XEP-0015: Account Transfer

Casey Crabb
mailto:crabbkw@nafai.dyndns.org
xmpp:airog@floobin.cx

2002-04-18
Version 0.4

Status	Type	Short Name
Rejected	Standards Track	

A proposal for enabling the ability to transfer an account from one Jabber server to another.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

I have found the need in the past to migrate an account from one server to another for various reasons. Many of the people who ask me about Jabber ask if there is a way to migrate their account from one server to another if the need arises. There is no reason Jabber can not handle this internally and update all the JID-references appropriately.
Jabber servers come and go, especially ones run by people who are just playing with the technology. Computers also die and funding runs out. It can be hard on users to have to re-create their rosters every time they have to change to a different server. Administrators also want to provide an 'out' for their users, so that they feel more secure in the time spent setting up rosters. For these reasons there should be a way to migrate an account from one server to another.

2 Main Body

A basic overview of the behavior would be as follows.

2.1 Preconditions

- Bob has a jabber account on a server, floobin.cx specifically: olduser@floobin.cx.
- Bob knows that the floobin.cx server is going to go down soon and therefore needs to find a new jabber server.
- He realizes that jabber.org offers accounts to anyone, so he sets out to migrate his account to jabber.org.
- Bob does not yet have an account on jabber.org.

2.2 Order of events

Throughout most of the account transfer the server hosting the old account will be acting for the user. The end client should have very little to do with the actual transfer.

1. Bob creates an account on Jabber.org: bobsnewaccount@jabber.org.
2. Bob logs into both bobsnewaccount@jabber.org and olduser@floobin.cx.
3. From olduser@floobin.cx he sends the 'I want to migrate my account' packet to Floobin.cx. This packet includes the JID to migrate to.
4. Floobin.cx sends the 'user wants to migrate account to you' packet to bobsnewaccount@jabber.org. This packet contains the JID of the old account.

5. bobsnewsaccount@jabber.org receives the 'user wants to migrate account to you' packet and authorizes the transfer.

6. Once the migration is authorized the floobin.cx server will start the migration process. The first part is to notify each person subscribed to olduser@floobin.cx's presence that the account has moved to bobsnewaccount@jabber.org.

7. Once that is complete the roster itself must be added into bobsnewaccount@jabber.cx's roster. There are many issues with that remain and should be dealt with in the future. See the section on scope for more information.

8. finally floobin.cx informs olduser@floobin.cx that the transfer was completed.

2.3 Protocol Example

Listing 1: User initiates process

```
<iq id='initacctxfer' to='floobin.cx' from='airog@floobin.cx' type='
    ask'>
          <query xmlns='jabber:iq:accountxfer'>
            <oldaccount>airog@jabber.org</oldaccount>
            <newaccount>newaccount@jabber.org</newaccount>
          </query>
        </iq>
```

Listing 2: Server asks for permission from newaccount@jabber.org

```
<iq id='acctxfer1' type='ask' from='floobin.cx' to='newaccount@jabber.
    org'>
          <query xmlns='jabber:iq:accountxfer'>
            <oldaccount>airog@jabber.org</oldaccount>
          </query>
        </iq>
```

Listing 3: XML received at user's new account

```
<iq id='acctxfer1' type='ask' from='floobin.cx' to='newaccount@jabber.
    org'>
          <query xmlns='jabber:iq:accountxfer'>
            <oldaccount>airog@jabber.org</oldaccount>
          </query>
        </iq>
```

Listing 4: newaccount@jabber.org accepts the migration

```
<iq id='acctxfer1' type='result' to='floobin.cx' from='
    newaccount@jabber.org'>
          <query xmlns='jabber:iq:accountxfer'>
            <allowed/>
```

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

On acceptance the server on which the old account resides starts the migration process by sending this to each person subscribed to the user's presence.

Listing 5: XML sent to each JID subscribed to airog@floobin.cx's presence

```
<iq id='acctxferss1' type='set' from='floobin.cx' to='jabber.org'>
        <query xmlns='jabber:iq:accountxfer'>
          <oldaccount>airog@jabber.org</oldaccount>
          <newaccount>newaccount@jabber.org</newaccount>
          <rosteritem jid='frienduser@jabber.org'/>
        </query>
      </iq>
```

Listing 6: The server hosting the account of the roster item responds

```
<iq id='acctxferss1' type='result' to='floobin.cx' from='jabber.org'/>
```

Once that update has been sent to all the contacts on the roster the floobin.cx server sends to the jabber.org server airog@floobin.cx's roster as follows:

Listing 7: airog@floobin.cx's roster is transferred into newuser@jabber.org's roster

```
<iq type='set' id='acctxferss2' from='floobin.cx' to='jabber.org'>
        <query xmlns='jabber:iq:accountxfer'>
          <oldaccount>airog@jabber.org</oldaccount>
          <newaccount>newaccount@jabber.org</newaccount>
          <item jid='frienduser@jabber.org' name='friend1'
              subscription='both'/>
          <item jid='annoyuser@jabber.org' ask='subscribe'/>
          <item jid='someone@jabber.org' subscription='from'/>
        </query>
      </iq>
```

Listing 8: floobin.cx responds saying the roster transfer was successful

```
<iq type='result' id='acctxferss2' to='floobin.cx' from='jabber.org'/>
```

Once the migration finishes a notification is sent to the user:

Listing 9: Process completed

```
<iq id='initacctxfer' from='floobin.cx' to='airog@floobin.cx' type='
   result'/>
```

3 Open Issues, Concerns and Scoping

3.1 How do we handle transferring transport stuff?

Because we cannot determine easily if the new server will support the same transports as the old server we cannot easily transfer entities that pass through the transport. Therefore, until jabber:iq:browse matures, or some other solution for determining if two transports support the same functionality we should not attempt to migrate transport information.
I propose the following algorithm for determining if a particular roster item is a sub-item of a transport. There are jabber roster items for each of the transports themselves, something to the effect of icq.jabber.org or aim.jabber.org. They contain no user portion of the jid. We record all of these in a list that we will call the 'transport-list'. Then for each roster item we want to migrate we compare its 'host' part of the jid to all items in the 'transport-list'. If the roster item matches, then the roster item is a hosted through the transport and shouldn't be migrated.

3.2 Empty Pointer Accounts

Does the server keep an empty account that redirects requests to the new account? I've been hearing mass rumblings of 'NO' here.

3.3 vCards and private storage data

How do we handle vCard information or server side stored preferences? Since the account we're migrating to can be any account some of that information might already be there, how do we resolve conflicts?
Also, we cannot be sure that the new server supports storage of private data. This again needs some sort of features negotiation, discovery which could be provided by jabber:iq:browse.
Until jabber:iq:browse is in the 'standards' stage, I recommend we only transfer regular jabber users, and not transfer anything but the roster. All the client software will have to set their preferences for themselves on the new server.