XEP-0486: MUC Avatars

Emmanuel Gil Peyrot
mailto:linkmauve@linkmauve.fr
xmpp:linkmauve@linkmauve.fr

2024-03-10
Version 0.1.0

Status	Type	Short Name
Experimental	Historical	NOT_YET_ASSIGNED

This specification describes how to publish and retrieve avatars in rooms.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

Avatars are small images people often use to identify each other very quickly in chat applications. They are well defined for users, in User Avatar (XEP-0084) [1] and vCard-Based Avatars (XEP-0153) [2], but until now chat rooms all shared a default icon. This extension provides a way for owners to associates an avatar to their chat room, and for users to discover that an avatar is associated and display it accordingly.
XMPP services have traditionally allowed owners to set a vCard-temp on a MUC using vcard-temp (XEP-0054) [3], this extension tries to keep as much of it as possible so existing applications don't have to be modified too much.
Some implementations recently chose to advertise those avatars using the existing vCard-Based Avatars (XEP-0153) [4] extension in <presence/>, but it exposed issues in other implementations, and was only available when the user is already present in the room, not before joining it (for example when listing all available rooms).
A future extension superseding this one could define a method based on User Avatar (XEP-0084) [5], with a PubSub service on the room's bare JID containing the metadata and data nodes. Such a specification should also define a compatibility profile similar to User Avatar to vCard-Based Avatars Conversion (XEP-0398) [6] for user avatars, enabling the coexistence of both versions until the present one is deemed obsolete.

2 Requirements

This specification SHOULD:

- Allow authorised entities to set an avatar on a MUC.
- Allow authorised entities to remove a previously-set avatar on a MUC.
- Allow users to discover an avatar is set on a MUC.
- Allow users to request the avatar of a MUC.
- Let users know that the avatar of a MUC changed while they are present in said MUC.
- Let users discover the avatar even when not present in the MUC.
- Stay as compatible as possible with the current usage of avatars in MUC.

[1] XEP-0084: User Avatar <https://xmpp.org/extensions/xep-0084.html>.
[2] XEP-0153: vCard-Based Avatars <https://xmpp.org/extensions/xep-0153.html>.
[3] XEP-0054: vcard-temp <https://xmpp.org/extensions/xep-0054.html>.
[4] XEP-0153: vCard-Based Avatars <https://xmpp.org/extensions/xep-0153.html>.
[5] XEP-0084: User Avatar <https://xmpp.org/extensions/xep-0084.html>.
[6] XEP-0398: User Avatar to vCard-Based Avatars Conversion <https://xmpp.org/extensions/xep-0398.html>.

3 Use Cases

3.1 Discover the features supported by a service

Before trying to use avatars, a client must check that the group chat service hosting a room does support them.

Listing 1: User's client discovers the features of a MUC service

```
<iq type='get'
    id='p87Ne'
    from='romeo@montague.example.net/garden'
    to='chat.shakespeare.example.org'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 2: Room advertises support for vcard-temp

```
<iq type='result'
    id='p87Ne'
    to='romeo@montague.example.net/garden'
    from='chat.shakespeare.example.org'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <identity
        category='conference'
        type='text'
        name='Shakespearean Chat Service'/>
    <feature var='http://jabber.org/protocol/muc'/>
    <feature var='vcard-temp'/>
    ...
  </query>
</iq>
```

3.2 Owner of the Room Publishes Avatar

Before anyone can see an avatar attached to the room, an owner or some other priviledged entity must publish a vCard-temp containing the avatar's data, using the protocol defined in vcard-temp (XEP-0054) [7].

Listing 3: Owner's client publishes avatar to the room

```
<iq type='set'
    id='7fP13'
    from='romeo@montague.example.net/garden'
    to='garden@chat.shakespeare.example.org'>
  <vCard xmlns='vcard-temp'>
```

[7] XEP-0054: vcard-temp <https://xmpp.org/extensions/xep-0054.html>.

```
    <PHOTO>
      <TYPE>image/svg+xml</TYPE>
      <BINVAL>
          PHN2ZyB4bWxucz0iaHR0cDovL3d3dy53My5vcmcvMjAwMC9zdmciIHdpZHRoPSIzMiIgaGVpZ2h0
          +
          CiA8cmVjdCB4PSIwIiB5PSIwIiB3aWR0aD0iMzIiIGhlaWdodD0iMzIiIGZpbGw9InJlZCIvPgo8
          </BINVAL>
    </PHOTO>
  </vCard>
</iq>
```

Listing 4: Room acknowledges publish action

```
<iq type='result'
    id='7fP13'
    to='romeo@montague.example.net/garden'
    from='garden@chat.shakespeare.example.org'/>
```

There is no other action required on the owner's end.
If the room doesn't support support avatars, it must return a service-unavailable error.

Listing 5: Room doesn't support vCard-temp

```
<iq type='error'
    id='7fP13'
    to='romeo@montague.example.net/garden'
    from='garden@chat.shakespeare.example.org'>
  <error type='cancel'>
    <service-unavailable xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

If the user trying to publish an avatar isn't allowed to do so, the room must return a forbidden error, see the Security Considerations.

