SIMPLE
Internet-Draft
Expires: July 26, 2005

J. Rosenberg
Cisco Systems

January 25, 2005

An Extensible Markup Language (XML) Document Format for Indicating Changes in XML Configuration Access Protocol (XCAP) Resources
draft-ietf-simple-xcap-package-03

Abstract

This specification defines a document format that can be used to describe the differences between versions of resources managed by the Extensible Markup Language (XML) Configuration Access Protocol (XCAP). Documents of this format can be delivered to clients using a number of means, including the Session Initiation Protocol (SIP) event package for configuration data. By subscribing to this event package, clients can learn about document changes made by other

clients.

Table of Contents

1. Introduction

The Extensible Markup Language (XML) Configuration Access Protocol (XCAP) [7] is a protocol that allows clients to manipulate XML documents stored on a server. These XML documents serve as configuration information for application protocols. As an example, resource list [11] subscriptions (also known as presence lists) allow a client to have a single SIP subscription to a list of users, where the list is maintained on a server. The server will obtain presence for those users and report it back to the client. This application requires the server, called a Resource List Server (RLS), to have access to the list of presentities. This list needs to be manipulated by clients so they can add and remove their friends as they desire.

Complexities arise when multiple clients attempt to simultaneously manipulate a document, such as a presence list. Frequently, a client will keep a copy of the current list in memory, so it can render it to users. However, if another client modifies the document, the cached version becomes stale. This information must be made known to all clients which have cached copies of the document, so that they can fetch the most recent one.

To deal with this problem, clients can use the Session Initiation Protocol (SIP) [9]event package [10] for subscribing to changes in configuration and profile information [8], including application data that resides on an XCAP server. With that package, a user gets notified that a particular document has changed. This notification can include the full content of the new document, or it can be a content indirection [14]. However, in both cases, the transfer of the entire document is ultimately required. This may require a lot of bandwidth, particularly for wireless devices with large documents (such as a resource list [11] with hundreds of users listed).

To resolve this problem, this document defines a data format which can convey changes in XML documents managed by an XCAP server. This data format is an XML document format, called an XCAP diff document. This specification also explains how this format is used in conjunction with the configuration profile framework.

2. Structure of an XCAP Diff Document

An XCAP diff document is an XML [2] document that MUST be well-formed and SHOULD be valid. XML-change documents MUST be based on XML 1.0 and MUST be encoded using UTF-8. This specification makes use of XML namespaces for identifying xml-change 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:xcap-diff

An XCAP diff document begins with the root element tag <xcap-diff>. This element has a single mandatory attribute, "xcap-root". The value of this attribute is the XCAP root URI in which the changes have taken place. A single XCAP diff document can only represent changes in documents within the same XCAP root. The content of the <xcap-diff> element is a sequence of <document> elements. Each <document> element specifies changes in a specific document within the XCAP root. It has three mandatory attributes - "new-etag", "previous-etag" and "doc-selector", and a single optional attribute, "hash". The "doc-selector" identifies the specific document within the XCAP root for which changes are included. Its content MUST be a relative path reference, with the base URL being equal to the XCAP root URL. The "previous-etag" and "new-etag" provide indentifiers for the document instance before the change, and then after the change. These need not have been sequentially assigned etags at the server. An XCAP diff document can describe changes that have occurred over a series of XCAP operations.

The optional "hash" attribute provides an HMAC of the new document, represented in canonical form. See Section 5 for details on how this value is computed. This attribute is optional, and a server can elect not to include it.

Each <document> element is followed by a series of operations, which if followed by the client, will convert the document whose etag is "previous-etag" into the one whose etag is "new-etag". Each operation is specified by an XML element. Six operations are defined:

<add-element>: Instructs the recipient of the document to add an element. The "parent" attribute contains a node-selector which selects the parent of the new element. The "position" attribute indicates that the new element is to be inserted as a child such that it has that position amongst it siblings. The content of <add-element> MUST be a CDATA section enclosing a single XML element, which is to be added.

<add-attribute>: Instructs the recipient of the document to add an attribute. The "element" attribute contains a node-selector which selects the element into which the attribute is to be inserted. The "att-name" attribute contains the name of the new attribute. The content of <add-attribute> is the value of the new attribute.

<remove-element>: Instructs the recipient of the document to delete an element. The "element" attribute contains a node-selector which selects the element to be removed.

<remove-attribute>: Instructs the recipient of the document to remove an attribute. The "element" attribute contains a node-selector which selects the element in which the attribute exists. The "att-name" attribute indicates the name of the attribute which is to be removed.

<replace-element>: Instructs the recipient of the document to replace an element. The "element" attribute contains a node-selector which selects the element to replace. The content of the <replace-element> element MUST be a CDATA section containing single XML element which is to replace the one identified by the node-selector.

