Internet Engineering Task Force (IETF)
Request for Comments: 5874
Category: Standards Track
ISSN: 2070-1721

J. Rosenberg
jdrosen.net
J. Urpalainen
Nokia

May 2010

An Extensible Markup Language (XML) Document Format for Indicating a Change in XML Configuration Access Protocol (XCAP) Resources

Abstract

This specification defines a document format that can be used to indicate that a change has occurred in a document managed by the Extensible Markup Language (XML) Configuration Access Protocol (XCAP). This format reports which document has changed and its former and new entity tags. It can report the differences between versions of the document, using an XML patch format. It can report existing element and attribute content when versions of an XCAP server document change. XCAP diff documents can be delivered to diff clients using a number of means, including a Session Initiation Protocol (SIP) event package.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 5741.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at http://www.rfc-editor.org/info/rfc5874.

Table of Contents

1. Introduction

The Extensible Markup Language (XML) Configuration Access Protocol (XCAP) [RFC4825] is a protocol that allows XCAP clients to manipulate XML documents stored on a server. These XML documents serve as configuration information for application protocols. As an example, resource list [RFC4662] subscriptions (also known as presence lists) allow a SIP 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 SIP client. This application requires the server, called a Resource List Server (RLS), to have access to the list of presentities [RFC2778]. This list needs to be manipulated by XCAP clients so they can add and remove their friends as they desire.

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

To deal with this problem, clients can use a Session Initiation Protocol (SIP) [RFC3261] event package [RFC3265] to subscribe to change events [RFC5875] in XCAP documents. This notification needs to indicate the specific resource that changed and how it changed. One solution for the format of such a change notification would be a content indirection object [RFC4483]. Though content indirection can tell a client that a document has changed, it provides it with a MIME Content-ID indicating the new version of the document. The MIME Content-ID is not the same as the entity tag, which is used by XCAP for document versioning. As such, a client cannot easily ascertain whether an indication of a change in a document is due to a change it just made or due to a change another XCAP client made at around the same time. Furthermore, content indirections don't indicate how a document changed; they are only able to indicate that it did change.

To resolve these problems, this document defines a data format that can convey the fact that an XML document managed by XCAP has changed. This data format is an XML document format, called an XCAP diff document. This format reports which document has changed and its former and new entity tags. It can report the differences between versions of the document, using an XML patch format [RFC5261], which indicate how to transform the locally cached XCAP document from the version prior to the change to the version after it. Its intent is to reduce the required overall bandwidth and the number of separate

transmissions. It can also report existing element and attribute content when versions of an XML document change at an XCAP server.

XML documents that are equivalent for the purposes of many applications may differ in their physical representation. Similar to XCAP, the canonical form with comments [W3C.REC-xml-c14n-20010315] of an XML document determines the logical equivalence when this format is used to patch locally cached XCAP documents.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119] and indicate requirement levels for compliant implementations.

This specification also defines the following additional terms:

Document: When the term document is used without the "(XCAP) diff" in front of it, it refers to the XCAP document resource about which the XCAP diff document is reporting a change.

Diff document: The XML document defined by this specification that reports on a set of changes in an XCAP document resource. It is delivered from a server to a diff client by a transport that is not defined by this specification.

XCAP server: A protocol entity that manages XCAP documents and their entity tags. It usually contains an integrated diff notifier.

Diff notifier: This is the entity of a server that generates XCAP diff documents based on its knowledge of a set of XCAP documents and their changes, and it transmits the generated diff documents to a diff client within a session.

Diff client: A client that consumes XCAP diff documents in order to construct a locally cached document that is equivalent to a specific version of a document resource stored at an XCAP server. It is typically a SIP User Agent (UA) and an XCAP client.

XCAP Client: A client that updates and retrieves documents stored at an XCAP server. It can also patch element and attribute content of XCAP documents located at an XCAP server.

Locally cached resource: A resource that has typically been downloaded by HTTP from an XCAP server to a diff client. It may have been patched locally by a diff client based on the XCAP diff document information. It is equivalent to a single version in its

change history at an XCAP server. Version history of XCAP documents is indicated by HTTP entity tags (ETags).

