XEP-0311: MUC Fast Reconnect

Kevin Smith
mailto:kevin.smith@isode.com
xmpp:kevin.smith@isode.com

2012-01-25
Version 0.1

Status	Type	Short Name
Deferred	Standards Track	MFR

This document provides a protocol that can be used for limiting the amount of presence history needed when rejoining a MUC room.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

A client joining a Multi-User Chat (XEP-0045) [1] room will receive a significant volume of data, in the form of presence from the current room occupants and past (”context” or ”history”) messages. If the client has recently been in the room (for example if it has needed to reconnect only because of a networking error) it may already know most of the current state, and receipt of these data will be redundant. XEP-0045 provides a method for limiting the context messages received when joining but no method for limiting the duplication of known presence; this document expands slightly upon the former and provides the latter.

2 Requirements

Reduce the volume of redundant data sent to a client.

3 Discovery

MUC Rooms supporting this should have a disco feature of ”urn:xmpp:presence-session:0”. To avoid extra roundtrips for discovery, clients may speculatively send elements when initially joining a MUC, and treat the absense of appropriate elements in the responses to indicate a lack of support.

4 Use Cases

4.1 Initial Join

In this example, Romeo (romeo@montague.lit) is joining the MUC room orchard@chats.capulet.lit. To use MUC Fast Reconnect for future joins, the initial MUC join presence stanza MUST also contain a presence-session element in the namespace ”urn:xmpp:presence-session:0” with no attributes.

Listing 1: Romeo sends ’room join’ presence

```
<presence
    from='romeo@montague.lit/lane'
    to='orchard@chat.capulet.lit/Romeo'>
  <x xmlns='http://jabber.org/protocol/muc'/>
  <presence-session xmlns="urn:xmpp:presence-session:0"/>
</presence>
```

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

If a client has indicated that it's using MUC Fast Reconnect on its session, the MUC service MUST annotate the presence stanzas it sends with elements containing a presence-session element with namespace "urn:xmpp:presence-session:0", a "session" attribute and an "id" attribute, described below.

Listing 2: Romeo receives an occupant's presence

```
<presence
    from='orchard@chat.capulet.lit/Juliet'
    to='romeo@montague.lit/lane'>
  <x xmlns='http://jabber.org/protocol/muc#user'>
    <item affiliation='owner' role='moderator'/>
  </x>
  <presence-session xmlns="urn:xmpp:presence-session:0" session="
     u8e8t2thu" id="893oehh"/>
</presence>
```

Attributes:

- session: This is an opaque, string, identifier provider by the MUC room and may be used by the server to distinguish between stanzas that will potentially have the same id, but be for logically different rooms (e.g. because the room was deleted and another created with the same name in the time between the client leaving and trying to fast rejoin).
- id: This (also opaque, string) is a per-stanza identifier that, when paired with the session, gives a unique identifier for every presence update.
- type: This is empty for the normal case, and has a value of "resume" when a client requests a session resumption, or "truncated" when a room is announcing that it is not transmitting the full history to a joining client (see below)

4.2 Fast rejoin

If Romeo then leaves the room and wants to rejoin, his client can attempt a fast rejoin. To request only the presence changes since he was last an occupant, it includes a presence-session element in his room join stanza, again with namespace "urn:xmpp:presence-session:0", with a type attribute whose value is "resume" and the session and id attributes of the last presence it received from the room prior to leaving. This corresponds to the "last known state".

Listing 3: Romeo performs a fast rejoin on the room

```
<presence
    from='romeo@montague.lit/lane'
    to='orchard@chat.capulet.lit/Romeo'>
  <x xmlns='http://jabber.org/protocol/muc'/>
  <presence-session xmlns="urn:xmpp:presence-session:0" session="
     u8e8t2thu" id="893oehh" type="resume"/>
```

```
</presence>
```

When the MUC room receives a room fast rejoin request, it MUST either satisfy the request by sending incremental updates to the room state *or* send a complete set of stanzas to reestablish the current state.
If the room is able to update the client's state incrementally, it SHOULD only send those presence stanzas needed by the client to remove any occupants no longer in the room, add any newly joined occupants and update the state of any occupants whose status has changed (either because they have changed their presence sent to the room (e.g. changed to an 'away' state) or because their status within the MUC has changed (e.g. they have become a moderator). If the incremental stanzas would present a greater volume of data than a fresh join, it is RECOMMENDED that the server sends fresh join information instead.
If the server is unable to calculate the stanzas required to send the client an incremental update (or if it is to send a fresh join for some other reason), it MUST first send an 'unavailable' presence from the room's bare JID, followed by a normal full join, as above
All the presence stanzas (apart from the initial unavailable presence used to reset state before a clean join) MUST contain the presence-session element as described above.

Listing 4: Romeo receives an 'unavailable' presence from the room, indicating that it is a clean join.

```
<presence
    to='romeo@montague.lit/lane'
    from='orchard@chat.capulet.lit'
    type='unavailable'/>
```

The room SHOULD only send the unavailable presence, forcing a sending of all the occupants' presence, if it would either result in fewer transmitted stanzas than sending the necessary delta, or it is unable to provide the necessary delta (such as if too much time has past and it no longer has records of the old state).

4.3 Receiving only unseen messages

XEP-0045 provides several ways to limit the history/context messages received on join, but none of these allow a client to accurately request only the messages they have yet to see. To address this, the MUC service annotates each message with an id (in the same manner as presence, above), and the room will consider only messages since the last stanza the client received when applying the default/maxchars/maxstanzas/seconds/since rules from -45 for sending context. If the room doesn't send the full history of messages that the client has yet to receive (e.g. due to the application of history controls or because the server hasn't stored them) it sends a message to the client such the client knows it only has partial history.
Some informal examples:

- There have been 9 messages since the client was last in the room, and the client joins requesting a maximum of 20 stanzas. The client receives the 9 messages.
- There have been 9 messages since the client was last in the room, and the client joins requesting a maximum of 5 stanzas. The client first receives a message that the history has been truncated and then the 5 most recent messages.
- There have been 9 messages since the client was last in the room, and the client requests that no history be sent. The client receives the truncation message and no history

Listing 5: Romeo receives annotated message while in room

```
<message
    to='romeo@montague.lit/lane'
    from='orchard@chat.capulet.lit/Juliet'
    type='groupchat'>
  <body>Ay me!</body>
  <presence-session xmlns="urn:xmpp:presence-session:0" session="
     u8e8t2thu" id="893ou22"/>
</message>
```

When sending broadcast messages from the room, the service MUST annotate them with a presence-session stanza with xmlns "urn:xmpp:presence-session:0" and session and id attributes as defined above for presence stanzas.

Listing 6: Room informs Romeo's client that it will not transmit the full missing history

```
<message
    to='romeo@montague.lit/lane'
    from='orchard@chat.capulet.lit'
    type='groupchat'>
  <presence-session xmlns="urn:xmpp:presence-session:0" type="
     truncated"/>
</message>
```

When the room does not send the full history of all messages that the client has not received, it MUST (prior to sending any history and subsequent to sending presence) send a message stanza with a payload whose name is 'presence-session' and namespace is "urn:xmpp:presence-session:0" with an attribute named "type" whose value is "truncated". This lets the client know that it is missing history and it could choose to display this to the user in some way.

5 Security Considerations

This specification doesn't add additional security considerations beyond those of its dependencies..

6 IANA Considerations

None.

7 XMPP Registrar Considerations

Needs a namespace.

8 XML Schema

When advanced.