Internet Engineering Task Force
Internet Draft

IMPP WG
Jonathan Rosenberg
Dean Willis
Robert Sparks
Ben Campbell
dynamicsoft

Henning Schulzrinne
Jonathan Lennox
Columbia U.

Bernard Aboba
Christian Huitema
David Gurle
Microsoft

draft-rosenberg-impp-watcherinfo-00.txt
June 15, 2000
Expires: December, 2000

An XML Based Format for Watcher Information

STATUS OF THIS MEMO

Abstract

Watchers are defined as entities that request (i.e., subscribe to) presence information about a user. There is fairly complex state associated with this subscription, and this state is dynamic. As a result, it is possible, and indeed useful, to subscribe to the watcher information for a particular subscriber. In order to enable this, a format is needed to describe the state of watchers. This specification describes an XML document format for such state.

1 Introduction

Presence is (indirectly) defined in RFC2778 [1] as subscription to and notification of changes in the communications state of a user. This communications state consists of the set of communications means, communications address, and status of that user. A presence protocol is a protocol for providing such a service over the Internet or any IP network. Such a protocol is described in [2].

While the focus on presence is the state associated with the communications capabilities and means of a presentity, it is not the only type of state that is important in a complete presence solution. Another piece of state, watcher information, is also needed. Watcher information is the state associated with the subscriptions of watchers (watcher is a generalization of subscriber. A user who fetches content is a watcher, but not a subscriber) for a particular presentity. This state contains dynamic elements, such as whether the subscription is pending or granted, and when the last update was sent to that subscriber. As with any other state that changes, the presence protocols allow users to subscribe to this state and receive notifications about changes in the state.

The ability to fetch watcher information is also useful for learning about who is currently subscribed to you, as a presentity of the system. System administrators might also like to know when a subsriber is added to the system for a particular presentity.

To support subscriptions and notifications of changes in watcher information, a format for describing watcher information is needed. This specification provides such a format. The format contains numerous parameters related to watcher information. As the data can be complex and structure, and will often be useful to render to end users, we have chosen to represent it with XML.

2 Structure of Watcher Information

Watcher information is an XML document that begins with the root element tag "watcherinfo". It consists of any number of "presentity" sub-elements, describing all the presentities being watched.

```
<!ELEMENT watcherinfo (presentity*)>
```

Each presentity element has an attribute giving the URI at which the watching subscribers subscribed to the presentity; it contains as sub-elements a list of the watchers watching this presentity.

```
<!ELEMENT presentity (watcher*)>
<!ATTLIST presentity uri CDATA  #REQUIRED>
```

The watcher element describes a user watching the enclosing presentity.

```
<!ELEMENT watcher (notify-address*)>
<!ATTLIST watcher    uri               CDATA            #REQUIRED
                     status (pending|active|rejected) #REQUIRED
                     first-subscribed CDATA            #IMPLIED
                     most-recently-subscribed CDATA   #IMPLIED>
```

The mandatory attributes of the watcher element are:

uri: The identity of the watcher, expressed as a URI. In the protocol described in [2], this is the URI from the From header of the SUBSCRIBE request.

status: The status of the subscription. Possible values are "pending" if the subscription is awaiting approval, "active" if the subscription is approved and active, and "rejected" if it has been rejected.

There are also two optional, informative attributes of the watcher element. These are:

first-subscribed: The date and time, expressed as an integral number of seconds since January 1, 1970, 00:00 UTC, when the very first SUBSRIBE by this watcher for this presentity was sent. This date and time remains unchanged even if the subscription expires and is later refreshed.

most-recently-subscribed: The date and time, also expressed as seconds since January 1 1970, when the subscription was most recently refreshed. This is only relevant for current subscriptions.

The watcher element may contain a list of addresses at which a particular watcher has asked to be notified.

```
<!ELEMENT notify-address EMPTY>
<!ATTLIST notify-address uri CDATA #REQUIRED>
```

This address is expressed as a URI. There can be more than one of

these, if the watcher has requested more than one notification address. For the protocol described in [2], these are the addresses from the Contact header in the SUBSCRIBE request.

The number of watchers present for each presentity, and the set of presentities listed, depends on the type of data being provided, and to whom. In the case where a watcher has asked for notification of approval of their subscription, the watcher list contains only the watcher information for that one watcher, for that one presentity. In the case where a user wishes to find out the list of users subscribed to his presentity, the list contains multiple watchers for a single presentity. In the case where an administrator wishes to learn the current status in the system, the list contains all watchers for all presentities. Which case to use depends on either modifiers to the subscription described in the body of the SUBSCRIBE, or policy information configured at the presence agent.

3 Document Identifiers

A watcher information document which appears as a top-level XML document is identified with the formal public identifier "-//IETF//DTD RFCxxxx XWATCHER 1.0//EN". If this document is published as an RFC, "xxxx" will be replaced by the RFC number. Watcher lists have the MIME type "application/xwatcher+xml".

> The "+xml" component of the type name conforms with the format of XML MIME types introduced by Murata et al [3]; this allows XML-aware but watcherlist-unaware rendering tools to display watcher lists usefully.

A watcher list embedded as a fragment within another XML document is identified with the XML namespace identifier "http://www.ietf.org/internet-drafts/draft-rosenberg-impp-watcherinfo-00.txt". If this document is published as an RFC, the namespace identifier will be "http://www.rfc-editor.org/rfc/rfcxxxx.txt", where xxxx is the RFC number.