ETag: A strong HTTP entity tag whose value is set by an XCAP server. Documents at an XCAP server are updated by XCAP clients. The XCAP server assigns a new ETag value to each document version according to the HTTP specification.

3. Structure of an XCAP Diff Document

An XCAP diff document is an XML [W3C.REC-xml-20060816] document that MUST be well-formed and SHOULD be valid. XCAP diff documents MUST be based on XML 1.0 and MUST be encoded using UTF-8. This specification makes use of XML namespaces for identifying XCAP diff documents and document fragments. The namespace URI for elements defined by this specification is a URN [RFC2141], using the namespace identifier 'ietf' defined by [RFC2648] and extended by [RFC3688]. 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 for the documents 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>, <element>, and <attribute> elements followed by any number of elements from other namespaces for the purposes of extensibility. Wherever the XML schema (see Section 4) allows extension elements or attributes, any such unknown content MUST be ignored by the diff client.

Each <document> element specifies changes in a specific document within the XCAP root. If several <document> elements pinpoint the same specific document, i.e., for example, the full entity tag (ETag) change history is indicated, the corresponding patches MUST be able to be applied in the given XCAP diff document order.

Note: This requirement simplifies applications that process XCAP diff documents since there's no need to sort patch instructions when applying them.

The <document> element has one mandatory attribute, "sel", and two optional attributes, "new-etag" and "previous-etag". The "sel" attribute of the <document> element identifies the specific document within the XCAP root for which changes are indicated. Its content MUST be a relative path reference, with the base URI being equal to the XCAP root URI. The "new-etag" attribute provides the entity tag

(ETag) for the document after the application of the changes, assuming the document exists after those changes. The "previous-etag" attribute provides an identifier for the document instance prior to the change. If the change being reported is the removal of a document, only the "previous-etag" MUST be included and the "new-etag" attribute MUST NOT be present. The "new-etag" attribute MUST only exist alone when the document either exists or it was just created (no patch included). Both attributes are present when a patch (or series of XCAP operations) has been applied to the resource. Also, both attributes MAY be used to indicate an ETag change without any document modifications (patches).

The "previous-etag" and "new-etag" need not have been sequentially assigned ETags at the server. An XCAP diff document can indicate changes that have occurred over a series of XCAP operations. The only requirement then is that the sequence of events, when executed serially, will result in the transformation of the document with the ETag "previous-etag" to the one whose ETag is "new-etag". Also, the series of operations do not have to be the same exact series of operations that occurred at the server.

Each <document> element contains either a sequence of patching instructions or an indication that the body hasn't semantically changed. The latter means that the document has been assigned a new ETag but its content is unchanged and it is indicated by the <body-not-changed> element. Patching instructions are described by the <add>, <replace>, and <remove> elements. These elements use the corresponding add, replace, and remove types defined in [RFC5261], and define a set of patch operations that can be applied to transform the locally cached document. See [RFC5261] for instructions on how this transformation is effected. The <document> element can also contain elements from other namespaces for the purposes of extensibility. The <add>, <replace>, and <remove> elements allow extension attributes from any namespace.

Figure 1 shows <document> element content and how the corresponding resource or metadata changes. In practice, an external document retrieval means HTTP GET requests for target resources. The asterisk character '*' means that a <document> element has child element(s): <add>, <replace>, or <remove>, or alternatively only a <body-not-changed> element. The hyphen character '-' means that the corresponding content (attribute or element) doesn't exist in a <document> element. The 'xxx' and 'yyy' are values of entity tags (ETag) of an XCAP document.

previous-etag	new-etag	<add> <replace> <remove>	<body-not-changed>	locally cached XCAP resource/ metadata change
xxx	yyy	*	-	resource patched, patch included
xxx	yyy	-	-	resource patched, external document retrieval
xxx	yyy	-	*	only ETag changed
-	yyy	-	-	resource created or exists, external document retrieval
xxx	-	-	-	resource removed

