XEP-0405: Mediated Information eXchange (MIX): Participant Server Requirements

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

Steve Kille
mailto:steve.kille@isode.com
xmpp:steve.kille@isode.com

2020-11-03
Version 0.5.1

Status	Type	Short Name
Experimental	Standards Track	MIX-PAM

This document defines an extension to Mediated Information eXchange (MIX) specified in XEP-0369. It specifies behaviour of an XMPP server to which MIX Clients connect in order to enable correct operation of these clients in conjunction with a MIX server.

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

The Mediated Information eXchange (MIX) protocol framework and core capabilities are specified in Mediated Information eXchange (MIX) (XEP-0369) [1] (MIX-CORE). MIX-CORE and Mediated Information eXchange (MIX): Presence Support. (XEP-0403) [2] define behaviour of a MIX server supporting MIX channels. In order for a MIX system to operate correctly, the XMPP server connecting MIX clients MUST follow the rules set out in this specification to achieve correct MIX operation. This specification also sets out requirements for XMPP clients, so a MIX client MUST follow both the rules of XMPP-CORE and this specification.
A MIX channel does not send messages and presence directly to the MIX client of a channel participant, but addresses them to the participant using the participant's bare JID. The participant's server MUST then handle these messages and pass them on to zero, one or multiple clients. To enable MIX to work, this behaviour is necessary and so the server of every MIX client MUST follow the rules set out in this specification. This approach enables flexible support of multiple clients for a MIX channel participant. The MIX model is that a user will join a channel over an extended period, and that the user (not a specific client used by the user) joins the channel. The primary subscription is with the client's bare JID. There are a number of MIX requirements on behaviour of the MIX Participant's server, which are summarized here:

1. Messages from a MIX client to a MIX channel will go direct to the MIX service. They are relayed, but not modified, by the participant's server.
2. Messages from a MIX channel to a MIX client are always sent to the MIX participant's server (addressed by bare JID) and MUST be handled by the MIX participant's server.
3. All MIX messages received by the MIX participant's server for a participant MUST be stored using MAM in the participant's archive.
4. The MIX participant's server will only forward messages to online clients and will not forward messages if no clients are online. This means that a MIX client needs to resynchronize with all MIX channels when it comes online. This message synchronization will happen between the MIX client and the MAM archive held on the MIX participant's server.
5. The MIX client will generally send management and other messages directly to the MIX channel and this MUST be done except where this specification requires otherwise.
6. The user's roster contains each MIX channel to which the user is subscribed. To achieve this the user's server needs to manage the roster on behalf of the user. For this reason, MIX join and leave commands MUST be sent by a client to the user's server. This enables the user's server to correctly manage the roster on behalf of the user.

[1] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.
[2] XEP-0403: Mediated Information eXchange (MIX): Presence Support. <https://xmpp.org/extensions/xep-0403.html>.

Messages from a MIX channel to a MIX participant (which will be of type=groupchat), presence information, and other information sent as a result of MIX channel subscription are sent to the participant's server using the participant's bare JID. This means that the MIX participant's server MUST implement the modification of the standard RFC 6121 [3] message processing rules specified here.

2 Use Cases

This section defines behaviour REQUIRED by MIX for servers supporting MIX participants. This functionality MUST be provided by servers used by clients that participate in MIX channels. In future, this specification MAY be incorporated into Pubsub Account Management (XEP-0376) [4] (PAM) which follows a similar model.

2.1 Server Identification of MIX Capabable Clients

A MIX User's server MUST determine which online clients support MIX. This will enable the server to send MIX traffic to all MIX capable clients, but not to other clients. A MIX capable client MAY choose to come online and not advertise MIX capability. The mechanism for a server to discover client capability is described in Discovering Client MIX Capability.

2.2 Messages From MIX Channels

Messages from a MIX channel will usually be handled by the participant's server. The only exception to this is where the MIX channel is responding directly to messages from the client. Messages and presence distributed by a MIX channel will always be sent to the participant's server and addressed to the user's bare JID. The participant's server will archive the message in MAM and send on the messages from the channel to each of the user's online clients which advertise MIX capability. If there are no such clients activated, the message is not sent to any clients.
Messages sent to the participant's sever will always be addressed to the user's bare JID. The participant's server will modify the recipient to the full JID of each client to which the message is forwarded. The following example, repeated from Mediated Information eXchange (MIX) (XEP-0369) [5], shows a message distributed by a MIX channel and directed to the participant's server with the participant's bare JID.

