XEP-0214: File Repository and Sharing

Nicholas Parker
mailto:nickbp@gmail.com
xmpp:nickp@jabber.org

2022-03-22
Version 0.3.1

Status	Type	Short Name
Deferred	Standards Track	NOT_YET_ASSIGNED

While a protocol has been described for initiating a file transfer from one user to another, there is not yet a defined way for users to designate a set of files as available for retrieval by other users of their choosing. This extension defines a common syntax for this purpose which is based on PubSub Collections.

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 Glossary **1**

4 Use Cases **2**

4.1 File Listing 2

4.1.1 Publication 2

4.1.2 Subscription 4

4.1.3 Addition 4

4.1.4 New Revisions 10

4.1.5 Modification/Deletion 11

4.2 File Requests 13

5 Implementation Notes **14**

6 Security Considerations **14**

7 IANA Considerations **14**

8 Jabber Registrar Considerations **14**

9 XML Schema **14**

1 Introduction

This document defines a common format which allows a Jabber user to submit, find, and retrieve files within XMPP. The file listing itself is stored on a Publish-Subscribe (XEP-0060) [1] server, allowing multiple users to manage the same listing. Other features specified by this extension include file metadata, revisions, and download mirrors.
Retrieval of files provided in a listing MAY be performed through any relevant protocol for transferring data (http, ftp, etc). This protocol standardizes the use of Publishing Stream Initiation Requests (XEP-0137) [2] to establish the file transfer, but also allows for usage of outside protocols such as http or ftp.

2 Requirements

The protocol defined herein provides the following functionality:

1. Publication of a list of available files to a PubSub server, with support for hierarchical listings, file metadata, user privileges, and file versioning.
2. Request that a file be sent from a Jabber contact directly to oneself using Stream Initiation.

3 Glossary

File Listing	A Root-level Pubsub Collection Node, containing information about files and/or subsections which a user or group of users have published.
Subsection	A Non-Root Collection Node which contains files and/or other subsections.
File	A Pubsub Node, stored within a File Listing, which describes all revisions of a given file. The filename and (optional) description are provided here.
Revision	A Pubsub Item which describes a given file revision. Other metadata which can vary between revisions, such as filesize, checksum, or available mirrors, is provided here.
Mirror	A location which has a given Revision available for download. Additional information about a Mirror MAY be specified in instances where the protocol requires it. A list of example file transfer protocols is provided below, but others may also be deployed.

[1] XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.
[2] XEP-0137: Publishing Stream Initiation Requests <https://xmpp.org/extensions/xep-0137.html>.

4 Use Cases

The following use cases describe tasks which are already covered by Publish-Subscribe (XEP-0060) [3] in a more generic context. These tasks are again being provided here in order to demonstrate the functionality provided by this protocol and convey the structure and syntax of the file listing. As a result of this close relationship, many details of PubSub are omitted here for brevity. Consult Publish-Subscribe (XEP-0060) [4] and PubSub Collection Nodes (XEP-0248) [5] for the full specification of node and user management commands as well as their server responses.

4.1 File Listing

4.1.1 Publication

Juliet wishes to make her sonnets available for retrieval by the public. She creates a Root Pubsub Collection Node which will contain her file listing:

Listing 1: Creating a New File Listing

```
<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='create3'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <create node='juliets_sonnets'/>
    <configure>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#node_config</value>
        </field>
        <field var='pubsub#node_type'><value>collection</value></field
          >
      </x>
    </configure>
  </pubsub>
</iq>

<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='config2'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub#owner'>
    <configure node='juliets_sonnets'/>
```

[3] XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.
[4] XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.
[5] XEP-0248: PubSub Collection Nodes <https://xmpp.org/extensions/xep-0248.html>.

```
      <x xmlns='jabber:x:data' type='submit'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#meta-data</value>
        </field>
        <field var='pubsub#title'><value>Juliet's␣Sonnets</value></
            field>
␣␣␣␣␣␣␣␣<field␣var='pubsub#description'><value>Optional␣Description</
    value></field>
␣␣␣␣␣␣</x>
␣␣␣␣</configure>
␣␣</pubsub>
</iq>
```

Juliet also wishes to add a subsection for her sonnets about Romeo. She creates another PubSub Collection Node under the Root Node:

Listing 2: Adding a Subsection to the Listing

```
<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='create3'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <create node='35227eec194a4f3971a5f3771e9c2271'/>
    <configure>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#node_config</value>
        </field>
        <field var='pubsub#collection'><value>juliets_sonnets</value><
            /field>
        <field var='pubsub#node_type'><value>collection</value></field
            >
      </x>
    </configure>
  </pubsub>
</iq>

<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='config2'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <configure node='35227eec194a4f3971a5f3771e9c2271'/>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#meta-data</value>
        </field>
```

```
        <field var='pubsub#title'><value>Sonnets About Romeo</value></
            field>
        <field var='pubsub#description'><value>Optional Description</
            value></field>
      </x>
    </configure>
  </pubsub>
</iq>
```

4.1.2 Subscription

Romeo wishes to view all of Juliet's shared sonnets. To do this, Romeo subscribes to the Root Collection Node:

Listing 3: Subscription to entire File Listing

```
<iq type='set'
    from='romeo@montague.net/orchard'
    to='pubsub.shakespeare.lit'
    id='collsub2'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <subscribe jid='romeo@montague.net' node='juliets_sonnets'/>
    <options>
      <x xmlns='jabber:x:data'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#subscribe_options</
              value>
        </field>
        <field var='pubsub#subscription_type'><value>items</value></
            field>
        <field var='pubsub#subscription_depth'><value>all</value></
            field>
      </x>
    </options>
  </pubsub>
</iq>
```

4.1.3 Addition

Juliet has just finished a new sonnet and wishes to announce its availability on her File Listing. She adds the sonnet as a new PubSub Node stored in her Collection Node, then inserts a first revision of her sonnet as an Item within that Node:

Listing 4: Adding a new File

```
<iq type='set'
```

```
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='create4'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub#node_config'>
    <create node='a6190c5d38e22452041d1c5798eff3f5'/>
    <configure>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='pubsub#collection'><value>juliets_sonnets</value><
           /field>
      </x>
    </configure>
  </pubsub>
</iq>

<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='config2'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <configure node='a6190c5d38e22452041d1c5798eff3f5'>
      <x xmlns='jabber:x:data' type='submit'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#meta-data</value>
        </field>
        <field var='pubsub#title'><value>sonnet.txt</value></field>
        <field var='pubsub#description'><value>Sonnet 42</value></
           field>
      </x>
    </configure>
  </pubsub>
</iq>

<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='publish1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <publish node='a6190c5d38e22452041d1c5798eff3f5'>
      <item id='1'>
        <entry xmlns='http://jabber.org/protocol/fileshare#
           item_metadata'>
          <size>5623</size>
          <modified>2006-12-13T18:30:02Z</modified>
          <checksum type="sha1">59282
             c5db190bdc3b152c5b38363442bfda8ebdd</checksum>
          <mime>text/plain</mime>
          <description>My Latest Sonnet!</description>
          <mirrors>
            <mirror type='sipub' description='File␣Transfer␣via␣
```

```
      capulet.com fileserver'>
    <sipub xmlns='http://jabber.org/protocol/sipub'
           from='fileserver@capulet.com'
           id='publish-sonnet.txt'
           mime-type='text/plain'
           profile='http://jabber.org/protocol/si/profile/
              file-transfer'>
      <file xmlns='http://jabber.org/protocol/si/profile/
         file-transfer'
            name='sonnet.txt'
            size='5623'/>
    </sipub>
  </mirror>
  <mirror type='sipub' description='Jingle HTTP File
     Transfer via capulet.com fileserver'>
    <sipub xmlns='http://jabber.org/protocol/sipub'
           from='fileserver-httpft@capulet.com'
           id='publish-sonnet.txt'
           mime-type='text/plain'
           profile='http://jabber.org/protocol/si/profile/
              jingle-httpft'>
      <description xmlns='http://www.xmpp.org/extensions/xep
         -XXXX.html#ns'>
        <manifest>
          <file>
            <name>sonnet.txt</name>
          </file>
        </manifest>
        <http>
          <url name='source-path'>/source/23A53F01/</url>
          <url name='preview-path'>/preview/90266EA1/</url>
        </http>
      </description>
    </sipub>
  </mirror>
  <mirror type='http' description='Shakespeare.lit Torrent'
          address='www.shakespeare.lit'
          ref='torrents/sonnet.torrent'/>
  <mirror type='http' description='Shakespeare.lit Website'
          address='www.shakespeare.lit'
          ref='~juliet/sonnet.txt'/>
  <mirror type='https' description='Shakespeare.lit Website
     (SSL)'
          address='ssl.shakespeare.lit'
          ref='~juliet/sonnet.txt'/>
  <mirror type='ftp' description='Shakespeare.lit FTP'
          user='guest' pass='guest'
          address='files.shakespeare.lit' port='21'
          ref='public/sonnet.txt'/>
```

```
            <mirror type='sftp' description='Shakespeare.lit SFTP'
                    user='guest' pass='guest'
                    address='ssh.shakespeare.lit' port='22'
                    ref='public/sonnet.txt'/>
            <mirror type='smb' description='Capulet Intranet SMB Share
                '
                    user='guest' pass='guest'
                    address='smbfiles.capulet.com'
                    ref='juliet/mysonnets/sonnet.txt'/>
          </mirrors>
        </entry>
      </item>
    </publish>
  </pubsub>
</iq>
```

