XEP-0231: Bits of Binary

Peter Saint-Andre
mailto:stpeter@jabber.org
xmpp:stpeter@jabber.org
https://stpeter.im/

Pavel Šimerda
xmpp:pavlix@pavlix.net
http://www.pavlix.net/

2008-09-03
Version 1.0

Status	Type	Short Name
Draft	Standards Track	bob

This specification defines an XMPP protocol extension for including or referring to small bits of binary data in an XML stanza.

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 <http://xmpp.org/about-xmpp/xsf/xsf-ipr-policy/> or obtained by writing to XMPP Standards Foundation, 1899 Wynkoop Street, Suite 600, Denver, CO 80202 USA).

Contents

1 Introduction

Sometimes it is desirable to include a small bit of binary data in an XMPP stanza. Typical use cases might be to include icon or emoticon in a message, a thumbnail in a file transfer request, a rasterized image in a whiteboarding session, or a small bit of media in a data form. Currently, there is no lightweight method for including such data in an XMPP stanza, since existing methods (e.g., In-Band Bytestreams [1]) are designed for larger blobs of data and therefore require some form of negotiation (e.g., via SI File Transfer [2] or Jingle File Transfer [3]). This document specifies just such a lightweight method. The key building blocks are:

1. A Content-ID ("cid") that uniquely identifies the data.
2. A <data/> element (similar to the data: URL scheme defined in RFC 2397 [4]) that enables the sender and recipient to exchange the data identified by the cid.

2 Protocol

2.1 Data Exchange

The RECOMMENDED approach is for the sender to include the cid when communicating with the recipient. The recipient SHOULD then check its cache of data to determine if the data identified by that cid is cached. If the data is cached, the recipient would then load its cached data. If the data is not cached, the recipient would then retrieve the data by sending an IQ-get to the sender (or potentially some other entity) containing an empty <data/> element whose 'cid' attribute specifies the data to be retrieved, to which the sender would reply with an IQ-result containing a <data/> element whose XML character data provides the binary data.
The <data/> element MUST be used only to encapsulate small bits of binary data and MUST NOT be used for large data transfers. Naturally the definitions of "small" and "large" are rather loose. In general, the data SHOULD NOT be more than 8 kilobytes, and dedicated file transfer methods (e.g., SOCKS5 Bytestreams [5] or In-Band Bytestreams [6]) SHOULD be used for exchanging blobs of data larger than 8 kilobytes. However, implementations or deployments MAY impose their own limits.
If the data to be shared is particularly small (e.g., less than 1k), then the sender MAY send it directly by including a <data/> element directly in a <message/>, <presence/>, or <iq/> stanza. The following rules apply:

1. When the <data/> element is directly included in an XMPP <message/> or <presence/> stanza, it SHOULD be a first-level child of the stanza.

[1] XEP-0047: In-Band Bytestreams <http://xmpp.org/extensions/xep-0047.html>.
[2] XEP-0096: SI File Transfer <http://xmpp.org/extensions/xep-0096.html>.
[3] XEP-0234: Jingle File Transfer <http://xmpp.org/extensions/xep-0234.html>.
[4] RFC 2397: The data: URL scheme <http://tools.ietf.org/html/rfc2397>.
[5] XEP-0065: SOCKS5 Bytestreams <http://xmpp.org/extensions/xep-0065.html>.
[6] XEP-0047: In-Band Bytestreams <http://xmpp.org/extensions/xep-0047.html>.

2. When the <data/> element is directly included in an XMPP <iq/> stanza, it MUST be a child of the appropriate first-level child (since the IQ stanza must not include more than one first-level child).

3. When the <data/> element is used to retrieve the data from the sender as described under Retrieving Uncached Data, it MUST be a first-level child of the stanza.

2.2 Referencing Data

The sender can refer to data that it hosts by including a cid in the data it sends. The following example shows how to include the cid in XHTML-IM [7] but any appropriate format can be used, such as Data Forms Media Element [8].

Listing 1: An XHTML-IM message with a cid

```
<message from='ladymacbeth@shakespeare.lit/castle'
         to='macbeth@chat.shakespeare.lit'
         type='groupchat'>
  <body>Yet here's a spot.</body>
  <html xmlns='http://jabber.org/protocol/xhtml-im'>
    <body xmlns='http://www.w3.org/1999/xhtml'>
      <p>
        Yet here's a spot.
        <img alt='A spot'
             src='cid:sha1+8
                  f35fef110ffc5df08d579a50083ff9308fb6242@bob.xmpp.org'
                  />
      </p>
    </body>
  </html>
</message>
```

The recipient can then retrieve the data from the sender as described in the next section.