Figure 1: <document> element content / corresponding resource changes

Each <element> element indicates the existing element content of an XCAP document. It has one mandatory attribute, "sel", and optionally, an "exists" attribute and extension attributes from any namespace. The "sel" attribute of the <element> element identifies an XML element of an XCAP document. It is a percent-encoded relative URI following XCAP conventions when selecting elements. The XCAP Node Selector MUST always locate a unique node, the "exists" attribute thus shows whether an element exists or not in the XCAP document. When the "exists" attribute is absent from the <element> element, the indicated element still exists in the XCAP document. The located element exists as a child element of the <element> element. In a corner case where the content of this element cannot be presented for some reason (e.g., the payload is too large) although it exists in the XCAP document, the <element> element MUST NOT have any child nodes.

As the located XML element is typically namespace qualified, all needed namespace declarations MUST exist within the <xml-diff> document. The possible local namespace declarations within the located element exist unmodified as in the source document, similar to XCAP conventions. Other namespace references MUST be resolved from the context of the <element> or its parent elements. The

prefixes of qualified names (QNames) [W3C.REC-xml-names-20060816] of XML nodes also remain as they originally exist in the source XCAP document.

Each <attribute> element indicates the existing attribute content of an XCAP document. It has one mandatory attribute, "sel", and optionally, an "exists" attribute and extension attributes from any namespace. The "sel" attribute of the <attribute> element identifies an XML attribute of an XCAP document. It is a percent-encoded relative URI following XCAP conventions when selecting attributes. The "exists" attribute indicates whether or not an attribute exists in the XCAP document. When the "exists" attribute is absent from the <attribute> element, the indicated attribute still exists in the XCAP document. The child text node of the <attribute> element indicates the value of the located attribute. Note that if the attribute is namespace qualified, the query parameter of the XCAP URI indicates the attached namespace URI and the prefix in the XCAP source document.

Namespaces of the "sel" attribute of the <attribute> and <element> elements MUST also be resolved properly. Section 6.4. of [RFC4825] describes the rules when using namespace prefixes in XCAP Node Selectors. Without a namespace prefix in an element selector, an XCAP Default Document Namespace MUST be applied. The namespace resolving rules of Patch operation elements: <add>, <replace>, and <remove> are described in Section 4.2.1 of [RFC5261].

4. XML Schema

The XML Schema for the XCAP diff format.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
 xmlns="urn:ietf:params:xml:ns:xcap-diff"
 targetNamespace="urn:ietf:params:xml:ns:xcap-diff"
 elementFormDefault="qualified"
 attributeFormDefault="unqualified">

 <!-- include patch-ops -->
 <xs:include
  schemaLocation="urn:ietf:params:xml:schema:patch-ops"/>

 <!-- document root -->
 <xs:element name="xcap-diff">
  <xs:complexType>
   <xs:sequence minOccurs="0">
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
     <xs:choice>
```

```
        <xs:element name="document" type="documentType"/>
        <xs:element name="element" type="elementType"/>
        <xs:element name="attribute" type="attributeType"/>
       </xs:choice>
      </xs:sequence>
      <xs:any namespace="##other" processContents="lax"
              minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="xcap-root" type="xs:anyURI" use="required"/>
    <xs:anyAttribute processContents="lax"/>
   </xs:complexType>
  </xs:element>

  <!-- xcap document type -->
  <xs:complexType name="documentType">
   <xs:choice minOccurs="0">
    <xs:element name="body-not-changed" type="emptyType"/>
    <xs:sequence minOccurs="0" maxOccurs="unbounded">
     <xs:choice>
      <xs:element name="add">
       <xs:complexType mixed="true">
        <xs:complexContent>
         <xs:extension base="add">
          <xs:anyAttribute processContents="lax"/>
         </xs:extension>
        </xs:complexContent>
       </xs:complexType>
      </xs:element>
      <xs:element name="remove">
       <xs:complexType>
        <xs:complexContent>
         <xs:extension base="remove">
          <xs:anyAttribute processContents="lax"/>
         </xs:extension>
        </xs:complexContent>
       </xs:complexType>
      </xs:element>
      <xs:element name="replace">
       <xs:complexType mixed="true">
        <xs:complexContent>
         <xs:extension base="replace">
          <xs:anyAttribute processContents="lax"/>
         </xs:extension>
        </xs:complexContent>
       </xs:complexType>
      </xs:element>
      <xs:any namespace="##other" processContents="lax"/>
     </xs:choice>
```

