XEP-0132: Presence Obtained via Kinesthetic Excitation (POKE)

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

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

2004-04-01
Version 1.0

Status	Type	Short Name
Active	Humorous	poke

This document defines an XMPP protocol extension that enables probing for presence via physical rather than electronic means.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

XMPP Core [1] and XMPP IM [2] define methods for exchanging information about a person's network availability via the XML <presence/> stanza. In general, such presence information is generated only when a person initiates interaction with a client, although it can be generated programmatically through features such as auto-away. However, sometimes a user is present in the vicinity of a client but is not actively engaged with the client interface. In such circumstances, it would be helpful to have a mechanism that is sometimes referred to as <presence type='probe-irl'/>: the ability to invoke a real-life means of determining the physical presence of the user. This document defines just such a mechanism.

2 Approach

Physical presence is best determined through direct interaction with an object. In this document, our approach is labelled "kinesthetic excitation": some form of physical contact is initiated with the object (in most cases a user), resulting in hard evidence of presence obtained by a sense modality such as sight, touch, or hearing. To ensure reliability, the physical contact MUST impinge upon the object or user to such an extent that it measurably reacts in the form of motion through space (e.g., moving in relation to a visual observation device), generation of an auditory event (e.g., vocalization), and the like. The exact means of excitation and perception are implementation-specific and therefore not specified fully in this document, although suggestions are provided in the Methods section below.

3 Protocol

In the past, some members of the Jabber community have suggested the addition of a new presence type: "probe-irl". However, this has several drawbacks. First, the XMPP specifications (XMPP Core and XMPP IM) approved by the IETF do not allow any values for the 'type' attribute other than those defined in the XML schemas for the 'jabber:client' and 'jabber:server' namespaces. Second, presence probes are handled by a server on behalf of a user and therefore are not routed to clients (which presumably often have the best opportunity for discovering evidence of physical presence); an <iq/> stanza is more appropriate for client-to-client information exchange. Therefore, this document defines a general extension mechanism that can be used in both <presence/> and <iq/> stanzas.

The extension mechanism is encapsulated in a <poke/> element qualified by the 'http://jabber.org/protocol/poke' namespace; this element MAY be included as a direct child of a <presence/> stanza of type "probe" or an <iq/> stanza of type "get" (for a request), "result" (for a successful response), or "error" (for an unsuccessful response).

[1] RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc6120>.
[2] RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence <http://tools.ietf.org/html/rfc6121>.

The requesting entity MAY specify a preferred method of excitation and observation; in general, these methods correspond to particular sense modalities such as sight, touch, and hearing (see the Methods section below).

3.1 Presence

As defined in XMPP IM, presence stanzas of type "probe" are handled on behalf of the target entity by the entity's server. While normally these presence stanzas are generated by the requesting entity's server (e.g., when the requesting entity sends initial presence), the requesting entity itself (or, more precisely, its client) is allowed to generate presence stanzas of type "probe". In this document we make use of this ability to query the target entity's server regarding the entity's physical presence.
In the following example, a star-crossed lover pokes the server of his beloved to determine her physical presence (notice that the value of 'to' address lacks a resource identifier and therefore is a bare JID, not a full JID).

Listing 1: Poking via the server

```
<presence
    type='probe'
    from='romeo@montague.net/orchard'
    to='juliet@capulet.com'
    id='poke1'>
  <poke xmlns='http://jabber.org/protocol/poke'/>
</presence>
```

If the user's server does not support the POKE protocol, it SHOULD ignore the extension and treat the presence stanza as a normal (non-IRL) presence probe. However, the user's server MAY return a "Service Unavailable" error to the requesting entity to inform the requesting entity that IRL probes are not supported (for details regarding error syntax, refer to Error Condition Mappings (XEP-0086) [3]):

Listing 2: Server returns service unavailable error

```
<presence
    type='error'
    from='juliet@capulet.com'
    to='romeo@montague.net/orchard'
    id='poke1'>
  <poke xmlns='http://jabber.org/protocol/poke'/>
  <error code='503' type='cancel'>
    <service-unavailable
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
```