<replace-attribute>: Instructs the recipient of the document to replace an attribute. The "element" attribute contains a node-selector which selects the element to replace. The "att-name" attribute contains the name of the attribute to replace. The content of the <replace-attribute> element is the value of the new attribute.

When the node selector appears as an attribute value, any quotation marks MUST be replaced with ".

It is possible for the list of instructions for a <document> to be empty. In that case, the entity tag in the "new-etag" may equal the entity tag in the "previous-etag". These entity tags may differ in the event that the document has changed entity tags, but its content has not been altered.

3. XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="urn:ietf:params:xml:ns:xcap-diff"
 xmlns:tns="urn:ietf:params:xml:ns:xcap-diff"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified" attributeFormDefault="unqualified">
 <xs:element name="xcap-diff">
  <xs:complexType>
   <xs:sequence>
    <xs:element name="document" maxOccurs="unbounded">
     <xs:complexType>
      <xs:sequence minOccurs="0" maxOccurs="unbounded">
       <xs:choice>
```

```
            <xs:element name="add-element">
             <xs:complexType>
              <xs:simpleContent>
               <xs:extension base="xs:string">
                <xs:attribute name="parent" type="xs:string" use="required"/>
                <xs:attribute name="position" type="xs:int" use="required"/>
               </xs:extension>
              </xs:simpleContent>
             </xs:complexType>
            </xs:element>
            <xs:element name="add-attribute">
             <xs:complexType>
              <xs:simpleContent>
               <xs:extension base="xs:string">
                <xs:attribute name="element" type="xs:string" use="required"/>
                <xs:attribute name="att-name" type="xs:string" use="required"/>
               </xs:extension>
              </xs:simpleContent>
             </xs:complexType>
            </xs:element>
            <xs:element name="remove-element">
             <xs:complexType>
              <xs:attribute name="element" type="xs:string" use="required"/>
             </xs:complexType>
            </xs:element>
            <xs:element name="remove-attribute">
             <xs:complexType>
              <xs:attribute name="element" type="xs:string" use="required"/>
              <xs:attribute name="att-name" type="xs:string" use="required"/>
             </xs:complexType>
            </xs:element>
            <xs:element name="replace-element">
             <xs:complexType>
              <xs:simpleContent>
               <xs:extension base="xs:string">
                <xs:attribute name="element" type="xs:string" use="required"/>
               </xs:extension>
              </xs:simpleContent>
             </xs:complexType>
            </xs:element>
            <xs:element name="replace-attribute">
             <xs:complexType>
              <xs:simpleContent>
               <xs:extension base="xs:string">
                <xs:attribute name="element" type="xs:string" use="required"/>
                <xs:attribute name="att-name" type="xs:string" use="required"/>
               </xs:extension>
              </xs:simpleContent>
```

```
           </xs:complexType>
          </xs:element>
         </xs:choice>
        </xs:sequence>
        <xs:attribute name="doc-selector" type="xs:anyURI" use="required"/>
        <xs:attribute name="new-etag" type="xs:string" use="required"/>
        <xs:attribute name="previous-etag" type="xs:string" use="required"/>
        <xs:attribute name="hash" type="xs:string" use="optional"/>
       </xs:complexType>
      </xs:element>
     </xs:sequence>
     <xs:attribute name="xcap-root" type="xs:anyURI" use="required"/>
   </xs:complexType>
  </xs:element>
 </xs:schema>
```

4. Example Document

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