2.3 Retrieving Uncached Data

Data is requested and transferred using the XMPP <iq/> stanza type by making reference to the cid. In particular, the recipient requests the binary data by sending an IQ-get containing an empty <data/> element with a 'cid' attribute that matches the cid URI previously communicated.

Listing 2: Requesting data

```
<iq from='doctor@shakespeare.lit/pda'
```

[7] XEP-0071: XHTML-IM <http://xmpp.org/extensions/xep-0071.html>.
[8] XEP-0221: Data Forms Media Element <http://xmpp.org/extensions/xep-0221.html>.

```
    id='get-data-1'
    to='ladymacbeth@shakespeare.lit/castle'
    type='get'>
  <data xmlns='urn:xmpp:bob'
        cid='sha1+8f35fef110ffc5df08d579a50083ff9308fb6242@bob.xmpp.
           org'/>
</iq>
```

The recipient then would either return an error (e.g., <item-not-found/> if it does not have data matching the Content-ID) or return the data.

Listing 3: Returning data

```
<iq from='ladymacbeth@shakespeare.lit/castle'
    id='get-data-1'
    to='doctor@shakespeare.lit/pda'
    type='result'>
  <data xmlns='urn:xmpp:bob'
        cid='sha1+8f35fef110ffc5df08d579a50083ff9308fb6242@bob.xmpp.
           org'
        max-age='86400'
        type='image/png'>
    iVBORw0KGgoAAAANSUhEUgAAAAoAAAAKCAYAAACNMs+9AAAABGdBTUEAALGP
    C/xhBQAAAAlwSFlzAAALEwAACxMBAJqcGAAAAAd0SU1FB9YGARc5KB0XV+IA
    AAAddEVYdENvbW1lbnQAQ3JlYXRlZCB3aXRoIFRoZSBHSU1Q72QlbgAAAF1J
    REFUGNO9zL0NglAAxPEfdLTs4BZM4DIO4C7OwQg2JoQ9LE1exdlYvBBeZ7jq
    ch9//q1uH4TLzw4d6+ErXMMcXuHWxId3KOETnnXXV6MJpcq2MLaI97CER3N0
    vr4MkhoXe0rZigAAAABJRU5ErkJggg==
  </data>
</iq>
```

2.4 Caching Data

It is RECOMMENDED for the recipient to cache data; however, the recipient MAY opt not to cache data, for example because it runs on a device that does not have sufficient space for data storage.

The default behavior is for the recipient to cache the data only for the life of the entity's application session (not a client's presence session with the server or the controlling user's communication session with the contact from whom the user received the data); that is, the recipient would clear the cache when the application is terminated or restarted.

As a hint regarding the suggested period for caching the data, the sender MAY include a 'max-age' attribute whenever it sends a <data/> element. The meaning of the 'max-age' attribute exactly matches that of the Max-Age attribute from RFC 2965.

If it is not suggested to cache the data (e.g., because it is ephemeral), the value of the 'max-age' attribute MUST be "0" (the number zero).

A recipient SHOULD cache data based on the hash of the data as encapsulated in the cid.

However, if a hash cannot be extracted from the cid, if the recipient does not support the hashing algorithm used, or the recipient does not support hashes, then the recipient SHOULD cache based on the JID of the sender.

2.5 Format of the <data/> Element

To exchange binary data, the data is encapsulated as the XML character data of a <data/> element qualified by the 'urn:xmpp:bob' namespace, where the data MUST be encoded as Base64 in accordance with Section 4 of RFC 4648 [9] (note: the Base64 output MUST NOT include whitespace and MUST set the number of pad bits to zero).
The following attributes are defined for the <data/> element.

Attribute	Description	Inclusion
cid	A Content-ID that can be mapped to a cid: URL as specified in RFC 2111 RFC 2111: Content-ID and Message-ID Uniform Resource Locators <http://tools.ietf.org/html/rfc2111>.. The 'cid' value SHOULD be of the form algo+hash@bob.xmpp.org, where the "algo" is the hashing algorithm used (e.g., "sha1" for the SHA-1 algorithm as specified in RFC 3174 RFC 3174: US Secure Hash Algorithm 1 (SHA1) <http://tools.ietf.org/html/rfc3174>.) and the "hash" is the hex output of the algorithm applied to the binary data itself.	REQUIRED
max-age	A suggestion regarding how long (in seconds) to cache the data; the meaning matches the Max-Age attribute from RFC 2965 RFC 2965: HTTP State Management Mechanism <http://tools.ietf.org/html/rfc2965>..	RECOMMENDED

[9] RFC 4648: The Base16, Base32, and Base64 Data Encodings <http://tools.ietf.org/html/rfc4648>.