Listing 1: Channel Reflects Message to Participants

[3] RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence <http://tools.ietf.org/html/rfc6121>.
[4] XEP-0376: Pubsub Account Management <https://xmpp.org/extensions/xep-0376.html>.
[5] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.

```
<message from='coven@mix.shakespeare.example/123456'
         to='hecate@shakespeare.example'
         id='77E07BB0-55CF-4BD4-890E-3F7C0E686BBD'
         type='groupchat'>
  <body>Harpier cries: 'tis␣time,␣'tis time.</body>
  <mix xmlns='urn:xmpp:mix:core:1'>
    <nick>thirdwitch</nick>
    <jid>hag66@shakespeare.example</jid>
  </mix>
</message>
```

The server receiving the message will then deliver the messages to all online clients. Messages are delivered to all available online resources irrespective of status and resource priority. The following example shows how the participant's server modifies the inbound message to replace the bare JID in the 'to' with a full JID for each of two active MIX clients.

Listing 2: Participant's Server Sends Modified Message to two Clients

```
<message from='coven@mix.shakespeare.example/123456'
         to='hecate@shakespeare.example/UUID-x4r/2491'
         id='77E07BB0-55CF-4BD4-890E-3F7C0E686BBD'
         type='groupchat'>
  <body>Harpier cries: 'tis␣time,␣'tis time.</body>
  <mix xmlns='urn:xmpp:mix:core:1'>
    <nick>thirdwitch</nick>
    <jid>hag66@shakespeare.example</jid>
  </mix>
</message>

<message from='coven@mix.shakespeare.example/123456'
         to='hecate@shakespeare.example/UUID-b5b/0114'
         id='77E07BB0-55CF-4BD4-890E-3F7C0E686BBD'
         type='groupchat'>
  <body>Harpier cries: 'tis␣time,␣'tis time.</body>
  <mix xmlns='urn:xmpp:mix:core:1'>
    <nick>thirdwitch</nick>
    <jid>hag66@shakespeare.example</jid>
  </mix>
</message>
```

2.3 Messages To MIX Channels

Messages sent by a MIX channel participant to the MIX channel are always sent from a MIX client directly to the channel. The participant's server relays the message but does not modify the messages.

2.4 Client Determines MIX Capability of Client's Server

Servers supporting this specification MUST advertise this to clients for which they wish to support this specification. A client wishing to use MIX MUST check for this capability in the local server before using MIX, by verifying support for the client's account. The capability is represented by the 'urn:xmpp:mix:pam:2' feature. In addition to this the server MAY advertize the 'urn:xmpp:mix:pam:2#archive' feature, which shows that the local server archives MIX messages.

Listing 3: Client Determines MIX Capability for Server Account

```
<iq from='hag66@shakespeare.example/UUID-c8y/1573'
    id='lx09df27'
    type='get'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>

<iq id='lx09df27'
    to='hag66@shakespeare.example/UUID-c8y/1573'
    type='result'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <feature var='urn:xmpp:mix:pam:2'/>
    <feature var='urn:xmpp:mix:pam:2#archive'/>
  </query>
</iq>
```

2.5 MIX Management and Discovery

Most interaction between a MIX client and a MIX channel is directly between the client and the channel. The participant's server relays the message but does not modify the messages. In particular configuration management and discovery is direct. Interaction will be direct, unless explicitly stated otherwise in this specification.

2.6 MIX Join and Leave Support on Local Server

Channel Join and Leave functions operate indirectly through the participant's server. The reason for this is that where a channel shares user presence, the channel is included in the user's roster which is managed in the local server. The Join and Leave functions lead to roster changes and so they MUST go through the participant's server. To achieve this, this specification wraps the operations so that the server can correctly route messages.

2.7 Joining a Channel

A user joins a channel by sending a MIX "client-join" command from one of the user's clients, which wraps the "join" command specified in Mediated Information eXchange (MIX) (XEP-0369) [6]. Mediated Information eXchange (MIX) (XEP-0369) [7] specifies how the join command works, and so this specification considers only the wrapping and client actions. The <client-join/> is a child element of <iq/> element. The <client-join/> element is qualified by the 'urn:xmpp:mix:pam:2' namespace. The channel being joined is specified by a 'channel' attribute in the <client-join/> element, which is used by the server to correctly address the join. The <join> is specified in Mediated Information eXchange (MIX) (XEP-0369) [8] and is a child element of <client-join/>.

