XEP-0126: Invisibility

Peter Saint-Andre
mailto:peter@andyet.net
xmpp:stpeter@stpeter.im
https://stpeter.im/

2005-08-19
Version 1.1

Status	Type	Short Name
Active	Informational	N/A

This specification defines best practices regarding implementation of invisible presence by means of XMPP privacy lists.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction **1**

2 Requirements **1**

3 Use Cases **1**

3.1 Log In as Globally Invisible 2

3.2 Become Selectively Visible 3

3.2.1 Becoming Visible by JID 3

3.2.2 Becoming Visible by Roster Group 4

3.2.3 Becoming Visible by Subscription Type 4

3.3 Become Globally Visible 5

3.4 Become Selectively Invisible 6

3.4.1 Becoming Invisible by JID 7

3.4.2 Becoming Invisible by Roster Group 8

3.4.3 Becoming Invisible by Subscription Type 9

3.5 Become Globally Invisible 10

4 Implementation Notes **10**

5 Security Considerations **11**

6 IANA Considerations **12**

7 XMPP Registrar Considerations **12**

8 XML Schema **12**

1 Introduction

Several popular instant messaging services implement a feature known as invisibility: the ability to remain online yet appear offline to some or all of one's contacts. A number of Jabber servers and clients have also implemented such a feature, using special values of the <presence/> element's 'type' attribute (e.g., <presence type='invisible'/>). Unfortunately, such implementations are not compliant with XMPP Core [1] and XMPP IM [2], which specify that only the 'type' attribute values defined in the XML schema for the 'jabber:client' and 'jabber:server' namespaces are allowed in XMPP (and those values do not include "invisible"). However, RFC 3921 also defines a privacy lists protocol (i.e., the 'jabber:iq:privacy' namespace) that can be used to implement invisibility in an XMPP-compliant manner. This specification documents how to do just that.

2 Requirements

This document addresses the following requirements:

- Enable users to appear visible or invisible to some or all contacts at any time.
- Enable users to selectively change visibility based on roster group, subscription state, or individual JID.
- Do so in an XMPP-compliant manner.

3 Use Cases

This document addresses the following use cases:

1. Log In as Globally Invisible
2. Become Selectively Visible
3. Become Globally Visible
4. Become Selectively Invisible
5. Become Globally Invisible

[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>.

These use cases are defined below in "chronological" order by following a scenario in which a user (1) logs in as invisible, (2) becomes selectively visible to certain contacts, (3) becomes visible to all contacts, (4) becomes selectively invisible to certain contacts, and finally (5) becomes invisible to all contacts.

3.1 Log In as Globally Invisible

If a user wants to log in as invisible, a certain order of events MUST be followed. Specifically, after authenticating but before sending initial presence, the user MUST define (if necessary) and set as active a privacy list that blocks all outbound presence notifications.

Listing 1: User Defines and Sets Global Invisibility Privacy Rule

```
... authentication / session establishment ...

<iq from='bilbo@tolkien.lit/shire' type='set' id='inv1'>
  <query xmlns='jabber:iq:privacy'>
    <list name='invisible'>
      <item action='deny' order='1'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>

<iq from='bilbo@tolkien.lit/shire' type='set' id='active1'>
  <query xmlns='jabber:iq:privacy'>
    <active name='invisible'/>
  </query>
</iq>
```

Naturally, the user could have defined this list during a previous session and could simply set the relevant list as active when logging in, rather than defining the list on login. Both steps are shown here for completeness.
The user may now send initial presence to the server.

Listing 2: User Sends Initial Presence

```
<presence>
  <status>I'm not really here, you understand!</status>
</presence>
```

Even though the user has sent initial presence, that presence information will not be broadcasted to any of the user's contacts, since the active privacy list blocks all outbound presence notifications.

3.2 Become Selectively Visible

Let us now suppose that the user tires of being globally invisible, and decides to become visible to some -- but not all -- contacts (not even famous magic rings possess this feature). The 'jabber:iq:privacy' namespace gives the user three options:

- become visible only to specific JIDs
- become visible only to specific roster groups
- become visible based on subscription state

Examples of these options are shown below.