```
      </xs:sequence>
     </xs:choice>
     <xs:attribute name="sel" type="xs:anyURI" use="required"/>
     <xs:attribute name="new-etag" type="xs:string"/>
     <xs:attribute name="previous-etag" type="xs:string"/>
     <xs:anyAttribute processContents="lax"/>
    </xs:complexType>

    <!-- xcap element type -->
    <xs:complexType name="elementType">
     <xs:complexContent mixed="true">
      <xs:restriction base="xs:anyType">
       <xs:sequence>
        <xs:any processContents="lax" namespace="##any"
                minOccurs="0" maxOccurs="1"/>
       </xs:sequence>
       <xs:attribute name="sel" type="xs:string"
                     use="required"/>
       <xs:attribute name="exists" type="xs:boolean"/>
       <xs:anyAttribute processContents="lax"/>
      </xs:restriction>
     </xs:complexContent>
    </xs:complexType>

    <!-- xcap attribute type -->
    <xs:complexType name="attributeType">
     <xs:simpleContent>
      <xs:extension base="xs:string">
       <xs:attribute name="sel" type="xs:string"
                     use="required"/>
       <xs:attribute name="exists" type="xs:boolean"/>
       <xs:anyAttribute processContents="lax"/>
      </xs:extension>
     </xs:simpleContent>
    </xs:complexType>

    <!-- empty type -->
    <xs:complexType name="emptyType"/>
   </xs:schema>
```

5. Example Document

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

```
<?xml version="1.0" encoding="UTF-8"?>
<d:xcap-diff xmlns:d="urn:ietf:params:xml:ns:xcap-diff"
             xmlns="urn:ietf:params:xml:ns:rls-services"
             xcap-root="http://xcap.example.com/root/">

 <d:document new-etag="7ahggs"
           sel="resource-lists/users/sip:joe@example.com/coworkers"
           previous-etag="8a77f8d"/>

 <d:element sel="rls-services/users/sip:joe@example.com/index/~~
/*/service%5b@uri='sip:marketing@example.com'%5d"
          xmlns:rl="urn:ietf:params:xml:ns:resource-lists"
    ><service uri="sip:marketing@example.com">
      <list name="marketing">
        <rl:entry uri="sip:joe@example.com"/>
        <rl:entry uri="sip:sudhir@example.com"/>
      </list>
      <packages>
        <package>presence</package>
      </packages>
    </service></d:element>

 <d:attribute
 sel="rls-services/users/sip:joe@example.com/index/~~/*/service/@uri"
      >sip:marketing@example.com</d:attribute>