Listing 4: Client sends request to local server to Join a MIX Channel

```
<iq type='set'
    from='hag66@shakespeare.example/UUID-a1j/7533'
    to='hag66@shakespeare.example'
    id='E6E10350-76CF-40C6-B91B-1EA08C332FC7'>
  <client-join xmlns='urn:xmpp:mix:pam:2' channel='coven@mix.
      shakespeare.example'>
    <join xmlns='urn:xmpp:mix:core:1'>
      <subscribe node='urn:xmpp:mix:nodes:messages'/>
      <subscribe node='urn:xmpp:mix:nodes:presence'/>
      <subscribe node='urn:xmpp:mix:nodes:participants'/>
      <subscribe node='urn:xmpp:mix:nodes:info'/>
    </join>
  </client-join>
</iq>
```

The information in this message is used to derive the message sent to the MIX channel. The 'from' is the bare JID of the user. The 'to' is the channel from the client join 'channel' attribute. The 'id' is taken from the client join message. The <join> is taken from the client join message. This is shown in the following example, repeated from the earlier specification of channel behaviour.

Listing 5: Participant's Server sends Join to MIX Channel

```
<iq type='set'
    from='hag66@shakespeare.example'
    to='coven@mix.shakespeare.example'
    id='E6E10350-76CF-40C6-B91B-1EA08C332FC7'>
  <join xmlns='urn:xmpp:mix:core:1'>
    <subscribe node='urn:xmpp:mix:nodes:messages'/>
    <subscribe node='urn:xmpp:mix:nodes:presence'/>
```

[6] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.
[7] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.
[8] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.

```
    <subscribe node='urn:xmpp:mix:nodes:participants'/>
    <subscribe node='urn:xmpp:mix:nodes:info'/>
  </join>
</iq>
```

The MIX service will process this request and respond as specified by Mediated Information eXchange (MIX) (XEP-0369) [9]. An example response taken from Mediated Information eXchange (MIX) (XEP-0369) [10] is shown below.

Listing 6: Channel responds to User's Server

```
<iq type='result'
    from='coven@mix.shakespeare.example'
    to='hag66@shakespeare.example'
    id='E6E10350-76CF-40C6-B91B-1EA08C332FC7'>
  <join xmlns='urn:xmpp:mix:core:1' jid='123456#coven@mix.shakespeare.
      example'>
    <subscribe node='urn:xmpp:mix:nodes:messages'/>
    <subscribe node='urn:xmpp:mix:nodes:presence'/>
    <subscribe node='urn:xmpp:mix:nodes:participants'/>
    <subscribe node='urn:xmpp:mix:nodes:info'/>
  </join>
</iq>
```

The user's server will then make roster modifications as set out in a later section of this specification. After making these changes, the user's server will send the client-join response containing the server's join response back to the client that made the join request. This has the 'from' set to the user's bare JID and the 'to' set to the client's full JID. This is illustrated below:

Listing 7: User's Server sends response to Client

```
<iq type='result'
    from='hag66@shakespeare.example'
    to='hag66@shakespeare.example/UUID-a1j/7533'
    id='E6E10350-76CF-40C6-B91B-1EA08C332FC7'>
      <client-join xmlns='urn:xmpp:mix:pam:2'>
        <join xmlns='urn:xmpp:mix:core:1'
              jid='123456#coven@mix.shakespeare.example'>
          <subscribe node='urn:xmpp:mix:nodes:messages'/>
          <subscribe node='urn:xmpp:mix:nodes:presence'/>
          <subscribe node='urn:xmpp:mix:nodes:participants'/>
          <subscribe node='urn:xmpp:mix:nodes:info'/>
        </join>
      </client-join>
</iq>
```

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

2.8 Leaving a Channel

Users generally remain in a channel for an extended period of time. The process for leaving a MIX channel is specified in Mediated Information eXchange (MIX) (XEP-0369) [11]. When a user desires to leave a channel, it will issue a client-leave request to the local server. The <client-leave/> is a child element of <iq/> element. The <client-leave/> element is qualified by the 'urn:xmpp:mix:pam:2' namespace. The channel being left is specified by a 'channel' attribute in the <client-leave/> element, which is used by the server to correctly address the join. The <leave> is specified in Mediated Information eXchange (MIX) (XEP-0369) [12] and is a child element of <client-leave/>. This shown in the following example.

