XEP-0259: Message Mine-ing

Joe Hildebrand
mailto:jhildebr@cisco.com
xmpp:hildjj@jabber.org

2009-01-21
Version 0.1

Status	Type	Short Name
Deferred	Standards Track	mine

In servers that deliver messages intended for the bare JID to all resources, the resource that claims a conversation notifies all of the other resources of that claim.

Legal

Copyright

This XMPP Extension Protocol is copyright © 1999 – 2017 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

At the time of original writing of this XEP, many XMPP servers handle message stanzas sent to a user@host (or "bare") JID with no resource by delivering that message only to the resource with the highest priority for the target user. Some server implementations, however, have chosen to send these messages to all of the online resources for the target user. If the target user is online with multiple resources when the orginal message is sent, a conversation ensues on one of the user's devices; if the user subsequently switches devices, parts of the conversation may end up on the alternate device, causing the user to be confused, misled, or annoyed.
This XEP proposes an approach for cleaning up the leftover conversation shards on alternate devices, paving the way for servers to deliver messages to multiple devices. As the basic approach, the receiving server asks all of the resources of a user "whose message is this?". The first resource to say "mine!" wins.

2 Requirements

- Large changes SHOULD NOT be required to existing servers
- Clients that do not implement the new protocol MUST be able participate in conversations
- All messages MUST NOT be delivered to all devices at all times, due to scale concerns
- Clients that do not own the message MUST be notified when a different device claims ownership of the message
- Multiple clients MUST be able to uambiguosly decide which of them owns a given message.

3 Use Cases

3.1 Determining Support: Servers

If a server implements the Mine capability, it MUST specify the 'urn:xmpp:tmp:mine:0' feature in its service discovery information features as specified in Entity Capabilities (XEP-0115) [1] or Service Discovery (XEP-0030) [2]. Clients MUST NOT send ownership changes if their server does not support this feature.

Listing 1: Client requests information about its own server

```
<iq type='get'
```

[1] XEP-0115: Entity Capabilities <https://xmpp.org/extensions/xep-0115.html>.
[2] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.

```
    from='romeo@montague.net/orchard'
    id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 2: Server responds with mine feature

```
<iq type='get'
    to='romeo@montague.net/home'
    from='montague.net'
    id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
...
    <feature var='urn:xmpp:tmp:mine:0'/>
...
  </query>
</iq>
```

3.2 Determining Support: Clients

Clients that support this protocol MUST support XEP-0115, and MUST add the 'urn:xmpp:tmp:mine:0' feature to their entity capabilities, in order to allow for potential server optimizations.

Listing 3: Romeo publishes his capabilities

```
<presence from='romeo@example.net/home'>
  <c xmlns='http://jabber.org/protocol/caps'
     hash='sha-1'
     node='http://example.com/clients/Mine'
     ver='j+5eLRCz6NP6IEPob80JB6sWR3Y='/>
</presence>
```

Listing 4: Romeo responds to capabilities inquiry from his server

```
<iq from='romeo@example.net/home'
    id='disco1'
    to='example.net'
    type='result'>
  <query xmlns='http://jabber.org/protocol/disco#info'
         node='http://example.com/clients/Mine#/
             WmLAKHhB87dOqn5NUgxrr5NbfE='>
    <identity category='client' type='pc' name='Mine'/>
    <feature var='urn:xmpp:tmp:mine:0'/>
  </query>
</iq>
```

3.3 Receving a message to the bare JID

When a server that implements the Mine capability receives a message addressed to a user's bare JID, it MUST:

- Ensure that no "whose" element is already on the message. See the Errors section for processing.
- Add a whose element to the message, containing an id attribute with a new value
- Ensure that the the same value of the "id" attribute is never sent to the same session

Messages that have been processed to include a valid "whose" element, are now also considered an "ownership request"

Listing 5: Juliet sends Romeo an undirected message

```
<message
    from='juliet@example.com/balcony'
    to='romeo@example.net'
    type='chat'>
  <body>Wherefore art thou, Romeo?</body>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
</message>
```

Listing 6: The ownership request, before broadcasting

```
<message
    from='juliet@example.com/balcony'
    to='romeo@example.net'
    type='chat'>
  <body>Wherefore art thou, Romeo?</body>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
  <whose xmlns='urn:xmpp:tmp:mine:0' id='4'/>
</message>
```

3.4 Broadcasting ownership requests

The receiving server MUST send a copy of the ownership request to each of that user's non-negative priority resources. Each copy of the message MUST contain a whose element, each of which has the same id attribute.

Listing 7: Romeo's server forwards copies of the message to all of his resources

