. In this case, the file owner MUST NOT include a feature of "http://jabber.org/protocol/files" in its response to a service discovery information request, as shown in the following example (an end user querying another end user):

Listing 5: User Queries Contact Regarding Support

```
<iq type='get'
    from='juliet@capulet.com/chamber'
    to='romeo@montague.net/home'
    id='disco3'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 6: Contact Returns Info

```
<iq type='result'
    from='romeo@montague.net/home'
    to='juliet@capulet.com/chamber'
    id='disco3'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    ...
  </query>
</iq>
```

However, in this case the file owner SHOULD still include the offering entity (e.g., a hosting service) in its response to a service discovery items request:

Listing 7: User Queries Contact Regarding Items

```
<iq type='get'
    from='juliet@capulet.com/chamber'
    to='romeo@montague.net/home'
    id='disco4'>
  <query xmlns='http://jabber.org/protocol/disco#items'/>
</iq>
```

Listing 8: Contact Returns Items

```
<iq type='result'
```

```
    from='romeo@montague.net/home'
    to='juliet@capulet.com/chamber'
    id='disco4'>
  <query xmlns='http://jabber.org/protocol/disco#items'>
    ...
    <item jid='romeo@files.shakespeare.lit'
          node='files'
          name='Romeo&apos;s Files'/>
    ...
  </query>
</iq>
```

5 Finding Files

5.1 Determining if Files are Available

If an offering entity has files to share, it SHOULD reply positively when a requesting entity sends a service discovery items request to the well-known service discovery node "files":

Listing 9: Requesting the File List

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='avail'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files'/>
</iq>
```

If the requesting entity is not allowed to view the offering entity's files (the requesting entity is not an occupant of a chatroom, is not registered with the offering entity, is not a contact in a user's roster, etc.) or the offering entity has no files to share, the offering entity SHOULD return an empty <query/> element:

Listing 10: No Files to Share

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='avail'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files'/>
</iq>
```

If the requesting entity is allowed to view the offering entity's files and the offering entity has files to share, the offering entity SHOULD return a list of items:

Listing 11: Returning the File List

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='avail'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files'>
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='files/somedir'/>
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='files/somefile'
          name='file1'/>
  </query>
</iq>
```

Note: The NodeID MUST begin with the string 'files' followed by the '/' character followed the name of the directory or file; further subdirectories or files within a directory MUST follow the same pattern (e.g., "files/somedir/anotherfile"). Thus the protocol defined herein enforces semantic meaning on NodeIDs; this is OPTIONAL within **Service Discovery** but REQUIRED by this document.

If the offering entity has only a few files to share, it may be appropriate to make them available via service discovery only, thus requiring the requesting entity to "walk the tree" of directories and files as described in the Finding All Files via Service Discovery section. However, if the offering entity has a larger number of files to share, the number of service discovery requests and responses required to "walk the tree" of all directories and files might result in excessive amounts of traffic between the requesting entity and the offering entity; in this case, the offering entity SHOULD provide a "tree file" that defines the hierarchy of directories and files in the standardized format specified in the Retrieving the Tree File section. The number of files that counts as "large" is not defined herein and is left up to the implementation or deployment; in practice, it is RECOMMENDED for the offering entity to provide a tree file if it has more than five (5) files to share.

5.2 Finding All Files via Service Discovery

If the offering entity does not provide a tree file, the requesting entity will need to "walk the tree" via service discovery in order to find all the files shared by the offering entity. The previous example showed an offering entity that had two items available: a directory and a file. In order to determine if an item is a directory or a file, the requesting entity MUST send a disco#info request to the relevant node:

Listing 12: Requesting Further Information (1)

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
```

```
    id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somedir'/>
</iq>
```

If the item is a directory, the offering entity SHOULD return information about the directory, including an identity whose category is "filesys" and whose type is "directory":

Listing 13: Returning Further Information (1)

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somedir'>
    <identity category='filesys' type='directory'/>
  </query>
</iq>
```

If the requesting entity wants information about every item, it MUST query each item individually:

Listing 14: Requesting Further Information (2)

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='info2'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somefile'/>
</iq>
```

If the item is a file, the offering entity SHOULD return information about the file, including an identity whose category is "filesys" and whose type is "file":

Listing 15: Returning Further Information (2)

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='info2'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somefile'>
    <identity category='filesys' type='file' name='file1'/>
  </query>
</iq>
```