Listing 6: User is not allowed to set vCard-temp

```
<iq type='error'
    id='7fP13'
    to='romeo@montague.example.net/garden'
    from='garden@chat.shakespeare.example.org'>
  <error type='auth'>
    <forbidden xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <text>Only owners are allowed to set avatars.</text>
  </error>
</iq>
```

The room should then broadcast a notification that the configuration changed to all users present.

Listing 7: Room broadcasts a configuration change

```
<message type='groupchat'
         to='romeo@montague.example.net/garden'
         from='garden@chat.shakespeare.example.org'>
  <x xmlns='http://jabber.org/protocol/muc#user'>
    <status code='104'/>
  </x>
</message>

<message type='groupchat'
         to='juliet@capulet.example.com/balcony'
         from='garden@chat.shakespeare.example.org'>
  <x xmlns='http://jabber.org/protocol/muc#user'>
    <status code='104'/>
  </x>
</message>
```

Setting an empty vCard unpublishes the avatar.

Listing 8: Owner's client removes a room's avatar

```
<iq type='set'
    id='83tFs'
    from='romeo@montague.example.net/garden'
    to='garden@chat.shakespeare.example.org'>
  <vCard xmlns='vcard-temp'/>
</iq>
```

3.3 User Discovers the Avatar

At any point, whether it is during a join in order to display it in its UI, after having discovered the list of the rooms and to list them with additional information, or when receiving a <status code='104'/> configuration change notification, a user's client can discover information about a room.

Listing 9: User's client discovers information about a room

```
<iq type='get'
    id='K92am'
    from='juliet@capulet.example.com/balcony'
    to='garden@chat.shakespeare.example.org'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

If the room has had an avatar published, it should advertise it in its 'muc#roominfo' extension form, using the vCard-Based Avatars (XEP-0153) [8] hash computation method.

[8] XEP-0153: vCard-Based Avatars <https://xmpp.org/extensions/xep-0153.html>.

Listing 10: Room advertises its avatar hash

```
<iq type='result'
    id='K92am'
    to='juliet@capulet.example.com/balcony'
    from='garden@chat.shakespeare.example.org'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <identity
        category='conference'
        type='text'
        name='The Garden'/>
    <feature var='http://jabber.org/protocol/muc'/>
    <feature var='vcard-temp'/>
    ...
    <x xmlns='jabber:x:data' type='result'>
      <field var='FORM_TYPE' type='hidden'>
        <value>http://jabber.org/protocol/muc#roominfo</value>
      </field>
      ...
      <field var='muc#roominfo_avatarhash'
             type='text-multi'
             label='Avatar hash'>
        <value>a31c4bd04de69663cfd7f424a8453f4674da37ff</value>
      </field>
      ...
    </x>
  </query>
</iq>
```

This 'muc#roominfo_avatarhash' will not be present when the room doesn't have an avatar set.

3.4 User Retrieves the vCard

At this point the client knows the hash and can retrieve the room's vCard-temp.

Listing 11: User's client retrieves the vCard-temp

```
<iq type='get'
    id='uD10h'
    from='juliet@capulet.example.com/balcony'
    to='garden@chat.shakespeare.example.org'>
  <vCard xmlns='vcard-temp'/>
</iq>
```

Listing 12: Room returns the vCard-temp containing the avatar

```
<iq type='result'
```

```
    id='uD10h'
    to='juliet@capulet.example.com/balcony'
    from='garden@chat.shakespeare.example.org'>
  <vCard xmlns='vcard-temp'>
    <PHOTO>
      <TYPE>image/svg+xml</TYPE>
      <BINVAL>
          PHN2ZyB4bWxucz0iaHR0cDovL3d3dy53My5vcmcvMjAwMC9zdmciIHdpZHRoPSIzMiIgaGVpZ2h0
          +
          CiA8cmVjdCB4PSIwIiB5PSIwIiB3aWR0aD0iMzIiIGhlaWdodD0iMzIiIGZpbGw9InJlZCIvPgo8
          </BINVAL>
    </PHOTO>
  </vCard>
</iq>
```

The client then has to decode the <BINVAL/> content from base64, hash it with sha1 and compare it with the advertised hash, and if it matches uses it as the room avatar under the <TYPE/> media type.

4 Business Rules

An application MUST support the image/png media type, SHOULD support image/jpeg, image/gif and image/svg+xml, and MAY support additional formats.
A room SHOULD NOT include a 'muc#roominfo_avatarhash' field if it doesn't have an avatar set.

5 Implementation Notes

5.1 Multiple Stored Version of an Avatar

Multiple <PHOTO/> elements may be present in a vCard, in which case they should all represent the same image and the 'muc#roominfo_avatarhash' field must contain a hash of all of them.

Listing 13: Owner's client publishes avatar in two different formats