</d:xcap-diff>
```

This indicates that the document with the URI "http://
xcap.example.com/root/resource-lists/users/sip:joe@example.com/
coworkers" has changed. Its previous entity tag is "8a77f8d" and its new one is "7ahggs", but actual changes are not shown. The <service> element exists in the rls-services "index" document and its full content is shown. Note that the <service> element is attached with a default namespace declaration within the original document. Similarly, "uri" attribute content is shown from the same "index" document as an illustrative example.

6. Basic Requirements for a System Exchanging XCAP Diff Documents

Documents at an XCAP server are identified by URIs, and updated by XCAP clients with HTTP (PUT and DELETE) methods. The XCAP server assigns a new entity tag value for each document version. An entity tag value is defined by Section 3.11 of RFC 2616 [RFC2616]: "An

entity tag MUST be unique across all versions of all entities associated with a particular resource". These entity tags are used to protect requests from making overriding changes when multiple XCAP clients update the same XCAP document. An entity tag value can be interpreted as a unique identifier to a specific version of an XCAP document in its change history.

The entity tag values of XCAP resources also enable a reliable way to update the locally cached XCAP resource copies in an XCAP diff implementation. When a diff client applies XCAP diff document changes, it MUST apply a resource state change only if entity tag values match with octet-by-octet equivalence according to the table defined in Figure 1. If a diff client notices inconsistencies and/or errors when it applies reported resource changes, it SHOULD tear down the session.

State changes of an XCAP document MUST be delivered reliably from a diff notifier to a diff client, and a diff client MUST be able to apply all changes of an XCAP document in the same chronological order that occurred at an XCAP server. When using an unreliable transport with retransmissions, the application protocol used with the XCAP diff MUST ensure that duplicates are dropped. If an XCAP diff delivery is lost, the diff session MUST be torn down. Note that a diff notifier can easily notice a lost notification when a diff client must respond to each XCAP diff delivery.

A diff notifier doesn't necessarily report all of these XCAP document updates with ETags; it MAY skip over some intermediate version of a document, for example, with rapidly changing resources. However, it MUST always report changes consistently to a diff client so that it can properly update the latest state (content and ETag) of its locally cached resources.

> As an example, an XCAP document is updated by different 'a', 'b', and 'c' versions identified with the same corresponding ETag values in a relatively short period. The first reported notification contains the 'a' "new-tag" information (no "previous-etag" attribute), and the diff notifier decides to skip the update notification identified by the 'b' ETag value. The second notification to a diff client MUST then contain the 'a' "previous-etag" and 'c' "new-etag" values with optional corresponding content changes (from version 'a' to 'c').

Since XCAP documents are typically confidential, diff notifiers MUST obey the XCAP authorization rules. In practice, this means following the read privilege rules of XCAP resources when notifying the authenticated diff clients of changes. Transport SHOULD be secured by encryption.

Note: This format specification doesn't define how to select the resources whose differences a diff notifier should report. It also doesn't define whether actual content changes should be reported. Typically, however, a diff client starts a session by sending a resource listing request. Then it compares the remote resource listings with locally cached ones, and probably downloads those resources that aren't locally cached or whose entity tags differ. When a diff client receives an XCAP diff with a "previous-etag" value that matches its current cached copy of a document, it can apply the diffs to the cached copy. As it takes some time to download reference documents, and diff notifications appear after actual resource state changes, several round trips may be needed before a full synchronization is achieved, especially with rapidly changing resources.

7. Security Considerations

XCAP diff documents can include changes from one version of a document to another version. As a consequence, if the document itself is sensitive and requires confidentiality, integrity, or authentication, then the same applies to the XCAP diff format. Therefore, protocols that transport XCAP diff documents must provide sufficient security capabilities for transporting the document itself. Confidential XCAP documents are typically transported using TLS-encrypted (Transport Layer Security) [RFC5246] communication; see RFC 4825 [RFC4825] for further security details.

When this format is used to report content changes of XCAP documents, all security considerations of RFC 5261 [RFC5261] apply. Very frequent updates of XCAP documents and/or many diff clients per subscribed resource impose a Denial-of-Service attack possibility to the servers processing XCAP diff documents. An efficient patch processing and throttling can, however, decrease the required overall processings and transactions.

The SIP event package framework specified in RFC 3265 [RFC3265] is the most typical use-case for this format. Then, an end-to-end SIP encryption mechanism, such as Secure/Multipurpose Internet Mail Extensions (S/MIME) described in Section 26.2.4 of RFC 3261 [RFC3261], SHOULD be used. If that is not available, it is RECOMMENDED that TLS [RFC5246] be used between elements to provide hop-by-hop authentication and encryption mechanisms as described in Section 26.2.2 ("SIPS URI Scheme") and Section 26.3.2.2 ("Interdomain Requests") of RFC 3261 [RFC3261]. Event packages MAY also have other specific threats that MUST be considered on an application-by-application basis.

8. IANA Considerations

There are several IANA considerations associated with this specification.

8.1. application/xcap-diff+xml MIME Type

MIME media type name: application

MIME subtype name: xcap-diff+xml

Mandatory parameters: none

Optional parameters: Same as the charset parameter application/xml as specified in RFC 3023 [RFC3023].

Encoding considerations: Same as the encoding considerations of application/xml as specified in RFC 3023 [RFC3023].

Security considerations: See Section 10 of RFC 3023 [RFC3023] and Section 7 of RFC 5874.

Interoperability considerations: none.

Published specification: This document.

Applications that use this media type: This document type has been used to support manipulation of resource lists [RFC4826] using XCAP.

Additional Information:

Magic Number: None

File Extension: .xdf

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.

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

This section registers a new XML namespace, as per the guidelines in [RFC3688].

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="http://www.rfc-editor.org/rfc/rfc5874.txt">RFC5874</a>.</p>
</body>
</html>
END
```