Note: The offering entity MAY also include detailed information about the file, as described in the Determining File Details section of this document.
If the requesting entity wants to find all files, it needs to send a disco#items query to the directory:

Listing 16: Requesting Further Items (1)

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='items1'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files/somedir'/>
</iq>
```

The offering entity will then return a list of files and directories contained within the queried directory:

Listing 17: Returning Further Items (1)

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='items1'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files/somedir'>
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='files/somedir/anotherdir'/>
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='files/somedir/anotherfile'
          name='file2'/>
  </query>
</iq>
```

The requesting entity then needs to send further disco#info and disco#items requests to the offering entity, specifying the appropriate service discovery nodes...

Listing 18: Requesting Further Information (3)

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='info3'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somedir/anotherdir'/>
</iq>
```

Listing 19: Returning Further Information (3)

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='info3'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somedir/anotherdir'>
    <identity category='filesys' type='directory'/>
  </query>
</iq>
```

Listing 20: Requesting Further Information (4)

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='info4'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somedir/anotherfile'/>
</iq>
```

Listing 21: Returning Further Information (4)

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='info4'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somedir/anotherfile'>
    <identity category='filesys' type='file' name='file2'/>
  </query>
</iq>
```

Listing 22: Requesting Further Items (2)

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='items2'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files/somedir/anotherdir'/>
</iq>
```

Listing 23: Returning Further Items (2)

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='items2'>
```

```
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files/somedir/anotherdir'>
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='files/somedir/anotherdir/yetanotherfile'
          name='file3'/>
  </query>
</iq>
```

Listing 24: Requesting Further Information (5)

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='info5'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somedir/anotherdir/yetanotherfile'/>
</iq>
```

Listing 25: Returning Further Information (5)

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='info5'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somedir/anotherdir/yetanotherfile'>
    <identity category='filesys' type='file' name='file3'/>
  </query>
</iq>
```

In this scenario, we can reconstruct the file tree as follows:

```
share
\--somefile
\--somedir
   \--anotherfile
   \--anotherdir
      \--yetanotherfile
```

5.3 Retrieving the Tree File

Obviously, finding all files via service discovery is a tedious process. Therefore, it is RECOMMENDED that the offering entity provide a "tree file" if it has more than five (5) files to share. The format of the tree file is defined by the 'http://jabber.org/profile/si/profile/tree-transfer' namespace that is specified in Tree Transfer Stream Initiation Profile (XEP-0105) [4]. The tree file MUST be named "tree.xml" and MUST be available at the well-known service discovery

[4] XEP-0105: Tree Transfer Stream Initiation Profile <https://xmpp.org/extensions/xep-0105.html>.

node "tree.xml". The offering entity MAY create a different tree file for each requesting entity (depending on the requesting entity's permissions to view certain directories and files); for this reason, the tree file SHOULD NOT be contained in the root "files" directory itself (note that its NodeID is "tree.xml", not "files/tree.xml").
If the offering entity provides a tree file, it MUST communicate that fact in the disco#items result it returns to the requesting entity in response to the initial request:

Listing 26: Requesting the File List

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='avail'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files'/>
</iq>
```

Listing 27: Returning the File List

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='avail'>
  <query xmlns='http://jabber.org/protocol/disco#items'
         node='files'>
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='files/somedir'/>
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='files/somefile'
          name='file1'/>
    <item jid='darkcave@macbeth.shakespeare.lit'
          node='tree.xml'
          name='Tree␣File'/>
  </query>
</iq>
```

If the offering entity includes a service discovery item whose NodeID is "tree.xml", the requesting entity SHOULD retrieve that file (using the protocol specified in the Retrieving a File section) before sending any further service discovery requests to the offering entity.
The following example shows the exact format of the tree file that would represent the file and directory hierarchy discovered via service discovery in the preceding section:

Listing 28: A Tree File

```
<tree xmlns='http://jabber.org/profile/si/profile/tree-transfer'
      numfiles='3'
      size='72521'>
  <directory name='files'>
```

```
    <file sid='file1' name='somefile'/>
    <directory name='somedir'>
      <file sid='file2' name='anotherfile'/>
      <directory name='anotherdir'>
        <file sid='file3' name='yetanotherfile'/>
      </directory>
    </directory>
  </directory>
</tree>
```

If the offering entity provides a tree file, it is RECOMMENDED (but not required) for the offering entity to also make information about its files discoverable via **Service Discovery** as described in the following section.