```
<?xml version="1.0" encoding="UTF-8"?>
<xcap-diff xmlns="urn:ietf:params:xml:ns:xcap-diff"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <document doc-selector="resource-lists/users/joe/friends"
  new-etag="7hahsd" previous-etag="7hahsd"/>
 <document doc-selector="resource-lists/users/joe/coworkers"
  new-etag="ffds66a" previous-etag="xkkkaisu">
   <add-element parent="resource-lists/list[@name=&quot;l1&quot;]"
    position="1"><![CDATA[<entry uri="sip:new-worker@example.com"/>]
]></add-element>
 </document>
</xcap-diff>
```

5. Usage with the Config Framework

The framework for user agent profile delivery [8] defines an event package which can be used to subscribe to user, device, application or local-network data. This data can be present in an XCAP server. Normally, content indirection [14] will be used as the NOTIFY body format, to indicate the specific document that has changed, and should be re-fetched. However, if the client includes an Accept header field including the MIME type "application/xcap-diff+xml", the server has the option of returning documents in this format instead.

When the client performs an initial subscription, the rules in [8] are used to select the set of documents which the subscription applies to. Upon initial subscription, the server does not know which instance (where each instance is identified by an etag) the client currently posessses, if any. Indeed, upon startup, the client will not have any documents. The initial NOTIFY in this case MUST include a <document> element for each document associated with the subscription. The content of each of those <document> elements MUST be empty. The "previous-etag" and "new-etag" attributes MUST be identical, and contain the entity tag for the current version of that resource. An XML diff document structured this way is called a "reference" XML diff document. It establishes the baseline etags and document URIs for the documents covered by the subscription.

Upon receipt of this document, the client can determine whether its local instance documents, if any, match the etags in the XCAP diff document. If they do not match, the client SHOULD perform a conditional GET for each document. The document URI is constructed by appending the XCAP root in the "xcap-root" attribute of the <xcap-diff> element to the escape coded "doc-selector" from each <document> element. The request is made conditional by including an If-Match header field, with the value of the etag from each <document> element. So long as the documents haven't changed between the NOTIFY and the GET, the client will obtain the reference versions that the server will use for subsequent notifications.

If the conditional GET should fail, the client SHOULD generate a SUBSCRIBE refresh request to trigger a new NOTIFY. The server will always generate a "reference" XML diff document on receipt of a SUBSCRIBE refresh. This establish a new set of baseline etags, and the client can then attempt to do another fetch. It is anticipated that future extensions to the profile delivery framework will allow a client to include, in its SUBSCRIBE request, an indicator of the current version of the documents it holds. That would obviate the need for a potentially never-ending stream of SUBSCRIBE/GET sequences should the documents be rapidly changing, for some reason.

Once the client has obtained the versions of the documents identified in the reference XML diff, it can process NOTIFY requests on that subscription. To process the NOTIFY requests, it makes sure that its current version matches the version in the "previous-etag" attribute of the <document> element. It then follows the list of instructions, in order, for that <document>. Specifically:

<add-element>: The "parent" attribute contains a node-selector. The client applies the node selector to the document according to the procedures defined in XCAP [7]. If the result is a single element, the client takes the content of the <add-element> element

and adds it as the position-th child. If the node selector doesnt select a single element, or the selected element has fewer than position-1 children already, the result is an error. The client MUST discard the XCAP-diff document, and MUST flush its current version of the document from memory. It can then obtain a new XML diff reference by sending a SUBSCRIBE refresh request on the dialog.

<add-attribute>: The "element" attribute contains a node-selector. The client applies the node selector to the document according to the procedures defined in XCAP [7]. If the result is a single element, the client takes adds a new attribute to the element, with the name equal to the content of the "att-name" attribute, and a value equal to the content of the <add-attribute> element. If the node selector doesnt select a single element, or the selected element already has an attribute with that name, the result is an error. The client MUST discard the XCAP-diff document, and MUST flush its current version of the document from memory. It can then obtain a new XML diff reference by sending a SUBSCRIBE refresh request on the dialog.

<remove-element>: The "element" attribute contains a node-selector. The client applies the node selector to the document according to the procedures defined in XCAP [7]. If the result is a single element, the client removes that element from the document. If the node selector doesnt select a single element the result is an error. The client MUST discard the XCAP-diff document, and MUST flush its current version of the document from memory. It can then obtain a new XML diff reference by sending a SUBSCRIBE refresh request on the dialog.

<remove-attribute>: The "element" attribute contains a node-selector. The client applies the node selector to the document according to the procedures defined in XCAP [7]. If the result is a single element, the client removes the attribute whose name is "att-name". If the node selector doesnt select a single element, or the selected element doesn't have an attribute with that name, the result is an error. The client MUST discard the XCAP-diff document, and MUST flush its current version of the document from memory. It can then obtain a new XML diff reference by sending a SUBSCRIBE refresh request on the dialog.

<replace-element>: The "element" attribute contains a node-selector. The client applies the node selector to the document according to the procedures defined in XCAP [7]. If the result is a single element, the client removes that element, and replaces it with the content of the <add-element> element. If the node selector doesnt select a single element, the result is an error. The client MUST

discard the XCAP-diff document, and MUST flush its current version of the document from memory. It can then obtain a new XML diff reference by sending a SUBSCRIBE refresh request on the dialog.

<replace-attribute>: The "element" attribute contains a node-selector. The client applies the node selector to the document according to the procedures defined in XCAP [7]. If the result is a single element, the client removes the content of the attribute whose name is "att-name", and replaces it with the content of the <replace-attribute> element. If the node selector doesnt select a single element, or the selected element doesn't have an attribute with that name, the result is an error. The client MUST discard the XCAP-diff document, and MUST flush its current version of the document from memory. It can then obtain a new XML diff reference by sending a SUBSCRIBE refresh request on the dialog.

Once the client has finished applying the instructions to the document, it should end up with the same document the server has. To verify this, the client applies the mandatory XML canonicalization defined in the Canonical XML 1.0 [1] specification, and computes an HMAC [12] using SHA1 over this canonical document, with a key whose value is 0x2238a. The resulting string is compared with the "hash" attribute of the <document> element. If they match, the client can be sure that it has the most up to date version. If they don't match, the client MUST flush its current version of the document from memory. It can then obtain a new XML diff reference by sending a SUBSCRIBE refresh request on the dialog.

Of course, this mechanism for computing the most current document from the hash is optional. A client can elect to ignore the information on what changed and simply fetch the most recent document every time it gets a change indication where the new version is not the same as the one cached by the client. Furthermore, the server may elect to not send the hash, in which case this check cannot be made.

6. Security Considerations

XCAP diff documents contain the same information in the documents whose differences they describe. As such, the security considerations associated with those documents apply to XCAP diff documents.

7. IANA Considerations

There are several IANA considerations associated with this specification.

7.1 application/xcap-diff+xml MIME Type

MIME media type name: application

MIME subtype name: xcap-diff+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 6 of RFCXXXX [[NOTE TO RFC-EDITOR/IANA: Please replace XXXX with the RFC number of this specification.]].

Interoperability considerations: none.

Published specification: This document.

Applications which use this media type: This document type has been used to support manipulation of resource lists [13] using XCAP.

Additional Information:

Magic Number: None

File Extension: .xdf or .xml

Macintosh file type code: "TEXT"

Personal and email address for further information: Jonathan Rosenberg, jdrosen@jdrosen.net

Intended usage: COMMON

Author/Change controller: The IETF.

7.2 URN Sub-Namespace Registration for urn:ietf:params:xml:ns:xcap-diff

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

URI: The URI for this namespace is
urn:ietf:params:xml:ns:xcap-diff.

Registrant Contact: IETF, SIMPLE working group, (simple@ietf.org),
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>XCAP Diff Namespace</title>
</head>
<body>
  <h1>Namespace for XCAP Diff</h1>
  <h2>urn:ietf:params:xml:ns:xcap-diff</h2>
  <p>See <a href="[URL of published RFC]">RFCXXXX[[NOTE
TO IANA/RFC-EDITOR: Please replace XXXX with the RFC number of this
specification.]]</a>.</p>
</body>
</html>
END
```