Listing 8: Client Requests to Leave a Channel

```
<iq type='set'
    from='hag66@shakespeare.example/UUID-a1j/7533'
    to='hag66@shakespeare.example'
    id='E6E10350-76CF-40C6-B91B-1EA08C332FC7'>
  <client-leave xmlns='urn:xmpp:mix:pam:2'
                channel='coven@mix.shakespeare.example'>
    <leave xmlns='urn:xmpp:mix:core:1'/>
  </client-leave>
</iq>
```

The user's server will then generate a matching leave request to the MIX channel based on this information. This example is taken from Mediated Information eXchange (MIX) (XEP-0369) [13].

Listing 9: User's Server sends Leave Request to a Channel

```
<iq type='set'
    from='hag66@shakespeare.example'
    to='coven@mix.shakespeare.example'
    id='E6E10350-76CF-40C6-B91B-1EA08C332FC7'>
  <leave xmlns='urn:xmpp:mix:core:1'/>
</iq>
```

The MIX channel will then process the leave and respond. The following example is taken from Mediated Information eXchange (MIX) (XEP-0369) [14].

Listing 10: Channel Confirms Leave to User's Server

```
<iq type='result'
    from='coven@mix.shakespeare.example'
    to='hag66@shakespeare.example'
```

[11] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.
[12] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.
[13] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.
[14] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.

```
    id='E6E10350-76CF-40C6-B91B-1EA08C332FC7'>
  <leave xmlns='urn:xmpp:mix:core:1'/>
</iq>
```

After receiving the confirmation that the user has left the MIX channel, the user's server will remove the MIX channel entry from the user's roster and follow other processing as specified below. The user's server will then notify the client with the servers response. This wraps the response from the server with a client-leave, with the 'from' set to the user's bare JID and the 'to' set to the client's full JID. This is illustrated below:

Listing 11: User's Server Confirms Leave to Client

```
<iq type='result'
    from='hag66@shakespeare.example'
    to='hag66@shakespeare.example/UUID-a1j/7533'
    id='E6E10350-76CF-40C6-B91B-1EA08C332FC7'>
  <client-leave xmlns='urn:xmpp:mix:pam:2'>
    <leave xmlns='urn:xmpp:mix:core:1'/>
  </client-leave>
</iq>
```

2.9 Roster Management

As part of the channel joining process, the user's server MUST add the MIX channel to the user's roster. This is done to share the user's presence status with the channel and channel participants subscribing to presence, when the user wishes this presence to be shared. These roster entries also enables the user's client to quickly determine which channels the user has joined. The user's server will need to record those roster entries that are associated with MIX channels in order to correctly handle MIX processing. This roster entry will lead to the user's server correctly sending user's presence from all the user's MIX clients to the MIX channel. Where the user wishes to share presence, the roster subscription is configured with one way presence, as presence is sent to the MIX channel but no presence information about the MIX channel is sent to the user.

The participant's server MUST ensure that only presence information from clients that advertise MIX capability is sent to the MIX channel. So, if a user has two online clients, but only one is MIX capable, then the channel will only receive presence information relating to the MIX client.

If the user does not wish to publish presence information to the channel, the user's server will add the roster entry with mode subscription=none. The roster entry will be present to record that the user has joined the channel, but it will not send presence information to the channel. The user's server MUST do this when the user has chosen Presence preference of 'not share' as specified in Mediated Information eXchange (MIX): JID Hidden Channels. (XEP-0404) [15]. If the user changes the value of the preference, the server MUST modify subscription mode to

[15] XEP-0404: Mediated Information eXchange (MIX): JID Hidden Channels. <https://xmpp.org/extensions/xep-0404.html>.

reflect this.
The user's server MUST remove the user's roster entry when the user leaves the channel.
A channel MAY publish an Avatar following User Avatar (XEP-0084) [16]. A client MAY make use of this information to associate an Avatar with the roster entry for a channel.

2.9.1 Setting User Presence

A user joins a channel over an extended period, and participation in a channel does not generally change when a user clients go online or offline. The user's participation in a channel is reflected by the user's bare JID in the participant node. When a user subscribes to presence as specified in Mediated Information eXchange (MIX): Presence Support. (XEP-0403) [17], all presence messages are sent to this JID. Presence updates are sent out to subscribing participants using standard presence stanzas.
A user MAY share presence information with the channel, for the user's online clients. This is achieved by a roster entry for the channel configured with one way presence, which will cause all presence changes for the user's MIX clients to be sent to the channel. When an XMPP client comes online or changes presence status, this will be communicated by the user to the user's server using broadcast presence. The user's XMPP server is then responsible to share this presence information to each entry in the roster and in particular to each MIX channel in the roster.
A user setting status is now used as an example. Unlike in Multi-User Chat (XEP-0045) [18] where coming online is a special action, coming online in MIX is implicit when presence status is set. Going offline is a achieved by setting presence status to unavailable, which removes the client full JID entry from the presence node. When a user sets a presence status, the user's server sends updated presence to the MIX channel, and the MIX service then publishes the user's availability to the presence node. If there is not an item named by the full JID of the client with updated presence status, this item is created. The sequence is shown in the following examples, starting with a client setting presences status on the connected server.