[3] XEP-0086: Error Condition Mappings <https://xmpp.org/extensions/xep-0086.html>.

```
</presence>
```

If the user's server supports the POKE protocol, it MUST first perform appropriate access checks to determine if the requesting entity has permission to view the user's presence (e.g., by checking presence subscriptions and privacy lists). If the user's server determines that the requesting entity is not allowed to learn the user's physical presence information, it MUST return a "Forbidden" error:

Listing 3: Server returns forbidden error

```
<presence
    type='error'
    from='juliet@capulet.com'
    to='romeo@montague.net/orchard'
    id='poke1'>
  <poke xmlns='http://jabber.org/protocol/poke'/>
  <error code='403' type='auth'>
    <forbidden
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</presence>
```

If the requesting entity has permission to discover the user's physical presence, the server SHOULD attempt to determine if the user is physically present. Methods for doing so are implementation-specific and therefore out of scope for this document, but possible mechanisms might include:

1. sending messages to contacts in the user's roster who would be likely to have knowledge of the user's whereabouts (perhaps derived from physical proximity information gleaned from vcard-temp (XEP-0054) [4] or User Geolocation (XEP-0080) [5] data)
2. generating an IQ "get" in the 'poke' namespace to each of the user's connected resources
3. sending specialized commands to each of the user's connected resources using Ad-Hoc Commands (XEP-0050) [6]

If the server determines that the user is physically present in the vicinity of a client, it SHOULD return that information to the requesting entity, including the appropriate resource:

Listing 4: Server returns success

```
<presence
    from='juliet@capulet.com/chamber'
```

[4] XEP-0054: vcard-temp <https://xmpp.org/extensions/xep-0054.html>.
[5] XEP-0080: User Geolocation <https://xmpp.org/extensions/xep-0080.html>.
[6] XEP-0050: Ad-Hoc Commands <https://xmpp.org/extensions/xep-0050.html>.

```
    to='romeo@montague.net/orchard'
    id='poke1'>
  <poke xmlns='http://jabber.org/protocol/poke'/>
</presence>
```

The server SHOULD NOT wait an inordinate amount of time before returning the presence information (e.g., usually not more than two minutes), but the timeout period SHOULD be configurable. If the request times out, the server SHOULD return a "Request Timeout" error to the requesting entity:

Listing 5: Server returns request timeout error

```
<presence
    type='error'
    from='juliet@capulet.com'
    to='romeo@montague.net/orchard'
    id='poke1'>
  <poke xmlns='http://jabber.org/protocol/poke'/>
  <error code='408' type='wait'>
    <remote-server-timeout
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</presence>
```

The server SHOULD NOT return a "Not Found" error unless the user does not exist. If the server determines that the user has died, it MAY return a "Gone" error with appropriate descriptive text, although it SHOULD wait to do so pending notification of next-of-kin; note well that such notification is out of scope for this document (though this seems like a sensible application of the Publish-Subscribe (XEP-0060) [7] protocol):

Listing 6: Server returns gone error

```
<presence
    type='error'
    from='juliet@capulet.com'
    to='romeo@montague.net/orchard'
    id='poke1'>
  <poke xmlns='http://jabber.org/protocol/poke'/>
  <error code='302' type='cancel'>
    <gone xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <text xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'>
      Please accept our condolences: the user you are
      trying to reach has died.
    </text>
  </error>
</presence>
```

[7] XEP-0060: Publish-Subscribe <https://xmpp.org/extensions/xep-0060.html>.

3.2 IQ

If the requesting entity knows at least one resource with which the user is currently connected, it MAY send an IQ to the user's full JID (<user@host/resource>) instead of sending a probe to the user's server.

