XEP-0430: Inbox

Dave Cridland
mailto:dave@hellopando.com
xmpp:dwd@dave.cridland.net

2020-02-03
Version 0.2.0

Status	Type	Short Name
Experimental	Standards Track	inbox

This specification proposes a mechanism by which clients can find a list of ongoing conversations and their state.

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

When initially run, a messaging client typically shows some list of contacts and chatrooms, and whether any new messages are present in each.
The current mechanism for achieving this UX involves a complete synchronization of the server-side archive, and is both time-consuming and bandwidth-intensive. This specification proposes a solution to directly obtain such data from the server.
Moreover, the information gathered by the server to support this can be used in support of mobile push notifications.

1.1 Terminology

Nomenclature used for instant messages versus ancillary messages will need to be adjusted to make it consistent with Message Fastening (XEP-0422) [1] et al.

2 Overview

2.1 Discovering Support

Support for this protocol is advertised by the Service Discovery protocol defined in Service Discovery (XEP-0030) [2] using a feature of urn:xmpp:inbox:1.

2.2 The Inbox

The Inbox consists semantically of a list of conversations in order of last activity. Each conversation is identified by a jid - for group chats this would be the chatroom, and for individual contacts this would be their bare jid.
Each Inbox entry includes a count of messages considered new, the last MAM stanza-id relating to this conversation, and the last MAM result for this conversation, as defined by Message Archive Management (XEP-0313) [3]. In addition, a client-controlled boolean marker can be used to indicate a manual "set unread" state.
Finding more messages from this conversation can be achieved via a MAM query using with to specify the conversation required.

[1] XEP-0422: Message Fastening <https://xmpp.org/extensions/xep-0422.html>.
[2] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.
[3] XEP-0313: Message Archive Management <https://xmpp.org/extensions/xep-0313.html>.

3 Protocol Elements

3.1 Querying

An <iq/> of type ”get” is used, containing a single element <inbox/>, containing an optional RSM filter as specified by Result Set Management (XEP-0059) [4]. This will typically be sent only to the user's own bare jid. If a client requests the inbox without RSM, the server MAY limit the number of conversations arbitrarily by either time or number. This element has a number of attributes:

- unread-only - Defaults to false. If true, the server will list only conversations with at least one unread message.
- messages - Defaults to true. If true, the server includes the last message; if false, this is elided.

The server responds with a sequence of <message/> stanzas, each containing an <entry/> element qualified by the urn:xmpp:inbox:1 namespace with a number of attributes:

- jid - contains the Jid of the conversation for this entry.
- unread - contains a count of messages which are deemed to be unread by the server. Clients may use this to indicate unread messages to the user.
- id - contains the last id in the MAM archive for this conversation. Clients may use this as a marker to fetch additional messages (or collated fastenings, see MAM Fastening Collation (XEP-0427) [5]) about the conversation via MAM.

If the messages attribute is missing or set to true, the <entry/> element is followed by the latest instant message, if any, which is encapsulated as a <result/> element as defined by Message Archive Management (XEP-0313) [6]. This contains collated fastenings if supported by the server.
After all entries required have been returned, the server then responds with an <iq/> result containing a <fin/> element qualified by urn:xmpp:inbox:1. This contains the RSM data, a total count of conversation entries within the inbox, a count of conversations with unread messages, and a total count of unread messages.

4 Unread Messages

Servers MUST track which instant messages sent to clients remain unread.

[4] XEP-0059: Result Set Management <https://xmpp.org/extensions/xep-0059.html>.
[5] XEP-0427: MAM Fastening Collation <https://xmpp.org/extensions/xep-0427.html>.
[6] XEP-0313: Message Archive Management <https://xmpp.org/extensions/xep-0313.html>.

- An instant message is always read if it is followed by an instant message which is read.
- An instant message always starts unread.
- A Chat Marker (see Chat Markers (XEP-0333) [7]) of "displayed" or "acknowledged" causes the message to be read (and also causes all prior messages to be read by implication).
- A Message Receipt (see Message Delivery Receipts (XEP-0184) [8]) does not cause messages to be considered unread.
- Unmarking a conversation always sets the unread counter to zero, and by implication sets all messages to be read.

5 Examples

Let us assume a user has only three jids they have exchanged messages with. Asking for their inbox is simple:

```
<iq type='get' id='iq_stanza_id'>
      <inbox xmlns='urn:xmpp:inbox:1'/>
    </iq>
```

The server responds with a list of conversations:

```
<message>
      <entry xmlns='urn:xmpp:inbox:1' unread='5' jid='
          first_contact@example.net' id='uuid-1'/>
      <result xmlns='urn:xmpp:mam:2' queryid='iq_stanza_id' id='uuid-1
          '>
        <forwarded xmlns='urn:xmpp:forward:0'>
          <message xmlns='jabber:client' from='first_contact@example.
              net' to='user@example.org' type='chat'>
            <body>Greetings from Alpha Centauri!</body>
          </message>
        </forwarded>
      </result>
    </message>

    <message>
      <entry xmlns='urn:xmpp:inbox:1' unread='0' jid='
          second_contact@example.net' id='uuid-5'/>
      <result xmlns='urn:xmpp:mam:2' queryid='iq_stanza_id' id='uuid-5
          '>
        <forwarded xmlns='urn:xmpp:forward:0'>
```

[7] XEP-0333: Chat Markers <https://xmpp.org/extensions/xep-0333.html>.
[8] XEP-0184: Message Delivery Receipts <https://xmpp.org/extensions/xep-0184.html>.

```
          <message xmlns='jabber:client' from='second_contact@example.
              net' to='user@example.org' type='chat'>
            <body>Greetings from Mars!</body>
          </message>
        </forwarded>
      </result>
    </message>

    <message>
      <entry xmlns='urn:xmpp:inbox:1' unread='1' jid='
          third_contact@example.net' id='uuid-8'/>
      <result xmlns='urn:xmpp:mam:2' queryid='iq_stanza_id' id='uuid-8
          '>
        <forwarded xmlns='urn:xmpp:forward:0'>
          <message xmlns='jabber:client' from='third_contact@example.
              net' to='user@example.org' type='chat'>
            <body>Greetings from Somewhere Else!</body>
          </message>
        </forwarded>
      </result>
    </message>
```

If the id of a conversation has changed, a client might fetch the missing messages and metadata by requesting the MAM archive with the jid of the entry, and after the previous known id for the conversation.
After the list of conversations, the server completes its response with a the reply to the original IQ.

```
<iq type='result' id='iq_stanza_id'>
        <fin xmlns='urn:xmpp:inbox:1' total='3' unread='2' all-unread=
            '6'>
          <!-{}- RSM -{}->
        </fin>
```

6 Schema

TODO - Hopefully roughly given by the examples.

7 Security Considerations

TODO

8 IANA Considerations

This XEP requires no interaction with the Internet Assigned Numbers Authority (IANA) [9].

9 XMPP Registrar Considerations

None.

10 Acknowledgements

The author notes that this protocol is heavily based on the mod_inbox system of MongooseIM. In addition, Kevin Smith and several others at the XMPP Summit 24 provided useful feedback which has shaped this specification.

[9] 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/>.