The Item ID is set to 1, signifying the first revision for this file. Subsequent revisions/items will have incremented ID values, like one would see in a versioning system such as CVS or SVN. Implementations MAY follow this convention, but are not required to do so. For example, a given implementation may instead mark revisions using version numbers (”Beta 1”, ”6.2”, etc) or use other arbitrary strings. However, no two revisions of a given file may share the same ID.

Node IDs MAY take the form of ”path/to/file.ext”, rather than the randomized string ”a6190c5d38e22452041d1c5798eff3f5” provided in the above use case. For example, Juliet’s sonnet MAY instead use a Node ID of ”juliets_sonnets/sonnet.txt”, as long as this ID is unique within the PubSub server. Randomized strings are used in this document to illustrate that Node IDs SHOULD NOT be used for providing information about files.

Here is a listing of the possible metadata in a file revision (Item), each field is OPTIONAL:

Size	The size, in bytes, of the file.
Modified	The last modified time of the revision. Follows the format described in XMPP Date and Time Profiles (XEP-0082) XEP-0082: XMPP Date and Time Profiles <https://xmpp.org/extensions/xep-0082.html>.. If a publisher prefers to only make a single revision available to clients, the publisher MAY instead update this value (and others, such as size and/or checksum) to announce that a new version of the file is available.
Checksum	A checksum of the revision, using the specified hash algorithm. Acceptable types are ”sha512”, ”sha1”, ”md5”, and ”crc32”.
Mime	The file’s MIME type.
Description	Description text for the revision. As an example, could contain release notes.
Mirrors	A list of mirrors; their properties are defined below. If no downloads are available, MAY be left empty or removed entirely.

Because Romeo is now subscribed, he receives notice of Juliet's addition:

Listing 5: Notification of Addition

```
<message from='pubsub.shakespeare.lit' to='romeo@montague.net' id='
    create4'>
  <event xmlns='http://jabber.org/protocol/pubsub#event'>
    <collection>
      <node id='a6190c5d38e22452041d1c5798eff3f5'>
        <x xmlns='jabber:x:data' type='result'>
          <field var='FORM_TYPE' type='hidden'>
            <value>http://jabber.org/protocol/pubsub#node_config</
                value>
          </field>
          <field var='pubsub#collection'><value>juliets_sonnets</value
              ></field>
        </x>
      </node>
    </collection>
  </event>
</message>

<message from='pubsub.shakespeare.lit' to='romeo@montague.net' id='
    config2'>
  <event xmlns='http://jabber.org/protocol/pubsub#event'>
    <configuration node='a6190c5d38e22452041d1c5798eff3f5'>
      <x xmlns='jabber:x:data' type='result'>
        <field var='FORM_TYPE' type='hidden'>
          <value>http://jabber.org/protocol/pubsub#meta-data</value>
        </field>
        <field var='pubsub#description'><var>Sonnet 42</var></field>
        <field var='pubsub#title'><var>sonnet.txt</var></field>
      </x>
    </configuration>
  </event>
</message>

<message from='pubsub.shakespeare.lit' to='romeo@montague.net' id='foo
    '>
  <event xmlns='http://jabber.org/protocol/pubsub#event'>
    <items node='a6190c5d38e22452041d1c5798eff3f5'>
      <item id='1'>
        <entry xmlns='http://jabber.org/protocol/fileshare#item_config
            '>
          <size>5623</size>
          <modified>2006-12-13T18:30:02Z</modified>
          <checksum type="sha1">59282
              c5db190bdc3b152c5b38363442bfda8ebdd</checksum>
          <mime>text/plain</mime>
```

```
          <description>My Latest Sonnet!</description>
          <mirrors>
            ... MIRRORS ...
          </mirrors>
        </entry>
      </item>
    </items>
  </event>
</message>
```

The above examples give a listing of several possible file transfer protocols in example configurations. Only the sipub mirror type is REQUIRED; the other types are OPTIONAL. Here is a full listing of those protocols and their available settings:

Protocol	Description	Ref	Address	Port (default)	User	Pass
sipub (REQUIRED)	OPTIONAL	N/A	N/A	N/A	N/A	N/A
http (OPTIONAL)	OPTIONAL	REQUIRED	REQUIRED	OPTIONAL (80)	OPTIONAL	OPTIONAL
https (OPTIONAL)	OPTIONAL	REQUIRED	REQUIRED	OPTIONAL (443)	OPTIONAL	OPTIONAL
ftp (OPTIONAL)	OPTIONAL	REQUIRED	REQUIRED	OPTIONAL (21)	OPTIONAL	OPTIONAL
sftp (OPTIONAL)	OPTIONAL	REQUIRED	REQUIRED	OPTIONAL (22)	OPTIONAL	OPTIONAL
smb (OPTIONAL)	OPTIONAL	REQUIRED	REQUIRED	OPTIONAL (445)	OPTIONAL	OPTIONAL

The Description field is where an arbitrary description of the mirror MAY be placed. For example, if a File Listing is advertising mirrors which are located in different geographic locations, then this field may be used to specify those locations.
The Ref field is a unique address or identifier for retrieving the file from the mirror server. In the above examples, it is used as a path to the file.
The address and port fields describe the server where the file may be retrieved using the specified protocol. If a port is not provided, the default value (specified in parentheses) MAY be assumed.
The User and Pass fields are for providing credentials which, if given by the File Listing, SHOULD be used when requesting the file. For example, an sftp mirror MAY require that the user log in using specified credentials before the file may be retrieved.

4.1.4 New Revisions

Juliet has revised her sonnet and wishes to publish the new version, while still leaving the original copy available for retrieval. To do this, she inserts a new Item, representing her new revision, into the file's Node:

Listing 6: Adding a new Revision

```
<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='publish1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <publish node='a6190c5d38e22452041d1c5798eff3f5'>
      <item id='2'>
        <entry xmlns='http://jabber.org/protocol/fileshare#item_config
            '>
          <size>6102</size>
          <modified>2007-01-13T18:30:02Z</modified>
          <checksum type="md5">6aaa20212a99548765b3b15f24f19aaa</
              checksum>
          <checksum type="sha1">97
              cbc0e445435af94db5cc2133b94ab5faf1399a</checksum>
          <mime>text/plain</mime>
          <description>A revised copy, fixed some spelling errors.</
              description>
          <mirrors>
            <mirror type='ftp' description='Shakespeare.lit␣FTP'
                    user='guest' pass='guest'
                    address='files.shakespeare.lit' port='21'
                    ref='public/juliet/sonnet2.txt'/>
            <mirror type='http' description='Shakespeare.lit␣Website'
                    address='www.shakespeare.lit'
                    ref='~juliet/sonnet2.txt'/>
            <mirror type='sipub' description='File␣Transfer␣via␣
                capulet.com␣fileserver'>
              <sipub xmlns='http://jabber.org/protocol/sipub'
                     from='fileserver@capulet.com'
                     id='publish-sonnet2.txt'
                     mime-type='text/plain'
                     profile='http://jabber.org/protocol/si/profile/
                         file-transfer'>
                <file xmlns='http://jabber.org/protocol/si/profile/
                    file-transfer'
                      name='sonnet2.txt'
                      size='6102'/>
              </sipub>
            </mirror>
          </mirrors>
```

```
        </entry>
      </item>
    </publish>
  </pubsub>
</iq>
```

4.1.5 Modification/Deletion

Juliet has uploaded a copy of her revised sonnet to a new mirror, and wishes to let her subscribers know about this secondary source. She is able to do this by modifying the revision in question to include a reference to her website, overwriting the existing mirrors in the Item with an updated list:

Listing 7: Modifying a Revision

```
<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='publish1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <publish node='a6190c5d38e22452041d1c5798eff3f5'>
      <item id='2'>
        <entry xmlns='http://jabber.org/protocol/fileshare#item_config
            '>
          <mirrors>
            <mirror type='ftp' description='Shakespeare.lit␣FTP'
                    user='guest' pass='guest'
                    address='files.shakespeare.lit' port='21'
                    ref='public/juliet/sonnet2.txt'/>
            <mirror type='http' description='Shakespeare.lit␣Website'
                    address='www.shakespeare.lit'
                    ref='~juliet/sonnet2.txt'/>
            <mirror type='sipub' description='File␣Transfer␣via␣
                capulet.com␣fileserver'>
              <sipub xmlns='http://jabber.org/protocol/sipub'
                     from='fileserver@capulet.com'
                     id='publish-sonnet2.txt'
                     mime-type='text/plain'
                     profile='http://jabber.org/protocol/si/profile/
                         file-transfer'>
                <file xmlns='http://jabber.org/protocol/si/profile/
                    file-transfer'
                      name='sonnet2.txt'
                      size='6102'/>
              </sipub>
            </mirror>
            <mirror type='http' description='Shakespeare.lit␣Boston␣
                Mirror'
```

```
                    address='www.capulet.com'
                    ref='~juliet/sonnet2.txt'/>
          </mirrors>
        </entry>
      </item>
    </publish>
  </pubsub>
</iq>
```

