



XMPP

XEP-0436: MUC presence versioning

JC Brand
<mailto:jc@opkode.com>
<xmpp:jc@opkode.com>

Matthew Wild
<mailto:mwild1@gmail.com>
<xmpp:me@matthewwild.co.uk>

2020-05-10
Version 0.2.0

Status	Type	Short Name
Experimental	Standards Track	omnipresent-muc-affiliates

This specification defines a versioning mechanism which reduces the amount of presence traffic in a XEP-0045 MUC

Legal

Copyright

This XMPP Extension Protocol is copyright © 1999 – 2020 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	How it works	1
3	Determining support	2
4	Requirements	3
4.1	Always broadcast presence for affiliated users	3
4.2	Include a reset token when the client's version number has expired	3
5	Business Rules	4
6	Security Considerations	4
7	IANA Considerations	4
8	XMPP Registrar Considerations	4
8.1	Protocol Namespaces	4
8.2	Protocol Versioning	5

1 Introduction

Many modern-day non-XMPP groupchat implementations have discarded the metaphor of physical presence inside a room that a user must enter and exit, as implemented by [Multi-User Chat \(XEP-0045\)](#)¹. The newer [Mediated Information eXchange \(MIX\) \(XEP-0369\)](#)² has therefore made presence subscriptions optional.

Often it no longer makes sense for a chat service to require that a user is "present" in order for them to be addressed by other occupants or to receive messages, especially if the chat implementation will inform you out-of-band, for example via push notifications or email. The notion of "room presence" is therefore less relevant than before, and in some cases can be done away with entirely.

Broadcasting all XEP-0045 MUC participants' presences to one another scales quadratically ($O(n^2)$) and can greatly increase the amount of network traffic, for potentially negligible gain.

Even though the metaphorical concept of presence inside a room might no longer be relevant for a groupchat implementation, `<presence/>` stanzas might still contain useful metadata, such as the user's affiliation or [Hats \(XEP-0317\)](#)³.

This XEP defines a versioning mechanism (similar to roster versioning in [RFC 6121](#)⁴) whereby the amount of presence traffic in a MUC may be greatly reduced.

2 How it works

A client that supports MUC presence versioning needs to keep track and store the presence states of all MUC occupants, across multiple MUC sessions. Similarly, a MUC service which supports presence versioning will also need to maintain a changelog of version numbers and corresponding presence states.

Before the client enters a MUC, it SHOULD use service discovery to check whether presence versioning is supported (see [determining support](#) below.). If MUC presence versioning is supported, the client MAY include a `<version>` tag with a 'ver' attribute set to the last known version inside the `<pathhttp://jabber.org/protocol/muc#user{x}>` tag of the `<presence/>` stanza, which it sends to join the MUC.

If MUC presence versioning is not supported by the server, the client MUST NOT include a 'ver' attribute.

Listing 1: User specifies the last known version when seeking to enter a MUC

```
<presence
```

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

²XEP-0369: Mediated Information eXchange (MIX) <<https://xmpp.org/extensions/xep-0369.html>>.

³XEP-0317: Hats <<https://xmpp.org/extensions/xep-0317.html>>.

⁴RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence <<http://tools.ietf.org/html/rfc6121>>.

```

    from='hag66@shakespeare.lit/pda'
    id='n13mt31'
    to='coven@chat.shakespeare.lit/thirdwitch'>
  <x xmlns='http://jabber.org/protocol/muc#user'>
    <version xmlns='urn:xmpp:muc-presence-versioning:0' ver='ver16' />
  </x>
</presence>

```

The MUC will return only those presences that have changed since the version indicated by the client, and in the self-presence of the joining user it will add a <version> tag with a 'ver' attribute set to the latest version number inside the <pathhttp://jabber.org/protocol/muc#user}x> tag. The client must save the version number (and continuously update with newer versions as they're received) and use that next time it joins the MUC.