3.2.1 Becoming Visible by JID

Listing 3: User Defines and Sets Selective Visibility Privacy Rule (by JID)

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='inv2'>
  <query xmlns='jabber:iq:privacy'>
    <list name='visible-to-Frodo'>
      <item type='jid'
            value='frodo@tolkien.lit'
            action='allow'
            order='1'>
        <presence-out/>
      </item>
      <item action='deny' order='2'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>

<iq from='bilbo@tolkien.lit/shire' type='set' id='active2'>
  <query xmlns='jabber:iq:privacy'>
    <active name='visible-to-Frodo'/>
  </query>
</iq>
```

The foregoing privacy list blocks outbound presence notifications to every JID except one. In order to ensure synchronization of presence notifications, the client SHOULD now re-send the user's presence for broadcasting to all contacts, which the active rule will block to all but the specified JID:

Listing 4: Client Sends Available Presence

```
<presence>
  <status>I'm not really here, you understand!</status>
</presence>
```

3.2.2 Becoming Visible by Roster Group

Listing 5: User Defines and Sets Selective Visibility Privacy Rule (by Roster Group)

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='inv3'>
  <query xmlns='jabber:iq:privacy'>
    <list name='visible-to-Bagginses'>
      <item type='group'
            value='Bagginses'
            action='allow'
            order='1'>
        <presence-out/>
      </item>
      <item action='deny' order='2'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>

<iq from='bilbo@tolkien.lit/shire' type='set' id='active3'>
  <query xmlns='jabber:iq:privacy'>
    <active name='visible-to-Bagginses'/>
  </query>
</iq>
```

The foregoing privacy list blocks outbound presence notifications to every JID except those in a certain roster group. In order to ensure synchronization of presence notifications, the client SHOULD now re-send the user's presence for broadcasting to all contacts, which the active rule will block to all but those JIDs in the specified roster group:

Listing 6: Client Sends Available Presence

```
<presence>
  <status>I'm not really here, you understand!</status>
</presence>
```

3.2.3 Becoming Visible by Subscription Type

Becoming visible or invisible by subscription type is probably much less likely than becoming visible by JID or roster group; however, it is described here for the sake of completeness.

Listing 7: User Defines and Sets Selective Visibility Privacy Rule (by Subscription State)

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='inv4'>
  <query xmlns='jabber:iq:privacy'>
    <list name='visible-to-both'>
      <item type='subscription'
            value='both'
            action='allow'
            order='1'>
        <presence-out/>
      </item>
      <item action='deny' order='2'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>

<iq from='bilbo@tolkien.lit/shire' type='set' id='active4'>
  <query xmlns='jabber:iq:privacy'>
    <active name='visible-to-both'/>
  </query>
</iq>
```

The foregoing privacy list blocks outbound presence notifications to every JID except those that are in the user's roster with a subscription type of "both".
In order to ensure synchronization of presence notifications, the client SHOULD now re-send the user's presence for broadcasting to all contacts, which the active rule will block to all but those JIDs with the specified subscription type:

Listing 8: Client Sends Available Presence

```
<presence>
  <status>I'm␣not␣really␣here,␣you␣understand!</status>
</presence>
```

3.3 Become Globally Visible

Let us now suppose that the user wants to become visible to all those who are subscribed to his presence. This easy to do by defining and setting as active a new privacy list (here again, the privacy list may have been defined previously).

Listing 9: User Defines and Sets Global Visibility Privacy Rule

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='inv5'>
  <query xmlns='jabber:iq:privacy'>
    <list name='visible'>
      <item action='allow' order='1'>
```

```
        <presence-out/>
      </item>
    </list>
  </query>
</iq>

<iq from='bilbo@tolkien.lit/shire' type='set' id='active5'>
  <query xmlns='jabber:iq:privacy'>
    <active name='visible'/>
  </query>
</iq>
```

Because globally allowing outbound presence notifications is most likely the default behavior of any server, a more straightforward way to become globally visible is to decline the use of any active rule (the equivalent, as it were, of taking off a magic invisibility ring):

Listing 10: User Declines the Use of Any Active Rule

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='active6'>
  <query xmlns='jabber:iq:privacy'>
    <active/>
  </query>
</iq>
```

In order to ensure synchronization of presence notifications, the client SHOULD now re-send the user's presence for broadcasting to all contacts, which the active rule will block to all but the specified JID:

Listing 11: Client Sends Available Presence

