XEP-0184: Message Delivery Receipts

Peter Saint-Andre
mailto:xsf@stpeter.im
xmpp:peter@jabber.org
http://stpeter.im/

Joe Hildebrand
mailto:jhildebr@cisco.com
xmpp:hildjj@jabber.org

2011-03-01
Version 1.2

Status	Type	Short Name
Draft	Standards Track	receipts

This specification defines an XMPP protocol extension for message delivery receipts, whereby the sender of a message can request notification that the message has been delivered to a client controlled by the intended recipient.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

While Advanced Message Processing (XEP-0079) [1] provides message acknowledgements at the server level, it does not extend that model all the way to the client. [2] However, sometimes client-level acknowledgements are needed, for example to provide ”receipts”. This document defines a mechanism for XMPP message delivery receipts, which are functionally equivalent to the ”delivered” or ”displayed” event in Message Events (XEP-0022) [3], which this specification in part obsoletes.
Note well that this specification does not distinguish between delivery and display, as was done in the message events protocol, in part because no implementations of XEP-0022 made that distinction. However, in the absence of such a distinction, readers need to understand that this protocol can provide only a notification that a message has been received at a client, i.e. delivered to a client, not that the message has been actively read or understood by a human user (if any). Therefore this extension is functionally equivalent to an Advanced Message Processing rule of ”receipt”, although it uses a dedicated namespace to simplify processing by clients and intermediate routers.

2 Requirements

This document addresses the following requirements:

1. Enable a sender to request notification that an XMPP message stanza has been received (i.e., delivered to a client, but not necessarily read or understood by a human user, if any).
2. Enable a recipient to provide message delivery receipts if desired.

3 Terminology

The term ”content message” refers to the stanza for which the original sender would like to receive a receipt.
The term ”ack message” refers to the stanza by which the recipient acknowledges receipt of the content message at a client (i.e., delivery to a client).

[1] XEP-0079: Advanced Message Processing <https://xmpp.org/extensions/xep-0079.html>.
[2] Naturally, message delivery receipts can be combined with the rules specified in Advanced Message Processing for more complete reporting.
[3] XEP-0022: Message Events <https://xmpp.org/extensions/xep-0022.html>.

4 What This Protocol Provides

This document defines a protocol that enables a sender to ask the recipient to acknowledge receipt of a content message by returning an ack message. Although the return of an ack message lets the sender know that the content message has been delivered to a client controlled by the intended recipient, there are many reasons why the sender might not receive an ack message immediately or at all, including but not limited to the following:

- The sender addressed the content message to the recipient's bare JID <localpart@domain.tld> and therefore does not know if the recipient even supports the Message Delivery Receipts protocol.
- The sender has not bothered to determine whether the recipient supports the Message Delivery Receipts protocol.
- The recipient (or the particular intended resource to which the sender addressed the content message) does not in fact support the Message Delivery Receipts protocol.
- The intended resource supports the Message Delivery Receipts protocol but the recipient's server delivers the content message to another resource that does not support the Message Delivery Receipts protocol.
- The recipient's client receives the content message but experiences a malfunction before generating an ack message.
- The recipient returns an ack message but the ack message is lost on the way back from the recipient to the sender (e.g., because of connectivity issues or software failures at any hop).
- The recipient simply does not wish to return a receipt for the content message.

Because of these significant limitations, this protocol does not provide complete or even partial reliability or guaranteed delivery. Therefore, the sender SHOULD NOT impute any meaning to the fact that it has not received an ack message, unless it has established with the recipient that receipt requests will be honored; however, methods for doing so are out of scope for this specification and it is NOT RECOMMENDED to take any particular action (such as resending the content message) without such methods. [4]

Because it is possible for a given content message to be delivered to multiple XMPP resources controlled by the recipient, the sender of the content message needs to be prepared to receive multiple ack messages.

