Abstract

This document describes a usage of the Extensible Markup Language (XML) Configuration Access Protocol (XCAP) for manipulating lists of presentities (also known as buddy lists or rosters). It does so by specifying an XML Schema that contains a list of presentities that a user is interested in watching.
1. Introduction

The Session Initiation Protocol (SIP) for Instant Messaging and Presence (SIMPLE) specifications allow a user, called a watcher, to subscribe to another user, called a presentity [7], in order to learn their presence information [8]. In many cases, a watcher will be interested in not just a single presentity, but a list of presentities. Such a list of presentities is called a presence list.

When a user wants to subscribe to their presence list, the procedures in [8] require the watcher to create and manage a subscription for each presentity in their list. For large lists, the bandwidth required to do this can be a problem, particularly for wireless networks. An extension to the SIP events framework [9] has been defined. The extension allows a watcher to subscribe to a list of resources, using a single subscription [10]. This mechanism assumes that a server, called the Resource List Server (RLS) has a copy of the presence list that the user wishes to subscribe to. By using protocols such as the XML Configuration Access Protocol (XCAP) [12], a client can place this list on the server, and manipulate it as needed.

XCAP requires application usages to standardize several pieces of information, including an application unique ID (AUID), an XML schema, and various other pieces of information. This specification fulfills those requirements.

The XML schema defined here has several other usages outside of XCAP:

1. A PC client application will need to know the users in the presence list, so that it can generate a subscription to each one. This information represents user provisioned data for the application. Typically, this information is stored on local disk in a proprietary file format. By defining a standard format, the same list can be used by a multiplicity of different client applications, providing portability across them.

2. It is common for users to share presence lists. As an example, user A may have three people in their list that they wish to tell user B about. User A would like to send an email to user B with an attachment describing these three people. Should user B open the attachment, the three people can be added to their own presence list. Doing this requires a standardized format for exchanging lists over email, instant messaging, and other communications protocols.
2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [1] and indicate requirement levels for compliant implementations.
3. Application Unique ID

XCAP requires application usages to define a unique application usage ID (AUID) in either the IETF tree or a vendor tree. This specification defines the "presence-lists" AUID within the IETF tree, via the IANA registration in Section 12.
4. Structure of a Presence List

A presence list is an XML [2] document that MUST be well-formed and SHOULD be valid. Presence list documents MUST be based on XML 1.0 and MUST be encoded using UTF-8. This specification makes use of XML namespaces for identifying presence list documents and document fragments. The namespace URI for elements defined by this specification is a URN [3], using the namespace identifier 'ietf' defined by [5] and extended by [6]. This URN is:

    urn:ietf:params:xml:ns:presence-lists

A presence list document begins with the root element tag 'presence-lists'. It consists of any number of 'list' sub-elements, each of which is a presence list. Other elements from different namespaces MAY be present for the purposes of extensibility; elements or attributes from unknown namespaces MUST be ignored. There are three attributes associated with this element. The first two, "name", and "subscribable" MUST be present:

    name: This attribute is a descriptive name for the list. It MUST be unique amongst all other list elements within the same parent element.

    subscribable: This boolean attribute indicates whether or not the list is subscribable or not.

The other attribute, "uri" MAY be present:

    uri: This attribute provides a URI that can be used to subscribe to the list, using the SIP event notification extension for lists [10]. As a result, the URI MUST be either a SIP URI or a pres URI [11]. [[OPEN ISSUE: Do we want this to be a comma separated list, so that a presence list can have any number of valid aliases?]]

[[OPEN ISSUE: We also need to define policy about who is allowed to subscribe to the list. We can either integrate that into the list definition described here, or handle that as a separate policy specification. The SEACAP proposal kept them together. I believe now that they should be kept separate.]]

Each list element is composed of a sequence of entry elements or list elements. The ability of a list element to contain other list elements means that a presence list can be hierarchically structured. An entry element describes a single presentity that is part of the list. A list element can also contain elements from other namespaces, for the purposes of extensibility.
The entry element describes a single presentity. The entry element has two attributes:

- **name**: This mandatory attribute is a unique identifier amongst all other entry elements of the same parent.
- **uri**: This optional attribute is a URI that is used to subscribe to the presentity. It MUST be either a SIP or pres URI.

The entry element contains a sequence of other elements. Only one such element is defined at this time, which is the **display-name**. This element provides a UTF-8 encoded string, meant for consumption by the user, that describes the presentity. Unlike the "name" attribute of the entry element, the display-name has no uniqueness requirements. [[OPEN ISSUE: Do we need this in addition to the name attribute?]]. Other elements from other namespaces MAY be included. This is meant to support the inclusion of other information about the entry, such as a phone number or postal address.
5. Computed Data

An XCAP server supporting this application usage need only worry about a single piece of computed data – the "uri" attribute of the list element.

If the "uri" attribute is absent in a document written to an XCAP server, but the "subscribable" flag is true, the XCAP server MUST allocate a URI for this list. This allocated URI MUST be globally unique, and MUST route to an RLS which will handle list subscriptions for the list defined by the document. The server MUST set the uri attribute of the document with this URI.