Listing 12: Client Sets Presence Status on Server

```
<presence xmlns='jabber:client' from='hag66@shakespeare.example/UUID-
    a1j/7533'>
  <show>dnd</show>
  <status>Making a Brew</status>
</presence>
```

The server then sends the presence information to roster entries. The following example then shows the presence message from the client's server to the MIX channel. The presence is then handled as specified in Mediated Information eXchange (MIX) (XEP-0369) [19].

[16] XEP-0084: User Avatar <https://xmpp.org/extensions/xep-0084.html>.
[17] XEP-0403: Mediated Information eXchange (MIX): Presence Support. <https://xmpp.org/extensions/xep-0403.html>.
[18] XEP-0045: Multi-User Chat <https://xmpp.org/extensions/xep-0045.html>.
[19] XEP-0369: Mediated Information eXchange (MIX) <https://xmpp.org/extensions/xep-0369.html>.

Listing 13: Server sends Presence Status to MIX Channel

```
<presence  from='hag66@shakespeare.example/UUID-a1j/7533'
           to='coven@mix.shakespeare.example'>
  <show>dnd</show>
  <status>Making a Brew</status>
</presence>
```

2.9.2 Receiving User Presence

A MIX channel will send out presence information to participants that subscribe to the presence node, as specified in Mediated Information eXchange (MIX): Presence Support. (XEP-0403) [20]. An example is shown below:

Listing 14: User's Server Receives Presence

```
<presence from='123435#coven@mix.shakespeare.example/UUID-a1j/7533'
          to='hag99@shakespeare.example'
          id='77E07BB0-55CF-4BD4-890E-3F7C0E686BBD'>
  <mix xmlns='urn:xmpp:presence:0'>
     <jid>hecate@shakespeare.example/UUID-x4r/2491</jid>
     <nick>thirdwitch</nick>
  </mix>
  <show>dnd</show>
  <status>Making a Brew</status>
</presence>
```

The user's server will then pass this on to all online clients, with 'from' unchanged and 'to' set to the client receiving presence. An example is shown below:

Listing 15: User's Server Sends Presence to Client

```
<presence from='123435#coven@mix.shakespeare.example/UUID-a1j/7533'
          to='hag99@shakespeare.example/UUID-rrr/1234'
          id='77E07BB0-55CF-4BD4-890E-3F7C0E686BBD'>
  <mix xmlns='urn:xmpp:presence:0'>
     <jid>hecate@shakespeare.example/UUID-x4r/2491</jid>
     <nick>thirdwitch</nick>
  </mix>
  <show>dnd</show>
  <status>Making a Brew</status>
</presence>
```

[20] XEP-0403: Mediated Information eXchange (MIX): Presence Support. <https://xmpp.org/extensions/xep-0403.html>.

The user's local server will receive a flow of all presence updates for the user. It will pass this presence information on to all online clients. This ensures that an online client is kept updated with presence. When a client goes offline, it will cease getting presence updates. Presence updates will continue to flow to the user's local server, and so the local server is able maintain up to date presence state for the channel, even when there are no online clients. When a user had no online clients the user's server MAY continue to maintain MIX presence status for the user or MAY discard inbound MIX presence information.

2.9.3 Client Coming Online and Obtaining Presence from the Local Server

When the client comes online, it will activate use of the MIX. When the user's server has been maintaining MIX presence, it will then send full presence status to the client using standard presence messages. This will enable the client to update presence information for the subscribed MIX channels. Note that this does not need any interaction with MIX servers.
There are two situations where a MIX participant's server will need to get presence status from the channel, before it can send presence to the client. The first time is when a user joins the channel as a participant and subscribes to presence. Upon this subscription the MIX channel will send to the participant's server (using the user's bare JID) presence for all of the items in the presence node using standard presence stanzas. This will give the participant's full current presence for the channel.
The second scenario is when the MIX participant's server needs to load or refresh presence status for a channel. This will be needed when the participant's server is started or when the server chooses to not maintain presence for a user when all clients go offline. This MIX participant's server requests presence update by sending a directed presence stanza to the MIX channel from the user's bare JID. The MIX channel can distinguish this from a presence update, which will always be sent from the clients full JID. This will cause the MIX channel to send a full presence update for the channel.