8.3. Schema Registration

This section registers a new XML schema per the procedures in [RFC3688].

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 4.

9. Acknowledgments

The authors would like to thank Pavel Dostal, Jeroen van Bemmel, Martin Hynar, Anders Lindgren, Mary Barnes, Ben Campbell, Francis Dupont, David Harrington, Alexey Melnikov, Dan Romascanu, and Robert Sparks for their valuable comments.

10. References

10.1. Normative References

[W3C.REC-xml-20060816]
Paoli, J., Bray, T., Yergeau, F., Maler, E., and C. Sperberg-McQueen, "Extensible Markup Language (XML) 1.0 (Fourth Edition)", World Wide Web Consortium FirstEdition REC-xml- 20060816, August 2006, <http://www.w3.org/TR/2006/REC-xml-20060816>.

[W3C.REC-xml-c14n-20010315]
Boyer, J., "Canonical XML Version 1.0", World Wide Web Consortium Recommendation REC-xml-c14n-20010315, March 2001, <http://www.w3.org/TR/2001/REC-xml-c14n-20010315>.

[W3C.REC-xml-names-20060816]
Hollander, D., Layman, A., and T. Bray, "Namespaces in XML 1.0 (Second Edition)", World Wide Web Consortium FirstEdition REC-xml-names-20060816, August 2006, <http://www.w3.org/TR/ 2006/REC-xml-names-20060816>.

[RFC2141] Moats, R., "URN Syntax", RFC 2141, May 1997.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC3023] Murata, M., St. Laurent, S., and D. Kohn, "XML Media Types", RFC 3023, January 2001.

[RFC2648] Moats, R., "A URN Namespace for IETF Documents", RFC 2648, August 1999.

[RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688, January 2004.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4825] Rosenberg, J., "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)", RFC 4825, May 2007.

[RFC5261] Urpalainen, J., "An Extensible Markup Language (XML) Patch Operations Framework Utilizing XML Path Language (XPath) Selectors", RFC 5261, September 2008.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

10.2. Informative References

[RFC5875] Urpalainen, J. and D. Willis, "An Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Diff Event Package", RFC 5875, May 2010.

[RFC2778] Day, M., Rosenberg, J., and H. Sugano, "A Model for Presence and Instant Messaging", RFC 2778, February 2000.

[RFC3261] 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.

[RFC3265] Roach, A., "Session Initiation Protocol (SIP)-Specific Event Notification", RFC 3265, June 2002.

[RFC4662] Roach, A., Campbell, B., and J. Rosenberg, "A Session Initiation Protocol (SIP) Event Notification Extension for Resource Lists", RFC 4662, August 2006.

[RFC4826] Rosenberg, J., "Extensible Markup Language (XML) Formats for Representing Resource Lists", RFC 4826, May 2007.

[RFC4483] Burger, E., "A Mechanism for Content Indirection in Session Initiation Protocol (SIP) Messages", RFC 4483, May 2006.

Appendix A. Informative Examples

These informative examples illustrate basic features of XCAP diff format.