Attribute	Description	Inclusion
type	The value of the 'type' attribute MUST match the syntax specified in RFC 2045 RFC 2045: Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies <http://tools.ietf.org/html/rfc2045>.. That is, the value MUST include a top-level media type, the "/" character, and a subtype; in addition, it MAY include one or more optional parameters (e.g., the "audio/ogg" MIME type in the example shown below includes a "codecs" parameter as specified in RFC 4281 RFC 4281: The Codecs Parameter for "Bucket" Media Types <http://tools.ietf.org/html/rfc4281>.). The "type/subtype" string SHOULD be registered in the IANA MIME Media Types Registry IANA registry of MIME media types <http://www.iana.org/assignments/media-types>., but MAY be an unregistered or yet-to-be-registered value.	REQUIRED if the <data/> element is non-empty

The following example illustrates the format (line endings are provided for readability only).

Listing 4: Data element format

```
<data xmlns='urn:xmpp:bob'
      cid='sha1+8f35fef110ffc5df08d579a50083ff9308fb6242@bob.xmpp.org'
      max-age='86400'
      type='image/png'>
  iVBORw0KGgoAAAANSUhEUgAAAAoAAAAKCAYAAACNMs+9AAAABGdBTUEAALGP
  C/xhBQAAAAlwSFlzAAALEwAACxMBAJqcGAAAAAd0SU1FB9YGARc5KB0XV+IA
  AAAddEVYdENvbW1lbnQAQ3JlYXRlZCB3aXRoIFRoZSBHSU1Q72QlbgAAAF1J
  REFUGNO9zL0NglAAxPEfdLTs4BZM4DIO4C7OwQg2JoQ9LE1exdlYvBBeZ7jq
  ch9//q1uH4TLzw4d6+ErXMMcXuHWxId3KOETnnXXV6MJpcq2MLaI97CER3N0
  vr4MkhoXe0rZigAAAABJRU5ErkJggg==
</data>
```

3 Determining Support

If an entity supports the protocol specified herein, it MUST advertise that fact by returning a feature of "urn:xmpp:bob" in response to Service Discovery [10] information requests.

Listing 5: Service discovery information request

```
<iq from='doctor@shakespeare.lit/pda'
    id='disco1'
    to='ladymacbeth@shakespeare.lit/castle'
    type='get'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 6: Service discovery information response

```
<iq from='ladymacbeth@shakespeare.lit/castle'
    id='disco1'
    to='doctor@shakespeare.lit/pda'
    type='result'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    ...
    <feature var='urn:xmpp:bob'/>
    ...
  </query>
</iq>
```

In order for an application to determine whether an entity supports this protocol, where possible it SHOULD use the dynamic, presence-based profile of service discovery defined in Entity Capabilities [11]. However, if an application has not received entity capabilities information from an entity, it SHOULD use explicit service discovery instead.

4 Security Considerations

The ability to include arbitrary binary data implies that it is possible to send scripts, applets, images, and executable code, which may be potentially harmful. To reduce the risk of such exposure, an implementation MAY choose to not display or process such data but instead either completely ignore the data, show only the value of the 'alt' attribute, or prompt a human user for approval (either explicitly via user action or implicitly via a list of approved entities from whom the user will accept binary data without per-event approval).

[10] XEP-0030: Service Discovery <http://xmpp.org/extensions/xep-0030.html>.
[11] XEP-0115: Entity Capabilities <http://xmpp.org/extensions/xep-0115.html>.

5 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [12].

6 XMPP Registrar Considerations

6.1 Protocol Namespaces

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

7 XML Schema

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

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

  <xs:annotation>
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-0231: http://www.xmpp.org/extensions/xep-0231.html
    </xs:documentation>
  </xs:annotation>

  <xs:element name='data'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:base64Binary'>
          <xs:attribute name='cid' type='xs:string' use='required'/>
          <xs:attribute name='max-age' type='xs:nonNegativeInteger'
              use='optional'/>
          <xs:attribute name='type' type='xs:string' use='optional'/>
        </xs:extension>
```

[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/>.

[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 <http://xmpp.org/registrar/>.

```
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

8 Acknowledgements

Thanks to Rachel Blackman, Dave Cridland, Zenon Kuder, and Tomasz Sterna for their feedback.