2.9.4 Going Offline

When a client goes offline, this presence update is sent by the client's server to the MIX channel. From the client perspective, this is the same as any other presence change. Going online and offline will simply be presence updates.

Listing 16: Client Goes Offline and Sends Presence to Channel

```
<presence type='unavailable'
          from='hag66@shakespeare.example/UUID-a1j/7533'
          to='coven@mix.shakespeare.example'/>
```

It is desirable to prevent clients from going offline briefly and then coming back online again, as this will lead to "flapping presence". The RECOMMENDED approach to achieve this is use

of Stream Management (XEP-0198) [21] to maintain an XMPP client connection in the event of short term TCP failure.

2.10 MIX Roster Item Capability Sharing

It is useful for a MIX client to know which roster members are MIX channels, as this will facilitate convenient presentation of subscribed MIX channels to the user. A MIX client MAY request that the server returns this additional information that annotates roster elements with MIX capability. The server MUST return the additional information. The request is made by extending the standard roster get request by adding a child element <annotate/> to the <query/> element. The <annotate/> element is qualified by the 'urn:xmpp:mix:roster:0' namespace.

Listing 17: Roster Get Requesting MIX Information

```
<iq from='juliet@example.com/balcony'
    id='bv1bs71f'
    type='get'>
  <query xmlns='jabber:iq:roster'>
    <annotate xmlns='urn:xmpp:mix:roster:0'/>
  </query>
</iq>
```

A standard roster item is encoded as follows.

Listing 18: Standard Roster Item Encoding

```
<item jid='romeo@example.net'/>
```

MIX channels in the roster information returned in response to a request for this additional MIX information MUST have an element <channel/> qualified by the 'urn:xmpp:mix:roster:0' namespace included in the roster item. The <channel/> element MUST also include a 'participant-id' attribute that is the stable ID of the client. This facilitates the client to match messages that reference this stable ID. A MIX extended roster item is shown in the following example.

Listing 19: Roster Item Encoding of a MIX Channel

```
<item jid='balcony@example.net'>
  <channel xmlns='urn:xmpp:mix:roster:0' participant-id='123456'/>
</item>
```

Once a client has made an IQ request to return additional MIX information, the server MUST return this information on all subsequent roster updates that are sent by the server

[21] XEP-0198: Stream Management <https://xmpp.org/extensions/xep-0198.html>.

to the client. The server MUST NOT send additional MIX information when this has not been explicitly requested. It is anticipated that a client will fetch the roster after connection has been established and will set its preference for this MIX capability information at that time.

2.11 MAM Archive Support

Archive of MIX channel messages MAY be performed by the participant's server. When this is done, the capability is advertized to MIX clients using the 'urn:xmpp:mix:pam:2#archive' feature. If archive is provided it MUST always be used, so that where a message is sent to the participant's server and discarded because there are no active clients, it will still be archived. This means that when archiving is provided, the messages will be available in the local archive and can be picked up by clients when they come online.

2.12 Blocking Considerations

This section describes an issue that needs to be addressed by servers that provide blocking capabilities based on JID. Messages distributed by a MIX channel come from JIDs containing the bare JID of the channel. For presence stanzas (specified in MIX-PRESENCE), IQ stanza relay (specified in MIX-PRESENCE), and private messages (specified in MIX-ANON) use an encoded JID, where the local part of the bare JID contains both the channel name and the senders Stable Participant ID, for example '123435#coven@mix.shakespeare.example'. A server implementing blocking and MIX-PAM needs to recognize this encoding, to prevent blocking these messages when this is not desired.

3 Internationalization Considerations

See considerations in Mediated Information eXchange (MIX) (XEP-0369) [22].

4 Security Considerations

See considerations in Mediated Information eXchange (MIX) (XEP-0369) [23].
When converting a 1:1 conversation to a channel there is potential to expose sensitive information and to present invalid information.

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

5 IANA Considerations

None.

6 XMPP Registrar Considerations

The urn:xmpp:mix namespace needs to be registered.

7 XML Schema

To be supplied when MIX progresses to proposed standard.

8 Acknowledgements

See Mediated Information eXchange (MIX) (XEP-0369) [24] for a list of contributors to the MIX Family of specifications.

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