Listing 2: Service Sends New Occupant's Presence to New Occupant

```

<presence
  from='coven@chat.shakespeare.lit/thirdwitch'
  id='n13mt31'
  to='hag66@shakespeare.lit/pda'>
  <x xmlns='http://jabber.org/protocol/muc#user'>
    <version xmlns='urn:xmpp:muc-presence-versioning:0' ver='ver17' />
    <item affiliation='member' role='participant' />
    <status code='110' />
  </x>
</presence>

```

When presence versioning is enabled, every subsequent <presence/> stanza sent by the server MUST include a new version number, which replaces the existing one saved by the client.

3 Determining support

If a MUC implements presence versioning, it MUST specify the 'urn:xmpp:muc-presence-versioning:0' feature in its service discovery information features, as specified in [Service Discovery \(XEP-0030\)](#)⁵.

Listing 3: Client queries for information about a specific MUC

```

<iq type='get'
  from='romeo@montague.example/orchard'
  to='room@muc.shakespeare.example'
  id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info' />

```

⁵XEP-0030: Service Discovery <<https://xmpp.org/extensions/xep-0030.html>>.

```
</iq>
```

Listing 4: The MUC advertises support for presence versioning

```
<iq type='result'
  to='romeo@montague.example/home'
  from='room@muc.shakespeare.example'
  id='info1'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
  ...
    <feature var='urn:xmpp:muc-presence-versioning:0' />
  ...
  </query>
</iq>
```

4 Requirements

4.1 Always broadcast presence for affiliated users

A MUC that supports presence versioning **MUST** broadcast presence for all affiliated users, including those who are currently 'unavailable'. XEP-0045 documents the status code '102', which is used to indicate that a MUC shows unavailable members.

The only exception is when a MUC has been explicitly configured to only broadcast presence from occupants above a certain affiliation, (see the [presence broadcast](#) section of XEP-0045). In order for a user to permanently join a room, and therefore become affiliated so that they are included in presence broadcasts, they **MAY** be allowed to register themselves as members in the MUC. XEP-0045 describes in [section 7.10 "Registering with a Room"](#) how a user may register themselves with a room, thereby receiving the "member" affiliation and having their preferred nickname reserved in that room.

4.2 Include a reset token when the client's version number has expired

If a MUC receives a presence version number that's so old, that it no longer has the corresponding state available, it **MUST** include a <reset> token in the first <presence> stanza it sends (regardless of whom in the MUC the presence relates to). The first presence stanza **MAY** be from the MUC itself (as shown in the example below). Afterwards, the MUC **MUST** then send all presences (including 'unavailable' for affiliated users) as if the client is starting from a blank slate.

Listing 5: Service Sends a Reset Token

```
<presence from='coven@chat.shakespeare.lit'
  id='r3s3t-m3'
  to='hag66@shakespeare.lit/pda'>
```

```
<x xmlns='http://jabber.org/protocol/muc#user'>
  <reset xmlns='urn:xmpp:muc-presence-versioning:0' ver='ver36'
    />
</x>
</presence>
```

5 Business Rules

Even if the client did not include a <version> tag with a 'ver' attribute in its "join" <presence/> stanza, the server SHOULD still return a <version> tag with 'ver' attribute (set to the latest version number) on all relevant <presence/> stanzas. This allows clients to bootstrap MUC presence versioning without having to do a service discovery query first.

If the client has not yet saved a presence version number and the corresponding presence states, then it MUST bootstrap presence versioning by sending a 'ver' attribute set to the empty string (i.e., ver="").

In some cases, the presence states being kept track of by the MUC service MAY be reduced to a minimum of only two states, 'available' and 'unavailable'. This can drastically reduce the number of states the server needs to keep track of, at the cost of not allowing users to provide more fine-grained reporting of their level of availability.

6 Security Considerations

The MUC service should strip the <version> tag from the user's <presence> before relaying it to other occupants, to avoid leaking information on when last the user joined the MUC.

7 IANA Considerations

None.

8 XMPP Registrar Considerations

8.1 Protocol Namespaces

The XMPP Registrar ⁶ includes 'urn:xmpp:muc-presence-versioning:0' in its registry of protocol namespaces (see <<https://xmpp.org/registrar/namespaces.html>>).

⁶The 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/>>.

- urn:xmpp:muc-presence-versioning:0

8.2 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.