5.4 Determining File Details

As is evident from the foregoing examples, neither "walking the tree" via **Service Discovery** nor retrieving the tree file will yield the kind of detailed information about a file (MIME type, file size, descriptive text, etc.) that can help a user or application decide whether to retrieve the file.
To address the felt need for more detailed information about files, an offering entity MAY provide such information in response to disco#info requests sent to a specific NodeID (file or directory) by including extended information structured according to Service Discovery Extensions (XEP-0128) [5]. The following examples illustrate this usage.

Listing 29: Requesting File Information

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='file1'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somefile'/>
</iq>
```

Listing 30: Returning Detailed File Information

```
<iq type='result'
    from='darkcave@macbeth.shakespeare.lit'
    to='hag66@shakespeare.lit/pda'
    id='info2'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='files/somefile'>
    <identity category='filesys' type='file' name='file1'/>
    <x xmlns='jabber:x:data' type='result'>
```

[5] XEP-0128: Service Discovery Extensions <https://xmpp.org/extensions/xep-0128.html>.

```
      <field var='FORM_TYPE' type='hidden'>
        <value>http://jabber.org/protocol/files</value>
      </field>
      <field var='date'>
        <value>2004-05-12T02:37:07Z</value>
      </field>
      <field var='description'>
        <value>This is just a test file.</value>
      </field>
      <field var='hash'>
        <value>552da749930852c69ae5d2141d3766b1</value>
      </field>
      <field var='mime-type'>
        <value>text/plain</value>
      </field>
      <field var='size'>
        <value>1022</value>
      </field>
    </x>
  </query>
</iq>
```

The fields shown are RECOMMENDED, and are specified more fully in the XMPP Registrar Considerations section of this document.

6 Retrieving a File

In order to retrieve a file, the requesting entity sends a retrieval request to the JID+NodeID of the relevant item:

Listing 31: Retrieving a File

```
<iq type='get'
    from='hag66@shakespeare.lit/pda'
    to='darkcave@macbeth.shakespeare.lit'
    id='retrieve1'>
  <retrieve xmlns='http://jabber.org/protocol/files'
            node='files/somefile'/>
</iq>
```

Note: If the requested file was found by means of the tree file rather than service discovery, the NodeID of the retrieve request MUST be constructed according to the rules specified above for service discovery NodeIDs (i.e., 'files' followed by the '/' character followed by the name of the directory or file, followed by additional '/' characters and subdirectory or file names as needed).
If the offering entity agrees to share the file with the requesting entity, it MUST return an IQ

result to the requesting entity and then immediately initiate a file transfer to the requesting entity following the protocol defined in SI File Transfer (XEP-0096) [6]:

Listing 32: Offering Entity Agrees to Share File

```
<iq type='result' id='retrieve1'/>
```

Listing 33: Offering Entity Initiates File Transfer