> Note that the URIs specifying XML namespaces are only globally unique names; they do not have to reference any particular actual object. The URI of a canonical source of this specification meets the requirement of being globally unique, and is also useful to document the format.

4 Example

The following is an example of watcher information for a presentity,

who is a professor. There are two watchers, one from a university, and another from an organization.

```
<?xml version="1.0"?>
<!DOCTYPE watcherinfo
   PUBLIC "-//IETF//DTD RFCxxxx XWATCHER 1.0//EN" "watcher.dtd">

<watcherinfo>
  <presentity uri="sip:professor@university.edu;method=SUBSCRIBE">
    <watcher uri="sip:subcriber@university.edu" status="active"
                first-subscribed="958525312" />
    <watcher uri="sip:subcriber@organization.org" status="active"
                first-subscribed="958699312">
      <notify-address
         uri="sip:subscriber@host.organization.org;method=NOTIFY" />
    </watcher>
  </presentity>
</watcherinfo>
```

5 Subscribing to watcher information

The protocol for presence described in [2] can be used to subscribe to watcher information. This is accomplished by definining a namespace that corresponds to watcher information for a particular presentity, and than placing a name within that namespace within the request URI of the SUBSCRIBE request.

Our proposal for the namespace construction is:

```
watcher-URL =  "sip:" watcher "%40" presentity "@" hostport
watcher = 1*(token | escaped)
presentity = 1*(token | escaped)
```

The watcher is a URL-encoded version of the identity of the watcher, and the presentity is a URL encoded version of the identify of the presentity, both only containing the username and hostname. For example, if the watcher is joe@ex.com, and the presentity is user@un.edu, the URL used to subscribe to joe's watcher information about user is:

```
sip:joe%40ex.com%40user%40un.edu@un.edu
```

To subscribe to this information, the SUBSCRIBE would look like:

```
SUBSCRIBE sip:joe%40ex.com%40user%40un.edu@un.edu SIP/2.0
Via: SIP/2.0/UDP mypc.ex.com
From: sip:joe@ex.com
To: sip:joe%40ex.com%40user%40un.edu@un.edu
Call-ID: 9hsdasd@1.2.3.4
CSeq: 1 SUBSCRIBE
Accept: application/xwatcher+xml
```

Note that the subscription indicates that this format defined here is required in the response and in notifications.

6 XML DTD

```
<?xml version="1.0" encoding="US-ASCII" ?>

<!--
    Draft DTD for Watcher info.
-->

<!-- Watcher element: document root -->

<!ELEMENT watcherinfo (presentity*)>

<!-- Presentity element: describes one entity being watched -->

<!ELEMENT presentity (watcher*)>
<!ATTLIST presentity uri CDATA  #REQUIRED>

<!-- Watcher element: describes one entity watching -->

<!ELEMENT watcher    (notify-address*)>
<!ATTLIST watcher    uri              CDATA           #REQUIRED
                     status (pending|active|rejected) #REQUIRED
                     first-subscribed CDATA           #IMPLIED
                     most-recently-subscribed CDATA   #IMPLIED>

<!-- Notify-address element: describes an address at which a watcher is to
      be notified. -->

<!ELEMENT notify-address EMPTY>
<!ATTLIST notify-address uri CDATA #REQUIRED>
```

7 Security Considerations

Watcher information is sensitive information. The protocol used to distribute it SHOULD ensure privacy, message integrity and authentication. Furthermore, the protcol should provide access controls which restrict who can see who elses watcher information.

8 Authors Addresses

Jonathan Rosenberg
dynamicsoft
200 Executive Drive
Suite 120
West Orange, NJ 07052
email: jdrosen@dynamicsoft.com

Dean Willis
dynamicsoft
200 Executive Drive
Suite 120
West Orange, NJ 07052
email: dwillis@dynamicsoft.com

Robert Sparks
dynamicsoft
200 Executive Drive
Suite 120
West Orange, NJ 07052
email: rsparks@dynamicsoft.com

Ben Campbell
dynamicsoft
200 Executive Drive
Suite 120
West Orange, NJ 07052
email: bcampbell@dynamicsoft.com

Henning Schulzrinne
Columbia University
M/S 0401
1214 Amsterdam Ave.
New York, NY 10027-7003
email: schulzrinne@cs.columbia.edu

Jonathan Lennox
Columbia University
M/S 0401

1214 Amsterdam Ave.
New York, NY 10027-7003
email: lennox@cs.columbia.edu

Christian Huitema
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
email: huitema@microsoft.com

Bernard Aboba
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
email: bernarda@microsoft.com

David Gurle
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052-6399
email: dgurle@microsoft.com

9 Bibliography

[1] M. Day, J. Rosenberg, and H. Sugano, "A model for presence and instant messaging," Request for Comments 2778, Internet Engineering Task Force, Feb. 2000.

[2] J. Rosenberg, R. Sparks, D. Willis, B. Campbell, H. Schulzrinne, J. Lennox, C. Huitema, B. Aboba, and D. Gurle, "SIP extensions for presence," Internet Draft, Internet Engineering Task Force, June 2000. Work in progress.

[3] M. Murata and S. S. Laurent, "XML media types," Internet Draft, Internet Engineering Task Force, Jan. 2000. Work in progress.