Juliet now wishes to allow others to contribute to her sonnet collection. She gives owner access for the entire Listing to Romeo, and publisher access to her nurse:

Listing 8: Modifying Users

```
<iq type='set'
    from='juliet@capulet.com/balcony'
    to='pubsub.shakespeare.lit'
    id='ent3'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub#owner'>
    <affiliations node='juliets_sonnets'>
      <affiliation jid='nurse@capulet.com' affiliation='publisher'/>
      <affiliation jid='romeo@montague.net' affiliation='owner'/>
    </affiliations>
  </pubsub>
</iq>
```

Romeo uses his owner access to remove the older revision of Juliet's sonnet:

Listing 9: Deleting a Revision

```
<iq type='set'
    from='romeo@montague.net/orchard'
    to='pubsub.shakespeare.lit'
    id='retract1'>
  <pubsub xmlns='http://jabber.org/protocol/pubsub'>
    <retract node='a6190c5d38e22452041d1c5798eff3f5'>
      <item id='1'/>
    </retract>
  </pubsub>
</iq>
```

Other deletion, modification, and user management operations are available as described in Publish-Subscribe (XEP-0060) [6] and PubSub Collection Nodes (XEP-0248) [7].

[6] XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.
[7] XEP-0248: PubSub Collection Nodes <https://xmpp.org/extensions/xep-0248.html>.

4.2 File Requests

Romeo is interested in seeing what files Juliet has made available. To do this, Romeo sends a request to Juliet for repositories which she is associated with:

Listing 10: Request for File Repository listing

```
<iq type='get'
    from='romeo@montague.net/orchard'
    to='juliet@capulet.com'
    id='repolistreq'>
  <fileshare xmlns='http://jabber.org/protocol/si/profile/fileshare'>
    <list/>
  </fileshare>
</iq>
```

Juliet responds with a list of PubSub nodes where she has published files or which she believes would be interesting to Romeo. If no such locations exist, Juliet SHOULD respond with an empty list.

Listing 11: File Repository listing

```
<iq type='get'
    from='romeo@montague.net/orchard'
    to='juliet@capulet.com'
    id='repolist'>
  <fileshare xmlns='http://jabber.org/protocol/si/profile/fileshare'>
    <list>
      <repo address='pubsub.shakespeare.lit'
            node='juliets_sonnets' description='My Sonnets'/>
    </list>
  </fileshare>
</iq>
```

After browsing Juliet's repository, Romeo has chosen to download her sonnet. The most recent revision of this file contains a listing of available mirrors, and Romeo sees that one of them is an SI stream. Romeo sends an SI request to that mirror:

Listing 12: Request that a file be sent

```
<iq type='get'
    id='sipub-request-0'
    from='romeo@montague.net/orchard'
    to='fileserver@capulet.com'>
  <start xmlns='http://jabber.org/protocol/sipub'
         id='publish-sonnet2.txt'/>
</iq>
```

The rest of the negotiation and file transfer occurs as described in Publishing Stream Initiation Requests (XEP-0137) [8].

5 Implementation Notes

Since PubSub is used for the File Listing, the access models described in Publish-Subscribe (XEP-0060) [9] and PubSub Collection Nodes (XEP-0248) [10] MUST be followed. Users MUST NOT be able to view or control information in the File Listing to which they do not have access. If user access to files is restricted, the Mirror servers and the PubSub server MUST be able to synchronize these restrictions between them. See Security Considerations.

6 Security Considerations

When restricted files are being distributed, mirrors need to know which users have sufficient privileges to access which files. If mirrors are not kept up to date on user privileges, unauthorized users could access files directly from those mirrors, thus bypassing any restrictions being set on the PubSub server.

7 IANA Considerations

No interaction with the Internet Assigned Numbers Authority (IANA) is required as a result of this XEP.

8 Jabber Registrar Considerations

TODO

9 XML Schema

TODO

[8] XEP-0137: Publishing Stream Initiation Requests <https://xmpp.org/extensions/xep-0137.html>.
[9] XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.
[10] XEP-0248: PubSub Collection Nodes <https://xmpp.org/extensions/xep-0248.html>.