```
<iq type='set'
    from='darkcave@macbeth.shakespeare.lit'
    id='offer1'
    to='hag66@shakespeare.lit/pda'>
  <si xmlns='http://jabber.org/protocol/si'
      id='file1'
      mime-type='text/plain'
      profile='http://jabber.org/protocol/si/profile/file-transfer'>
    <file xmlns='http://jabber.org/protocol/si/profile/file-transfer'
          name='somefile'
          size='1022'/>
    <feature xmlns='http://jabber.org/protocol/feature-neg'>
      <x xmlns='jabber:x:data' type='form'>
        <field var='stream-method' type='list-single'>
          <option><value>http://jabber.org/protocol/bytestreams</value
              ></option>
          <option><value>http://jabber.org/protocol/ibb</value></
              option>
        </field>
      </x>
    </feature>
  </si>
</iq>
```

The value of the <si/> element's 'id' attribute MUST be the same as the value of the 'sid' attribute communicated in the tree file or the 'name' attribute communicated via service discovery; for this reason, the service discovery 'name' attribute is REQUIRED for NodeIDs that correspond to files, and its value MUST follow the rules for the 'sid' attribute specified in XEP-0105.

Upon receiving the file transfer initiation from the offering entity, the requesting entity SHOULD check the SI 'id' in order to correlate the file transfer with the request; if there is a match, the requesting entity SHOULD silently accept the file transfer and not require intervention by a human before proceeding.

If the offering entity does not agree to share the file with the requesting entity, it MUST return an appropriate IQ error to the requesting entity, such as "Not Authorized", "Forbidden", "Payment Required", "Registration Required", or "Not Found" (see Error Condition Mappings

[6] XEP-0096: SI File Transfer <https://xmpp.org/extensions/xep-0096.html>.

(XEP-0086) [7] regarding error syntax).

7 File-Hosting Services

A dedicated file-hosting service may agree to host files on behalf of a user or other entity, in which case the hosting service (or, to be precise, a specific resource of the hosting service) becomes the offering entity in the use cases defined herein. While the nature of such hosting services is outside the scope of this document, the following guidelines may be helpful to implementers.

First, a file-hosting service SHOULD provide a distinct JID for each account on the service in order to enable communications between the requesting entity and the hosting service. For example, let us suppose that <files.shakespeare.lit> is a file-hosting service; specific accounts on the service could be structured as JIDs of the form <account@files.shakespeare.lit> or <files.shakespeare.lit/account>, in which case the requesting entity would communicate directly with that JID and treat that JID as the offering entity. The file-hosting service SHOULD enable the file owner (e.g., an end user whose JID is <romeo@montague.net>) to upload files to the service using standard Internet protocols (such as HTTP, FTP, scp, or XEP-0096), control who can view or retrieve files, and otherwise configure the offering entity. The file owner SHOULD also list the JID of the "offering entity" in response to service discovery items requests sent to the user's bare JID, so that requesting entities can find files hosted by the service on the file owner's behalf.

It is possible that some Jabber server deployments would choose to offer file-hosting capabilities for their users (if supported in the underlying server implementation), so that the offering entity would have the same <user@host> address as the file owner. In this case, the server itself can be considered a file-hosting service.

8 Security Considerations

Managing access to files and directories is the responsibility of the offering entity. However, the offering entity SHOULD NOT share files with requesting entities that are not known to it via presence subscription, prior registration, room occupancy, or some similar mechanism.

9 IANA Considerations

No interaction with the Internet Assigned Numbers Authority (IANA) [8] is required as a result of this document.

[7] XEP-0086: Error Condition Mappings <https://xmpp.org/extensions/xep-0086.html>.

[8] 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/>.

10 XMPP Registrar Considerations

10.1 Protocol Namespaces

Upon advancement of this document to a status of Draft, the XMPP Registrar [9] shall add the 'http://jabber.org/protocol/files' namespace to its registry of protocol namespaces.

10.2 Well-Known Service Discovery Nodes

Upon advancement of this document to a status of Draft, the XMPP Registrar shall add 'files' and 'tree.xml' to its registry of well-known service discovery nodes.

10.3 Service Discovery Identities

Upon advancement of this document to a status of Draft, the XMPP Registrar shall add a category of 'filesys' to its registry of service discovery identities, with two associated types: 'directory' and 'file'.

10.4 Field Standardization

Field Standardization for Data Forms (XEP-0068) [10] defines a process for standardizing the fields used within Data Forms scoped by a particular namespace. This document reserves the FORM_TYPE "http://jabber.org/protocol/files" as well as specific fields for use within the context of that FORM_TYPE, as specified in the following registry submission.

```
<form_type>
  <name>http://jabber.org/protocol/files</name>
  <doc>XEP-01xx</doc>
  <desc>Service Discovery extension for file descriptions.</desc>
  <field var='date'
         type='text-single'
         label='Equivalent to date attribute from XEP-0096'/>
  <field var='description'
         type='text-single'
         label='Equivalent to desc element from XEP-0096'/>
  <field var='hash'
```

[9] 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/>.

[10] XEP-0068: Field Data Standardization for Data Forms <https://xmpp.org/extensions/xep-0068.html>.

```
        type='text-single'
        label='Equivalent to hash attribute from XEP-0096'/>
  <field var='mime-type'
        type='text-single'
        label='Equivalent to mime-type attribute from XEP-0095'/>
  <field var='size'
        type='text-single'
        label='Equivalent to size attribute from XEP-0096'/>
</form_type>
```

11 XML Schema

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

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='http://jabber.org/protocol/files'
    xmlns='http://jabber.org/protocol/files'
    elementFormDefault='qualified'>

  <xs:element name='retrieve'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='empty'>
          <xs:attribute name='node' type='xs:string' use='required'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```