```
<message
    from='juliet@example.com/balcony'
    to='romeo@example.net/home'
```

```
    type='chat'>
  <body>Wherefore art thou, Romeo?</body>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
  <whose xmlns='urn:xmpp:tmp:mine:0' id='4'/>
</message>

<message
    from='juliet@example.com/balcony'
    to='romeo@example.net/work'
    type='chat'>
  <body>Wherefore art thou, Romeo?</body>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
  <whose xmlns='urn:xmpp:tmp:mine:0' id='4'/>
</message>

<message
    from='juliet@example.com/balcony'
    to='romeo@example.net/mobile'
    type='chat'>
  <body>Wherefore art thou, Romeo?</body>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
  <whose xmlns='urn:xmpp:tmp:mine:0' id='4'/>
</message>
```

3.5 Claiming ownership

When one client for a receiving user detects that the user's attention has been directed to a given message, that client MUST send an ownership claim (mine!) to the bare JID of the receiving user. If there was a thread element in the original message, it MUST be included in the acceptance notification. There MUST NOT be a body element in the message, and the message SHOULD use the same message type as the ownership request. The mine element MUST include an id element for each of the messages that the client wants to accept. The mine element MUST include at least one id.

Listing 8: Romeo's "work" client claims ownership

```
<message
    to='romeo@example.net'
    from='romeo@example.net/work'
    type='chat'>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
  <mine xmlns='urn:xmpp:tmp:mine:0'>
    <id>4</id>
  </mine>
</message>
```

3.6 Notification of ownership claim

As with all messages sent to a bare JID at a server implementing the Mine feature, the acceptance message MUST be forwarded to all of the non-negative priority resources.

Listing 9: Each of Romeo's clients receives the claim

```
<message
    to='romeo@example.net/home'
    from='romeo@example.net/work'
    type='chat'>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
  <mine xmlns='urn:xmpp:tmp:mine:0'>
    <id>4</id>
  </mine>
</message>

<message
    to='romeo@example.net/work'
    from='romeo@example.net/work'
    type='chat'>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
  <mine xmlns='urn:xmpp:tmp:mine:0'>
    <id>4</id>
  </mine>
</message>

<message
    to='romeo@example.net/mobile'
    from='romeo@example.net/work'
    type='chat'>
  <thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
  <mine xmlns='urn:xmpp:tmp:mine:0'>
    <id>4</id>
  </mine>
</message>
```

3.7 Claim processing

When a client receives an ownership claim that was sent from that client for an ID that has not been previously claimed, the client MUST note that the message associated with the ID has been confirmed, and ignore any further ownership claims for that ID.
When a client receives an ownership claim that was sent from a different client of the same user for a ID that has not been previously received, the client MUST note that the message associated with the ID has been retracted, and ignore any further ownership claims for that ID. Retracted messages SHOULD be removed from the client's user interface, or otherwise marked in some way as retracted.

Clients MUST ignore ownership claims for IDs for which they have no corresponding message. Assuming that messages are delivered and processed in order, these rules should ensure that exactly one client resource has a confirmed copy of the message

3.8 Claims for Multi-User Chat rooms

The same approach that has been described for one-to-one messages above can also be used by Multi-User Chat (XEP-0045) [3] (MUC) rooms. Rooms that want to participate MUST send the 'urn:xmpp:tmp:mine:0' feature in the room's disco info. The room MUST then perform the role of the server in the above descriptions, ensuring that unique ID's are assigned to all outbound groupchat messages that were addressed to the bare JID of the room. Ownership claims MUST be sent to the bare JID of the **room**, not the receiving user.
This capability might be used to distribute questions to multiple experts in a room, such that a single expert answers a question.

Listing 10: Message is sent to the room

```
<message
    from='hag66@shakespeare.lit/pda'
    to='darkcave@chat.shakespeare.lit'
    type='groupchat'>
  <body>Harpier cries: 'tis␣time,␣'tis time.</body>
</message>
```

Listing 11: Room forwards message to all participants as ownership request

```
<message
    from='darkcave@chat.shakespeare.lit/thirdwitch'
    to='crone1@shakespeare.lit/desktop'
    type='groupchat'>
  <body>Harpier cries: 'tis␣time,␣'tis time.</body>
  <whose xmlns='urn:xmpp:tmp:mine:0' id='5'/>
</message>

<message
    from='darkcave@chat.shakespeare.lit/thirdwitch'
    to='wiccarocks@shakespeare.lit/laptop'
    type='groupchat'>
  <body>Harpier cries: 'tis␣time,␣'tis time.</body>
  <whose xmlns='urn:xmpp:tmp:mine:0' id='5'/>
</message>

<message
    from='darkcave@chat.shakespeare.lit/thirdwitch'
    to='hag66@shakespeare.lit/pda'
```

[3] XEP-0045: Multi-User Chat <https://xmpp.org/extensions/xep-0045.html>.