A server MUST NOT delete the "uri" attribute, however, should a client change the subscribable flag to false after the server has allocated a URI.
6. Additional Constraints

There are no constraints on the document beyond those described in the schema.
7. Naming Conventions

There are no naming conventions that need to be defined for this application usage. A subscription to a presence list will be to a specific URI. That URI will be one of the "uri" attributes defined in a list within one of the documents managed by an XCAP server.
8. Authorization Policies

This application usage does not modify the default XCAP authorization policy, which is that only a user can read, write or modify their own documents. A server can allow privileged users to modify documents that they don’t own, but the establishment and indication of such policies is outside the scope of this document.
9. XML Schema

The following is the XML schema definition of the presence list:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">
  <xs:element name="presence-lists">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="list" type="listType" minOccurs="0"
            maxOccurs="unbounded"/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:complexType name="listType">
    <xs:sequence maxOccurs="unbounded">
      <xs:choice>
        <xs:element name="list" type="listType" minOccurs="0"
            maxOccurs="unbounded"/>
        <xs:element name="entry" type="entryType" minOccurs="0"
            maxOccurs="unbounded"/>
        <xs:any namespace="##other" processContents="lax"
            minOccurs="0" maxOccurs="unbounded"/>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="uri" type="xs:anyURI" use="optional"/>
    <xs:attribute name="subscribable" type="xs:boolean" use="required" />
  </xs:complexType>

  <xs:complexType name="entryType">
    <xs:sequence>
      <xs:element name="display-name" type="display-nameType"/>
      <xs:any namespace="##other" processContents="lax"
        minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" type="xs:string" use="required"/>
    <xs:attribute name="uri" type="xs:anyURI" use="optional"/>
  </xs:complexType>

  <xs:simpleType name="display-nameType">
    <xs:restriction base="xs:string"/>
  </xs:simpleType>
</xs:schema>
```
10. Example Document

The following is an example of a document compliant to the schema:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<presence-lists xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <list name="friends" uri="sip:friends@example.com" subscribable="true">
    <entry name="Bill" uri="sip:bill@example.com">
      <display-name>Bill Doe</display-name>
    </entry>
  </list>
  <list name="close-friends" uri="sip:close-friends@example.com" subscribable="true">
    <entry name="Joe" uri="sip:joe@example.com">
      <display-name>Joe Smith</display-name>
    </entry>
    <entry name="Nancy" uri="sip:nancy@example.com">
      <display-name>Nancy Gross</display-name>
    </entry>
  </list>
</presence-lists>
```
11. Security Considerations

The configuration information defined by this application usage is particularly sensitive. It represents the principle set of people with whom a user would like to communicate. As a result, clients SHOULD use TLS when contacting servers in order to fetch this information. Note that this does not represent a change in requirement strength from XCAP.
12. IANA Considerations

There are several IANA considerations associated with this specification.

12.1 XCAP Application Usage ID

This section registers a new XCAP Application Usage ID (AUId) according to the IANA procedures defined in [12].

Name of the AUId: presence-lists

Description: A presence list application is a usage of the SIP events framework (RFC 3265) [9] along with its list extension [10], for subscribing to a list of presentities (RFC2778) [7] stored at a server.

12.2 application/presence-lists+xml MIME Type

MIME media type name: application

MIME subtype name: presence-lists+xml

Mandatory parameters: none

Optional parameters: Same as charset parameter application/xml as specified in RFC 3023 [4].

Encoding considerations: Same as encoding considerations of application/xml as specified in RFC 3023 [4].

Security considerations: See Section 10 of RFC 3023 [4] and Section 11 of this specification.

Interoperability considerations: none.

Published specification: This document.

Applications which use this media type: This document type has been used to support subscriptions to lists of users [10] for SIP-based presence [8].

Additional Information:

  Magic Number: None
12.3 URN Sub-Namespace Registration for
urn:ietf:params:xml:ns:presence-lists

This section registers a new XML namespace, as per the guidelines in
[6]

URI: The URI for this namespace is

Registrant Contact: IETF, SIMPLE working group,
(simple@mailman.dynamicsoft.com), Jonathan Rosenberg
(jdrosen@jdrosen.net).

XML:

BEGIN
<?xml version="1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.0//EN"
   "http://www.w3.org/TR/xhtml-basic/xhtml-basic10.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta http-equiv="content-type"
       content="text/html;charset=iso-8859-1"/>
  <title>Presence Lists Namespace</title>
</head>
<body>
  <h1>Namespace for Presence Lists</h1>
  <h2>application/presence-lists+xml</h2>
  <p>See <a href="[[URL of published RFC]]">RFCXXX</a>.</p>
</body>
</html>
END
Normative References


Informative References


Author’s Address

Jonathan Rosenberg
dynamicsoft
600 Lanidex Plaza
Parsippany, NJ 07052
US

Phone: +1 973 952-5000
EMail: jdrosen@dynamicsoft.com
URI: http://www.jdrosen.net
Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF’s procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

Full Copyright Statement

Copyright (C) The Internet Society (2003). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assignees.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION
HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.