```
<iq type='set'
    id='7fP13'
    from='romeo@montague.example.net/garden'
    to='garden@chat.shakespeare.example.org'>
  <vCard xmlns='vcard-temp'>
    <PHOTO>
      <TYPE>image/svg+xml</TYPE>
```

```
      <BINVAL>
          PHN2ZyB4bWxucz0iaHR0cDovL3d3dy53My5vcmcvMjAwMC9zdmciIHdpZHRoPSIzMiIgaGVpZ2h0
          +
          CiA8cmVjdCB4PSIwIiB5PSIwIiB3aWR0aD0iMzIiIGhlaWdodD0iMzIiIGZpbGw9InJlZCIvPgo8
          </BINVAL>
    </PHOTO>
    <PHOTO>
      <TYPE>image/png</TYPE>
      <BINVAL>
          iVBORw0KGgoAAAANSUhEUgAAACAAAAAgAQMAAABJtOi3AAAAB3RJTUUH4ggVERoVAPsrMgAAAAlw
          +/
          AAAABl0RVh0U29mdHdhcmUAd3d3Lmlua3NjYXBlLm9yZ5vuPBoAAAAEZ0FNQQAAsY8L
          /GEFAAAAIGNIUk0AAHomAACAhAAA+
          gAAAIDoAAB1MAAA6mAAADqYAAAXcJy6UTwAAAAGUExURf8AAP///0
          EdNBEAAAABYktHRAH/
          Ai3eAAAADElEQVQI12NgGNwAAACgAAFhJX1HAAAAAElFTkSuQmCC</BINVAL
          >
    </PHOTO>
  </vCard>
</iq>
```

Listing 14: Room advertises both hashes

```
<iq type='result'
    id='K92am'
    to='juliet@capulet.example.com/balcony'
    from='garden@chat.shakespeare.example.org'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <identity
        category='conference'
        type='text'
        name='The Garden'/>
    <feature var='http://jabber.org/protocol/muc'/>
    <feature var='vcard-temp'/>
    ...
    <x xmlns='jabber:x:data' type='result'>
      <field var='FORM_TYPE' type='hidden'>
        <value>http://jabber.org/protocol/muc#roominfo</value>
      </field>
      ...
      <field var='muc#roominfo_avatarhash'
             type='text-multi'
             label='Avatar hash'>
        <value>a31c4bd04de69663cfd7f424a8453f4674da37ff</value>
        <value>b9b256f999ded52c2fa14fb007c2e5b979450cbb</value>
      </field>
      ...
    </x>
  </query>
</iq>
```

5.2 Previous Usage of Presence for Avatar Advertising

Some existing implementations send or expect a presence from the room's bare JID in order to detect an avatar being published. This had several issues, with existing clients handling that as a presence from a user with an empty nick or downright triggering an error, and was only available if the client was already present in the room, preventing any usecase where it would get displayed before entering the room.
For those reasons, this XEP doesn't encourage this way of advertising the presence of an avatar, but for reference it would look like a vCard-Based Avatars (XEP-0153) [9] presence payload:

Listing 15: Room advertises a non-standard vCard update in a presence

```
<presence from='garden@chat.shakespeare.example.org'>
  <x xmlns='vcard-temp:x:update'>
    <photo>a31c4bd04de69663cfd7f424a8453f4674da37ff</photo>
  </x>
</presence>
```

6 Security Considerations

A server should take care that only allowed entities can publish a vCard-temp on a MUC, for instance room owners or service administrators.

7 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [10].

8 XMPP Registrar Considerations

8.1 Field Standartization

The registrar shall add the following field to the 'muc#roominfo' data form:

[9] XEP-0153: vCard-Based Avatars <https://xmpp.org/extensions/xep-0153.html>.
[10] 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/>.

8.1.1 muc#roominfo FORM_TYPE

```
<form_type>
  <name>http://jabber.org/protocol/muc#roominfo</name>
  <doc>XEP-XXXX</doc>
  <desc>Form extension for avatar support in a Multi-User Chat (MUC)
     room.</desc>
  <field
      var='muc#roominfo_avatarhash'
      type='text-multi'
      label='Hash of the vCard-temp avatar of this room'/>
</form_type>
```

9 Acknowledgements

Thanks to the Ejabberd developers for their MUC vCard tutorial, and to Sam Whited and Matthew Wild for their feedback.