7.3 Schema Registration

This section registers a new XML schema per the procedures in [6].

URI: urn:ietf:params:xml:schema:xcap-diff

Registrant Contact: IETF, SIMPLE working group, (simple@ietf.org),
Jonathan Rosenberg (jdrosen@jdrosen.net).

The XML for this schema can be found as the sole content of
Section 3.

8. References

8.1 Normative References

[1] Boyer, J., "Canonical XML Version 1.0", W3C REC

REC-xml-c14n-20010315, March 2001.

[2] Bray, T., Paoli, J., Sperberg-McQueen, C. and E. Maler, "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C FirstEdition REC-xml-20001006, October 2000.

[3] Moats, R., "URN Syntax", RFC 2141, May 1997.

[4] Murata, M., St. Laurent, S. and D. Kohn, "XML Media Types", RFC 3023, January 2001.

[5] Moats, R., "A URN Namespace for IETF Documents", RFC 2648, August 1999.

[6] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[7] Rosenberg, J., "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)", draft-ietf-simple-xcap-05 (work in progress), November 2004.

[8] Petrie, D., "A Framework for Session Initiation Protocol User Agent Profile Delivery", draft-ietf-sipping-config-framework-05 (work in progress), November 2004.

8.2 Informative References

[9] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M. and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, June 2002.

[10] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.

[11] Roach, A., Rosenberg, J. and B. Campbell, "A Session Initiation Protocol (SIP) Event Notification Extension for Resource Lists", draft-ietf-simple-event-list-07 (work in progress), January 2005.

[12] Krawczyk, H., Bellare, M. and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, February 1997.

[13] Rosenberg, J., "Extensible Markup Language (XML) Formats for Representing Resource Lists", draft-ietf-simple-xcap-list-usage-04 (work in progress), October 2004.

[14] Burger, E., "A Mechanism for Content Indirection in Session

Initiation Protocol (SIP) Messages",
draft-ietf-sip-content-indirect-mech-05 (work in progress),
October 2004.

Author's Address

Jonathan Rosenberg
Cisco Systems
600 Lanidex Plaza
Parsippany, NJ 07054
US

Phone: +1 973 952-5000
EMail: jdrosen@cisco.com
URI: http://www.jdrosen.net

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

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