XEP-0283: Moved

Tory Patnoe
mailto:tpatnoe@cisco.com
xmpp:tpatnoe@cisco.com

2018-08-06
Version 0.1.1

Status	Type	Short Name
Deferred	Standards Track	moved

This document defines an XMPP protocol extension that enables a user to inform its contacts about a change in JID.

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

There are a variety of reasons why a user may wish to change their JID. For example, a surname change because of marriage or simply an easier JID to remember.
This XEP defines an approach for communicating that your JID has moved to a new JID, extending the existing subscription protocol documented in XMPP IM [1]. The steps outlined here may be done either through a client or automated by a server.

2 Requirements

- The methods described here maintain compatibility with XMPP Core [2] and RFC 6121.

3 The Move

In this scenario user@example.com moves to user2@example2.com. Both the user@example.com and user2@example2.com accounts have been created and still exist. The roster for user2@example2.com is empty and the user wants to populate it with their entries from user@example.com.

original JID user@example.com

new JID user2@example2.com

3.1 Unsubscribing the original JID from outbound subscriptions

Because the original JID is no longer going to be used, the user SHOULD unsubscribe from all the outbound subscriptions user@example.com had. These can be identified as those in the 'to' or 'ask' states as defined in the 'jabber:iq:roster' protocol in RFC 6121.
To unsubscribe all outbound subscriptions for the original JID send an unsubscribe <presence/> stanza to all the old contacts with a <moved/> element containing the new JID.
There is the potential for other users to send a malicious unsubscribe containing a spoofed <moved/> JID. Therefore, clients SHOULD NOT automatically subscribe to the JID contained in the <moved/> stanza when receiving a subscribe <presence/> stanza without displaying the <moved/> JID to the user. See the Security Considerations section for details.

Listing 1: Client unsubscribes outbound subscriptions from original JID

```
<presence from='user@example.com' to='contact@example.com' type='
    unsubscribe'>
```

[1] RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence <http://tools.ietf.org/html/rfc6121>.
[2] RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc6120>.

```
    <status>I've changed JIDs from user@example.com to user2@example2.
        com</status>
    <moved xmlns='urn:xmpp:moved:0' new='user2@example2.com'/>
</presence>
```

3.2 Unsubscribing the original JID from inbound subscriptions

Because the original JID is no longer going to be used, the user SHOULD unsubscribe from all contacts the user@example.com had an inbound subscription from. These can be identified as those in the 'from' subscription state as defined in in the 'jabber:iq:roster' protocol in RFC 6121.
To unsubscribe all inbound subscriptions send an unsubscribed <presence/> stanza to all the old contacts with a <moved/> element containing the new JID.
There is the potential for other users to send a malicious unsubscribed containing a spoofed <moved/> JID. Therefore, clients SHOULD NOT automatically subscribe to the JID contained in the <moved/> stanza without displaying the <moved/> JID to the user. See the Security Considerations section for details.

Listing 2: Client unsubscribes inbound subscriptions from original JID

```
<presence from='user@example.com' to='contact@example.com' type='
    unsubscribed'>
    <status>I've changed JIDs from user@example.com to user2@example2.
        com</status>
    <moved xmlns='urn:xmpp:moved:0' new='user2@example2.com'/>
</presence>
```

3.3 Subscribing the new JID to all your existing contacts

Once the new JID has been created on a server it is possible for the new JID to subscribe to the contacts they had on the original JID's roster. This is done by sending a new subscription request with a <moved/> element containing the new JID.
The new subscription MUST come from the new JID's server.
There is the potential for other users to send a malicious subscribe request and spoof the content of the <moved/> element identifying an original JID. Therefore, clients SHOULD NOT automatically unsubscribe an existing roster entry if is listed as the target in the <moved/> element when a subscribe is received. See the Security Consideration section for details.
Clients accepting the moved subscription SHOULD indicate to the user that that this subscription request was the result of a move operation and because of potential malicious behavior SHOULD NOT auto-accept the subscription without displaying the <moved/> JID to the user.

Listing 3: Client requests subscription from new JID

```
<presence from='user2@example2.com' to='contact@example.com' type='
    subscribe'>
       <status>I've␣changed␣JIDs␣from␣user@example.com␣to␣
           user2@example2.com</status>
␣␣␣␣␣␣<moved␣xmlns='urn:xmpp:moved:0'␣old='user@example.com'/>
␣␣</presence>
```

3.4 Contacts Offline at the Time the Rename Occurs

RFC 6120 clarifies that an incoming subscribe <presence/> stanza MUST be preserved by the server and <presence/> stanzas of type unsubscribe and unsubscribed are not preserved on the server. Therefore, for a contact who is offline, their servers MAY have automatically removed the original roster entry when seeing the unsubscribe and unsubscribed stanzas. At the time of writing this XEP, NOT saving and forwarding the presence stanzas will be the default behavior of most servers.
What this means is that a contact coming online after the rename outlined above MAY only see the <presence/> of type 'subscribe' with the <moved/> element. Clients should be aware of this behavior.

3.5 Presence Ordering

In following the principle of least surprise, it is considered good practice to send the subscribe stanza after the unsubscribe and unsubscribed stanzas.