```
    type='groupchat'>
  <body>Harpier cries: 'tis␣time,␣'tis time.</body>
  <whose xmlns='urn:xmpp:tmp:mine:0' id='5'/>
</message>
```

Listing 12: A participant claims ownership

```
<message
    to='darkcave@chat.shakespeare.lit'
    from='crone1@shakespeare.lit/desktop'
    type='groupchat'>
  <mine xmlns='urn:xmpp:tmp:mine:0'>
    <id>5</id>
  </mine>
</message>
```

4 Error Cases

4.1 Invalid "whose"

A server receives a message addressed to the bare JID of a user, from a different user than the one in the to address, containing a "whose" or "mine" element, it MUST NOT forward the message on to any clients. This case is always either an attack, a misconfiguration, or the result of bad code. If the user in the from address is already known to the user in the to address (for example, to user in the to address has a presence subscription to the user in the from address), the server MAY send back a helpful "bad-request" error.

Listing 13: Romeo responds to a bad request from his friend Juliet

```
<message
    to='juliet@example.com/balcony'
    from='romeo@example.net'
    type='error>
␣␣<thread>0e3141cd80894871a68e6fe6b1ec56fa</thread>
␣␣<body>My␣client␣runneth␣over</body>
␣␣<whose␣xmlns='urn:xmpp:tmp:mine:0'␣id='4'>
␣␣<error␣type='modify'>
␣␣␣␣<bad-request␣xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
␣␣␣␣<text>Yours</text>
␣␣</error>
</message>
```

However, if the user in the from address is not known to the user in the to address, or the server perfers not to send helpful errors, the server MUST treat the message as if it was addressed to an unknown user. Otherwise, sending a message with an invalid "whose" or "mine" could allow an attacker to probe for valid users at a site.

5 Business Rules

5.1 Generating IDs

The value of the id attribute sent by servers MUST be valid output from the NODEPREP profile of stringprep.

5.2 ID Semantics

The value of the id resource is completely opqaque; receiving clients MUST NOT use any apparent order or semantic in the value of the id to perform optimizations or business logic.

5.3 Comparing IDs

Clients MUST only compare the value of ID's for equality, never for order. ID's MUST be compared for equality octet-for-octet or codepoint-for-codepoint; a basic string comparison with no extra canonicalization.

5.4 Accepting Multiple IDs

A client MAY send multiple id elements in an accceptance. Clients that receive a notification with multiple IDs MUST process each ID individually, as if multiple claims had been sent.

5.5 When to send?

To avoid race conditions and edge cases (including invisibility), if both the client and server support the Mine capability, the client SHOULD send ownership queries regardless of whether or not the client sees other resources for the same user online, or the capabilities of those other resources.

5.6 Legacy Clients

Clients that do not implement the Mine capability MAY be sent notifications by the server. The server MAY be optimized to avoid these notifications, however.

6 Implementation Notes

Some examples of events that might lead to a client sending an ownsership claim:

- Clicking on a toast notification for the message
- Bringing the client window to the front within a short time after receiving the message, where the message is then displayed to the user
- Bringing the tab containing the message to the front
- Beginning to type a response to the message
- Closing the window containing the message at least several seconds after the message was received
- Clicking an accept button next to a message
- Shutting down the screen saver while the message is in the top-most window
- A camera notices the user's eyes directed at the message

7 Accessibility Considerations

Some care should be given to the events that can cause ownsership claims, particularly in the MUC client implementations, such that users with different abilities all have a chance to claim ownership.

8 Security Considerations

Clients MUST ignore acceptance notifications received from other users.

9 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [4].

10 XMPP Registrar Considerations

This XEP proposes the new namespace 'urn:xmpp:tmp:mine:0'.

[4] 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/>.

11 XML Schema

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

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

  <xs:element name='whose'>
    <xs:complexType>
      <xs:attribute name='id' type='xs:string' use='required'/>
    </xs:complexType>
  </xs:element>

  <xs:element name='mine'>
    <xs:complexType>
      <xs:sequence>
        <xs:element ref='id' minOccurs='1' maxOccurs='unbounded'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name='id'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:NMTOKEN'/>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>
</xs:schema>
```