Listing 7: Poking via the client

```
<iq type='get'
    from='romeo@montague.net/orchard'
    to='juliet@capulet.com/balcony'
    id='poke2'>
  <poke xmlns='http://jabber.org/protocol/poke'
        method='taste'/>
</iq>
```

The same errors as shown above for presence stanzas SHOULD be used by clients responding to IQ stanzas containing POKE protocols (e.g., "Request Timeout" if the user cannot be found in some reasonable period of time), and therefore are not repeated here.
Note that the preceding example includes the optional 'method' attribute. If the target entity does not support the specified method, it MAY return a "Feature Not Implemented" error:

Listing 8: Client returns feature not implemented error

```
<iq type='error'
    from='juliet@capulet.com/balcony'
    to='romeo@montague.net/orchard'
    id='poke1'>
  <poke xmlns='http://jabber.org/protocol/poke'/>
  <error code='501' type='cancel'>
    <feature-not-implemented
        xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

Alternatively, it MAY choose to use some other method that it does implement, in which case it SHOULD specify the method used in the IQ result (this is the recommended behavior).
If the client determines that the user is physically present, it SHOULD return presence to the requesting entity (subject to privacy lists and any other appropriate access controls):

Listing 9: Client returns success

```
<iq type='result'
    from='juliet@capulet.com/balcony'
    to='romeo@montague.net/orchard'
    id='poke1'>
  <poke xmlns='http://jabber.org/protocol/poke'
        method='touch'/>
```

```
</iq>
```

3.3 Methods

The following values of the 'method' attribute are defined and SHOULD be supported by a compliant implementation:

dna The physical presence of the target entity shall be determined by means of DNA testing; for example, the user's client may take a hair or skin sample from the user (not recommended if the testing time is inordinately long).

infrared The physical presence of the target entity shall be determined by means of infrared wavelengths; for example, the user's client may scan the area for the telltale heat signature of the user.

sight The physical presence of the target entity shall be determined by means of sight; for example, the user's client may flash a strobe light that draws the user within the range of visual observation (note: this method is limited to visual wavelengths).

smell The physical presence of the target entity shall be determined by means of olfaction; for example, the user's client may produce a stench known to make the user nervous and sweaty.

sound The physical presence of the target entity shall be determined by means of sound; for example, the user's client may generate a sound so annoying that when the user hears it, he or she reacts vocally in the form of a yell, scream, or imprecation.

taste The physical presence of the target entity shall be determined by means of taste; for example, the user's client may extend a device that licks the user's skin.

touch The physical presence of the target entity shall be determined by means of bodily contact; for example, the user's client may extend a probe that comes into contact with the user's body.

4 Privacy Considerations

Determination of physical presence necessarily involves an invasion of the target entity's "personal space". The XMPP Standards Foundation (XSF) [8] shall not be held liable for any use of this protocol. Client implementations MUST enable the user to disable support for this protocol via configuration options.

[8] The XMPP Standards Foundation (XSF) is an independent, non-profit membership organization that develops open extensions to the IETF's Extensible Messaging and Presence Protocol (XMPP). For further information, see <https://xmpp.org/about/xmpp-standards-foundation>.

5 Security Considerations

Responding entities (whether server or client) MUST NOT return physical presence information to requesting entities that are not entitled to discover such information.

6 IANA Considerations

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

7 XMPP Registrar Considerations

7.1 Protocol Namespaces

The XMPP Registrar [10] shall add the 'http://jabber.org/protocol/poke' namespace to its registry of protocol namespaces.

7.2 Method Values

The XMPP Registrar shall maintain a registry of values for the 'method' attribute. The following values shall be added initially:

- dna
- infrared
- sight
- smell
- sound
- taste
- touch

[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/>.

[10] 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/>.

8 XML Schema

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

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='http://jabber.org/protocol/poke'
    xmlns='http://jabber.org/protocol/poke'
    elementFormDefault='qualified'>

  <xs:annotation>
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-0132: http://www.xmpp.org/extensions/xep-0132.html
    </xs:documentation>
  </xs:annotation>

  <xs:element name='poke'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='empty'>
          <xs:attribute name='method'
                        use='optional'
                        type='xs:NCName'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

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

</xs:schema>
```