The following documents exist at an XCAP server (xcap.example.com) with an imaginary "tests" application usage (there's no default document namespace defined in this imaginary application usage).

http://xcap.example.com/tests/users/sip:joe@example.com/index:

```
<?xml version="1.0" encoding="UTF-8"?>
<doc id="bar">
  <note>This is a sample document</note>
</doc>
```

and then

http://xcap.example.com/tests/users/sip:john@example.com/index:

```
<?xml version="1.0" encoding="UTF-8"?>
<doc>
  <note>This is another sample document</note>
</doc>
```

A.1. Indicating Existing, Changed, or Removed Documents

Firstly, an XCAP diff document can indicate what documents exist in a collection. An XCAP diff document may then be:

```
<?xml version="1.0" encoding="UTF-8"?>
<xcap-diff xmlns="urn:ietf:params:xml:ns:xcap-diff"
           xcap-root="http://xcap.example.com/">

 <document new-etag="7ahggs"
           sel="tests/users/sip:joe@example.com/index"/>

 <document new-etag="terteer"
           sel="tests/users/sip:john@example.com/index"/>

</xcap-diff>
```

This listing indicates current ETags of existing documents and their relative URIs.

Let's say that Joe adds a new document to his collection:

```
PUT /tests/users/sip:joe@example.com/another_document HTTP/1.1
Host: xcap.example.com
....
Content-Type: application/xml
Content-Length: [XXX]

<?xml version="1.0" encoding="UTF-8"?>
<doc>
  <note>This is another sample document</note>
</doc>
```

The requests result header has an HTTP ETag "terteer" for this new document.

Then an XCAP diff document may then indicate only the creation of this single new document:

```
<?xml version="1.0" encoding="UTF-8"?>
<xcap-diff xmlns="urn:ietf:params:xml:ns:xcap-diff"
           xcap-root="http://xcap.example.com/">

 <document new-etag="terteer"
           sel="tests/users/sip:joe@example.com/another_document"/>

</xcap-diff>
```

A "new-etag" without a "previous-etag" attribute indicates a creation of a new document.

Then Joe decides to modify an existing resource:

```
PUT /tests/users/sip:joe@example.com/another_document HTTP/1.1
Host: xcap.example.com
....
Content-Type: application/xml
Content-Length: [XXX]

<?xml version="1.0" encoding="UTF-8"?>
<doc>
  <note>This is a modified document</note>
</doc>
```

The reported new HTTP ETag is "huwiias".

Then an XCAP diff document may be:

```
<?xml version="1.0" encoding="UTF-8"?>
<xcap-diff xmlns="urn:ietf:params:xml:ns:xcap-diff"
           xcap-root="http://xcap.example.com/">

 <document previous-etag="terteer" new-etag="huwiias"
           sel="tests/users/sip:joe@example.com/another_document"/>

</xcap-diff>
```

Both "previous-etag" and "new-etag" attributes signal that a modification has happened to a resource, but actual changes are not shown.

Let's say that Joe then removes a document from his collection:

```
DELETE /tests/users/sip:joe@example.com/another_document HTTP/1.1
Host: xcap.example.com
```

This HTTP DELETE request results in the unlinking of the resource, and the XCAP diff may be:

```
<?xml version="1.0" encoding="UTF-8"?>
<xcap-diff xmlns="urn:ietf:params:xml:ns:xcap-diff"
           xcap-root="http://xcap.example.com/">

 <document previous-etag="huwiias"
           sel="tests/users/sip:joe@example.com/another_document"/>

</xcap-diff>
```

Thus, a "previous-etag" without a "new-etag" attribute indicates the removal of a resource.

A.2. Indicating Actual Changes of Documents

Secondly, XCAP diff documents are capable of showing actual changes to documents with [RFC5261] patching semantics.

Now Joe's XCAP client utilizes the XCAP patching capability to add a new element to a document:

```
PUT /tests/users/sip:joe@example.com/index/~~/doc/foo HTTP/1.1
Host: xcap.example.com
....
Content-Type: application/xcap-el+xml
Content-Length: [XXX]

<foo>this is a new element</foo>
```

Since the insertion of the element is successful, Joe's XCAP client receives the new HTTP ETag "fgherhryt3" of the updated "index" document.

Immediately thereafter, Joe's XCAP client issues another HTTP request (this request could even be pipelined):

```
PUT /tests/users/sip:joe@example.com/index/~~/doc/bar HTTP/1.1
Host: xcap.example.com
....
Content-Type: application/xcap-el+xml
Content-Length: [XXX]

<bar>this is a bar element
</bar>
```

The reported new HTTP ETag of "index" is now "dgdgdfgrrr".