```
<presence>
  <status>I'm back!</status>
</presence>
```

3.4 Become Selectively Invisible

Let us now suppose that the user no longer wants to be globally visible, but desires to be invisible only to some -- but not all -- contacts. As with visibility, here again the 'jabber:iq:privacy' namespace gives the user three options:

- Become invisible only to specific JIDs
- Become invisible only to specific roster groups
- Become invisible based on subscription state

Examples of these options are shown below.
In general, the process for becoming selectively invisible is as follows:

1. Send unavailable presence to the server.
2. Define and set a privacy rule for selective invisibility.
3. Send available presence to the server.

This process is necessary so that the contacts covered by the rule will no longer see the user as available.

3.4.1 Becoming Invisible by JID

First, the user sends unavailable presence for broadcasting to all contacts:

Listing 12: User Sends Unavailable Presence

```
<presence type='unavailable'/>
```

The server then broadcasts that presence to all of the user's contacts.
Second, the user defines and sets a privacy rule that allows selective invisibility:

Listing 13: User Defines and Sets Selective Invisibility Privacy Rule (by JID)

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='inv6'>
  <query xmlns='jabber:iq:privacy'>
    <list name='invisible-to-Gandalf'>
      <item type='jid'
            value='gandalf@tolkien.lit'
            action='deny'
            order='1'>
        <presence-out/>
      </item>
      <item action='allow' order='2'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>

<iq from='bilbo@tolkien.lit/shire' type='set' id='active7'>
  <query xmlns='jabber:iq:privacy'>
    <active name='invisible-to-Gandalf'/>
  </query>
</iq>
```

The foregoing privacy list allows outbound presence notifications to every JID except one. In order to appear selectively invisible, the client MUST now re-send the user's presence for broadcasting to all contacts, which the active rule will block to the specified JID:

Listing 14: Client Sends Available Presence

```
<presence>
  <status>I'm back!</status>
</presence>
```

3.4.2 Becoming Invisible by Roster Group

First, the user sends unavailable presence for broadcasting to all contacts:

Listing 15: User Sends Unavailable Presence

```
<presence type='unavailable'/>
```

The server then broadcasts that presence to all of the user's contacts.
Second, the user defines and sets a privacy rule that allows selective invisibility:

Listing 16: User Defines and Sets Selective Invisibility Privacy Rule (by Roster Group)

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='inv7'>
  <query xmlns='jabber:iq:privacy'>
    <list name='invisible-to-Wizards'>
      <item type='group'
            value='Wizards'
            action='deny'
            order='1'>
        <presence-out/>
      </item>
      <item action='allow' order='2'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>

<iq from='bilbo@tolkien.lit/shire' type='set' id='active8'>
  <query xmlns='jabber:iq:privacy'>
    <active name='invisible-to-Wizards'/>
  </query>
</iq>
```

The foregoing privacy list allows outbound presence notifications to every JID except those in a certain roster group.

In order to appear selectively invisible, the client MUST now re-send the user's presence for broadcasting to all contacts, which the active rule will block to those in the specified roster group:

Listing 17: Client Sends Available Presence

```
<presence>
  <status>I'm back!</status>
</presence>
```

3.4.3 Becoming Invisible by Subscription Type

Becoming visible or invisible by subscription type is probably much less likely than becoming visible by JID or roster group; however, it is described here for the sake of completeness.
First, the user sends unavailable presence for broadcasting to all contacts:

Listing 18: User Sends Unavailable Presence

```
<presence type='unavailable'/>
```

The server then broadcasts that presence to all of the user's contacts.
Second, the user defines and sets a privacy rule that allows selective invisibility:

Listing 19: User Defines and Sets Selective Invisibility Privacy Rule (by Subscription State)

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='inv8'>
  <query xmlns='jabber:iq:privacy'>
    <list name='invisible-to-from'>
      <item type='subscription'
            value='from'
            action='deny'
            order='1'>
        <presence-out/>
      </item>
      <item action='allow' order='2'>
        <presence-out/>
      </item>
    </list>
  </query>
</iq>

<iq from='bilbo@tolkien.lit/shire' type='set' id='active9'>
  <query xmlns='jabber:iq:privacy'>
    <active name='invisible-to-from'/>
  </query>
</iq>
```

The foregoing privacy list allows outbound presence notifications to every JID except those that are in the user's roster with a subscription type of "to".
In order to appear selectively invisible, the client MUST now re-send the user's presence for broadcasting to all contacts, which the active rule will block to those with the specified subscription type:

Listing 20: Client Sends Available Presence

```
<presence>
  <status>I'm back!</status>
</presence>
```

3.5 Become Globally Invisible

In order to become globally invisible again, the user does the following.
First, the user sends unavailable presence for broadcasting to all contacts:

Listing 21: User Sends Unavailable Presence

```
<presence type='unavailable'/>
```

Second, the user sets as active the global invisibility list previously defined:

Listing 22: User Becomes Globally Invisible

```
<iq from='bilbo@tolkien.lit/shire' type='set' id='active10'>
  <query xmlns='jabber:iq:privacy'>
    <active name='invisible'/>
  </query>
</iq>
```

In order to appear globally invisible, the client MUST now re-send the user's presence for broadcasting to all contacts, which the active rule will block to all contacts:

Listing 23: Client Sends Available Presence

```
<presence>
  <status>I'm not really here, you understand!</status>
</presence>
```

4 Implementation Notes

The foregoing text explains the protocol used to implement invisibility. Naturally, client developers will most likely want to hide these protocol details from the end user. For example, rather than forcing the end user to navigate the details of privacy list management, a client

could simply provide a "Go Invisible" button that sets as active the appropriate privacy list. Note well that the privacy lists used to implement invisibility SHOULD be active lists and *not* the default list.

To help ensure cross-client compatibility, it is RECOMMENDED to use the privacy list names "visible" and "invisible" for simple global visibility and invisibility respectively. It is also RECOMMENDED to use list names of the form "visible-to-GroupName" and "invisible-to-JID" for simple lists that implement visibility or invisibility with regard to roster groups and JIDs. Obviously list names could become rather complex, such as "visible-to-Group1 Group2 Group3". Implementations MUST NOT attempt to derive semantic meaning from privacy list names; these recommendations are provided for ease of use only with regard to basic privacy lists related to visibility/invisibility.

In general it is probably easiest for users to become visible/invisible either globally or based on roster group, since these models are conceptually simple. Although, naturally, a client developer cannot tell users what to do, it probably best to encourage the use of conceptually simple models for privacy lists.

Privacy lists can become complex and must be carefully managed by clients. For example, let us imagine that the user is currently applying another active list unrelated to visibility (e.g., a list that blocks communications with a stalker); if the user then clicks "Go Invisible" and the client is not smart, it could overwrite the stalker blocking. Therefore, if the user has an active list that incorporates rules other than those related to visibility/invisibility, the client SHOULD either assume that visibility/invisibility is an overlay on the list currently in use (generating an appropriate privacy list that takes both into account) or prompt the user regarding their intentions. In the absence of privacy lists unrelated to visibility/invisibility, the client may proceed in a less cautious fashion.

5 Security Considerations

For security concerns related to privacy lists, refer to RFC 3921. Care must be taken regarding privacy lists, especially so that visibility/invisibility rules do not overwrite existing rules the user has set for the sake of security and privacy; for details, see the Implementation Notes section of this document.

It is important to recognize that invisibility can be defeated without more advanced privacy lists than those defined above and an awareness of context on the part of a client. For example, if a user usually logs in as the same resource (e.g., "Home"), a contact can send an IQ request to that resource's full JID using Last Activity (XEP-0012) [3], Service Discovery (XEP-0030) [4], Legacy Entity Time (XEP-0090) [5], or Software Version (XEP-0092) [6] and receive a reply, thus providing information that reveals the user's availability. In addition, Last Activity requests sent by a subscribed contact to the user's bare JID will normally reveal the user's availability

[3] XEP-0012: Last Activity <http://xmpp.org/extensions/xep-0012.html>.
[4] XEP-0030: Service Discovery <http://xmpp.org/extensions/xep-0030.html>.
[5] XEP-0090: Legacy Entity Time <http://xmpp.org/extensions/xep-0090.html>.
[6] XEP-0092: Software Version <http://xmpp.org/extensions/xep-0092.html>.

as well. To help ensure that the user's invisibility cannot be defeated in this way, the user's client SHOULD add IQ blocking to the relevant privacy list. Finally, the user's client SHOULD NOT return "is-composing" events as defined in Message Events (XEP-0022) [7] or Chat State Notifications (XEP-0085) [8].

6 IANA Considerations

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

7 XMPP Registrar Considerations

No namespaces or parameters need to be registered with the XMPP Registrar [10] as a result of this specification.

8 XML Schema

The XML schema for the 'jabber:iq:privacy' namespace is defined in RFC 3921.

[7] XEP-0022: Message Events <http://xmpp.org/extensions/xep-0022.html>.

[8] XEP-0085: Chat State Notifications <http://xmpp.org/extensions/xep-0085.html>.

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