Finally, this protocol does not enable the sender to know that the intended recipient has read the message or understood the message (if the intended recipient is a human being), that the intended recipient has processed the message (if the intended recipient is a bot or other

[4] This protocol merely provides a building block that could be used in conjunction with other protocols to asymptotically approach the eventual goal of messaging reliability and guaranteed delivery.

automated system), that an end user client has presented the message to a human user (if any), etc. This protocol provides delivery receipts only, not notifications about presentation, processing, reading, understanding, or any other action related to a message other than delivery to a client of some kind.

5 When to Request Receipts

A sender *could* request receipts on any non-error content message (chat, groupchat, headline, or normal) no matter if the recipient's address is a bare JID <localpart@domain.tld> or a full JID <localpart@domain.tld/resource>. Whether it is *appropriate* or *advisable* to do so it another question. This section provides recommendations about when and when not to request receipts, and what results to expect in each scenario.

5.1 Bare JID

If the sender knows only the recipient's bare JID, it cannot determine (via Service Discovery (XEP-0030) [5] or Entity Capabilities (XEP-0115) [6]) whether the intended recipient supports the Message Delivery Receipts protocol. In this case, the sender MAY request a receipt when sending a content message of type "chat", "headline", or "normal" to the recipient's bare JID. However, the sender MUST NOT depend on receiving an ack message in reply.

5.2 Full JID

If the sender knows a full JID for the recipient (e.g., via presence), it SHOULD attempt to determine (via service disco or entity capabilities) whether the client at that full JID supports the Message Delivery Receipts protocol before attempting to request receipts.
If the sender determines that the recipient's client does not support the Message Delivery Receipts protocol then it SHOULD NOT request a receipt when sending a content message to that full JID and MUST NOT depend on receiving an ack message.
If the sender determines that the recipient's client supports the Message Delivery Receipts protocol then it MAY request a receipt when sending a content message of type "chat", "headline", or "normal" to that full JID. However, even in this case the sender SHOULD NOT depend on receiving an ack message.

[5] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.
[6] XEP-0115: Entity Capabilities <https://xmpp.org/extensions/xep-0115.html>.

5.3 Groupchat

It is NOT RECOMMENDED to request a receipt when sending a content message of type "groupchat" in a Multi-User Chat (XEP-0045) [7] room because the logic for determining when a content message is truly "received" by all of the room occupants is complex, and because the sender would receive one ack message from each occupant of the room, thus significantly increasing the number of stanzas sent through the room.

5.4 Ack Messages

To prevent looping, an entity MUST NOT include a receipt request (i.e., the <request/> element) in an ack message (i.e., a message stanza that includes the <received/> element).

5.5 Archived Messages

An entity MUST NOT send an ack message when a user views messages that have been archived or stored on the client or the server (e.g., via Message Archiving (XEP-0136) [8]), only when first receiving the message.

6 Determining Support

If an entity supports the Message Delivery Receipts protocol, it MUST report that by including a Service Discovery (XEP-0030) [9] feature of "urn:xmpp:receipts" in response to disco#info requests:

Listing 1: Initial Service Discovery information request

```
<iq from='northumberland@shakespeare.lit/westminster'
    id='disco1'
    to='kingrichard@royalty.england.lit/throne'
    type='get'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 2: Service Discovery information response

```
<iq from='kingrichard@royalty.england.lit/throne'
    id='disco1'
    to='northumberland@shakespeare.lit/westminster'
    type='result'>
```

[7] XEP-0045: Multi-User Chat <https://xmpp.org/extensions/xep-0045.html>.
[8] XEP-0136: Message Archiving <https://xmpp.org/extensions/xep-0136.html>.
[9] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.