3.6 Preservation of Groups

One of the side effects of this scheme is the potential for a contact to lose the groups to which it had organized the original JID. Clients aware of the <moved/> element can mitigate this with the following rules.

- If the contacts client receives an unsubscribed with a <moved/> element, remove the subscription but initiate a roster push for the original JID with the subscription set to 'none'. When the new subscription is received the new JID MAY be placed into the roster in the same groups as the original JID and the original JID can then be removed from the roster.
- If a subscribe is received with a <moved/> element, the client MAY automatically place the new JID into the same groups as the original JID.

As discussed in 'Contacts Offline at the Time the Rename Occurs', a server MAY automatically handle the unsubscribe and unsubscribed stanzas. If this occurs it will be impossible to preserve the original groups.

3.7 One-way subcriptions and full roster state

3.7.1 One-way Inbound subscription

If the original JID, user@example.com, had only an inbound subscription (from or pending in), then the contact will only receive an unsubscribed <presence/> stanza. The contact's client, knowing the state of the subscription (which is 'to' or 'none' with 'ask='subscribe' from the contact's perspective), at that point MAY choose to prompt the user to subscribe to the new JID listed in the <moved/> element.
Because of the ability to spoof the <moved/> element, the client SHOULD NOT automatically subscribe to the <moved/> element target, but SHOULD present the new JID to the contact before sending out a subscription request.

3.7.2 One-way Outbound subscription

If the original JID, user@example.com, had only an outbound subscription (to or ask), then the contact SHOULD only receive an unsubscribe <presence/> stanza. The contact's client, knowing the state of the subscription (which is 'from' from the contact's perspective), at that point MAY choose to prompt the user to subscribe to the new JID listed in the <moved/> element.
Because of the ability to spoof the <moved/> element, the client SHOULD NOT automatically subscribe to the <moved/> element target.

3.7.3 Roster state and action table

Server state	Client state (jabber:iq:roster)	Send unsubscribe from original JID	Send unsubscribed from original JID	Send subscribe from new JID
none	none			
none + pending out	none + ask='subscribe'	yes		yes
none + pending in	n/a		yes - server only	
none + pending in/out	none + ask='subscribe'	yes	yes - server only	yes
to	to	yes		yes
to + pending in	to	yes	yes - server only	yes
from	from		yes	
from + pending out	from/none + ask='subscribe'	yes	yes	yes
both	both	yes	yes	yes

4 Security Considerations

It is not intended for servers to strip any <moved/> elements from <presence/> stanzas sent in from a client. This allows clients as well as servers to implement these same procedures.
In order to prevent other users from maliciously altering contacts the client SHOULD NOT automatically subscribe to a <moved/> JID when it receives an unsubscribe and SHOULD NOT automatically unsubscribe to a <moved/> JID when it receives a subscribe.
The following illustrates an example malicious attack.

1. userA@example.com subscribes to userB@example.com
2. The userB@example.com automatically accepts the subscription from userA@example.com. (Automatically done by the client using a simple domain trust).
3. userA@example.com unsubscribes with the <moved/> 'new' JID set to companyCEO@example.com.
4. The previous steps can be repeated and have userB@example.com subscribe to a large number of people.

A similar attack can be done with a new subscribe request causing users by guessing which users are subscribed to a contact.

1. hacker@example.com subscribes to userB@example.com guessing that userA@example.com is on userB's roster.

   <presence from='hacker@example.com' to='userB@example.com' type='subscribe'> <status>Subscribe to me!</status> <moved xmlns='urn:xmpp:moved:0' old='userA@example.com'/> </presence>

2. If userB's client automatically accepted the subscription without displaying at prompt to userB showing the new JID to be hacker@example.com, then the user has no idea that hacker@example.com was just added to the roster.

5 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [3].

[3] 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/>.

6 XMPP Registrar Considerations

6.1 Protocol Namespaces

This specification defines the following XML namespace:

- urn:xmpp:moved:0

Upon advancement of this specification from a status of Experimental to a status of Draft, the XMPP Registrar [4] shall add the foregoing namespace to the registry located at <https://xmpp.org/registrar/namespaces.html>, as described in Section 4 of XMPP Registrar Function (XEP-0053) [5].

6.2 Protocol Versioning

If the protocol defined in this specification undergoes a revision that is not fully backwards-compatible with an older version, the XMPP Registrar shall increment the protocol version number found at the end of the XML namespaces defined herein, as described in Section 4 of XEP-0053.

7 XML Schema

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

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:xmpp:moved:0'
    xmlns='urn:xmpp:moved:0'
    elementFormDefault='qualified'>

  <xs:element name='moved'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:string'>
          <xs:attribute name='new' type='xs:string' use='optional'/>
          <xs:attribute name='old' type='xs:string' use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
```

[4] 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/>.

[5] XEP-0053: XMPP Registrar Function <https://xmpp.org/extensions/xep-0053.html>.

```
  </xs:element>

</xs:schema>
```

8 Acknowledgements

The author wishes to thank Doug Abbink, Mikhail Belov, Peter Saint-Andre, and Peter Sheu for their feedback.