And then Joe's XCAP client issues yet another HTTP request:

```
PUT /tests/users/sip:joe@example.com/index/~~/doc/foobar HTTP/1.1
Host: xcap.example.com
....
Content-Type: application/xcap-el+xml
Content-Length: [XXX]

<foobar>this is a foobar element</foobar>
```

The reported new ETag of "index" is now "63hjjsll".

XCAP diff format document may then indicate these XCAP component changes by:

```
<?xml version="1.0" encoding="UTF-8"?>
<d:xcap-diff xmlns:d="urn:ietf:params:xml:ns:xcap-diff"
             xcap-root="http://xcap.example.com/">

 <d:document previous-etag="7ahggs3"
             sel="tests/users/sip:joe@example.com/index"
             new-etag="63hjjsll">
  <d:add sel="*"
    ><foo>this is a new element</foo><bar>this is a bar element
</bar><foobar>this is a foobar element</foobar></d:add>
 </d:document>

</d:xcap-diff>
```

Note how several XCAP component modifications were aggregated together, and full history information got lost.

Alternatively, the content could have been:

```
<?xml version="1.0" encoding="UTF-8"?>
<d:xcap-diff xmlns:d="urn:ietf:params:xml:ns:xcap-diff"
           xcap-root="http://xcap.example.com/">

 <d:document previous-etag="7ahggs"
             sel="tests/users/sip:joe@example.com/index"
             new-etag="fgherhryt3">
   <d:add sel="*"
    ><foo>this is a new element</foo></d:add></d:document>

 <d:document previous-etag="fgherhryt3"
             sel="tests/users/sip:joe@example.com/index"
             new-etag="dgdgdfgrrr">
   <d:add sel="*"
    ><bar>this is a bar element
</bar></d:add></d:document>

 <d:document previous-etag="dgdgdfgrrr"
             sel="tests/users/sip:joe@example.com/index"
             new-etag="63hjjsll">
   <d:add sel="*"
    ><foobar>this is a foobar element</foobar></d:add></d:document>

</d:xcap-diff>
```

This shows the full ETag change history of a document, and ETags change chronologically in the reported XML document order.

A.3. Indicating XCAP Component Contents

Lastly, the XCAP diff format can also indicate the existing full contents of XCAP components, i.e., elements or attributes:

```
<?xml version="1.0" encoding="UTF-8"?>
<d:xcap-diff xmlns:d="urn:ietf:params:xml:ns:xcap-diff"
            xcap-root="http://xcap.example.com/">

 <d:attribute sel="tests/users/sip:joe@example.com/index/~~/doc/@id"
  >bar</d:attribute>

 <d:element sel="tests/users/sip:joe@example.com/index/~~/*/foo"
    ><foo>this is a new element</foo></d:element>

</d:xcap-diff>
```

Note that the HTTP ETag value of the new document is not shown as it is irrelevant for this use-case.

Then Joe's XCAP client removes the "id" attribute:

```
DELETE /tests/users/sip:joe@example.com/index/~~/doc/@id HTTP/1.1
Host: xcap.example.com
....
Content-Length: 0
```

And the XCAP diff document may then be:

```
<?xml version="1.0" encoding="UTF-8"?>
<xcap-diff xmlns="urn:ietf:params:xml:ns:xcap-diff"
            xcap-root="http://xcap.example.com/">

 <attribute sel="tests/users/sip:joe@example.com/index/~~/doc/@id"
  exists="0"/>

</xcap-diff>
```

This indicates that the subscribed attribute was removed from the document. The element content in this use-case may be discarded from the XCAP diff document, for example, when the size of XCAP diff document would be impractically large to the transport layer.

Authors' Addresses

Jonathan Rosenberg
jdrosen.net
Monmouth, NJ
US

EMail: jdrosen@jdrosen.net
URI: http://www.jdrosen.net

Jari Urpalainen
Nokia
Itamerenkatu 11-13
Helsinki 00180
Finland

Phone: +358 7180 37686
EMail: jari.urpalainen@nokia.com