```
  <query xmlns='http://jabber.org/protocol/disco#info'>
    <feature var='urn:xmpp:receipts'/>
  </query>
</iq>
```

Support can also be determined via Entity Capabilities (XEP-0115) [10], a.k.a. "caps".

7 Protocol Format

In order to make it possible for senders to request and for recipients to generate message delivery receipts, we define a dedicated protocol extension qualified by the 'urn:xmpp:receipts' namespace.
There are two allowable elements in this namespace:

- <request/> -- included in a content message by a sending entity that wishes to know if the content message has been received, i.e., delivered to a client controlled by the intended recipient.
- <received/> -- included in an ack message by a receiving entity that wishes to inform the sending entity that the content message has been received, i.e., delivered to a client controlled by the intended recipient.

Specifically, the receiving entity shall return an ack message containing the <received/> extension if the content message has been delivered to a client controlled by the intended recipient. In general, a client will return a receipt only if the client has processed the content message (e.g., if the client has presented the content message to a human user or has completed any automated processing of the content message, such as generation of an error response if the application determines that the content message cannot be handled). However, the Message Delivery Receipts protocol does not provide notification that a human user has read or understood the content message, that an automated system has completed processed or acted upon the message, etc.
The following is an example of a content message that includes a request for return receipt.

Listing 3: A content message with receipt requested

```
<message
    from='northumberland@shakespeare.lit/westminster'
    id='richard2-4.1.247'
    to='kingrichard@royalty.england.lit/throne'>
  <body>My lord, dispatch; read o'er␣these␣articles.</body>
␣␣<request␣xmlns='urn:xmpp:receipts'/>
</message>
```

[10] XEP-0115: Entity Capabilities <https://xmpp.org/extensions/xep-0115.html>.

Note: A sender MUST include an 'id' attribute on every content message that requests a receipt, so that the sender can properly track ack messages.
The recipient shall generate an ack message if and only if:

1. it supports the Message Delivery Receipts protocol; and
2. it is configured to return receipts, either globally or for this recipient.

Otherwise it MUST NOT return a receipt and SHOULD NOT return an error.

Listing 4: A message delivery receipt

```
<message
    from='kingrichard@royalty.england.lit/throne'
    id='bi29sg183b4v'
    to='northumberland@shakespeare.lit/westminster'>
  <received xmlns='urn:xmpp:receipts' id='richard2-4.1.247'/>
</message>
```

When the recipient sends an ack message, it SHOULD ensure that the message stanza contains only one child element, namely the <received/> element qualified by the 'urn:xmpp:receipts' namespace. In addition, it SHOULD include an 'id' attribute that echoes the 'id' attribute of the content message. Naturally, intermediate entities might add other extension elements to the message when routing or delivering the receipt message, e.g., a <delay/> element as specified in Delayed Delivery (XEP-0203) [11].

8 Security Considerations

It is possible for a recipient to leak its presence when returning message delivery receipts; therefore, a recipient SHOULD NOT return message delivery receipts to senders who are not otherwise authorized to view its presence.

9 IANA Considerations

No interaction with the Internet Assigned Numbers Authority (IANA) [12] is necessary as a result of this document.

[11] XEP-0203: Delayed Delivery <https://xmpp.org/extensions/xep-0203.html>.
[12] 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/>.

10 XMPP Registrar Considerations

10.1 Protocol Namespaces

The XMPP Registrar [13] includes "urn:xmpp:receipts" in its registry of protocol namespaces (see <https://xmpp.org/registrar/namespaces.html>).

11 XML Schema

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:xmpp:receipts'
    xmlns='urn:xmpp:receipts'
    elementFormDefault='qualified'>

  <xs:annotation>
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-0184: http://xmpp.org/extensions/xep-0184.html
    </xs:documentation>
  </xs:annotation>

  <xs:element name='received'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='empty'>
          <xs:attribute name='id' type='xs:string' use='optional'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name='request' type='empty'/>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```

[13] The 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/>.

12 Acknowledgements

Thanks to Steven te Brinke, Bruce Campbell, Joe Kemp, Kevin Smith, Remko Tronçon, Matthew Wild, and Kurt Zeilenga for their input.