SIMPLE
Internet-Draft
Expires: April 26, 2004

J. Rosenberg
dynamicsoft

October 27, 2003

Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Usages for Setting Presence Authorization
draft-ietf-simple-xcap-auth-usage-01

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on April 26, 2004.

Copyright Notice

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

Abstract

This document describes two usages of the Extensible Markup Language (XML) Configuration Access Protocol (XCAP) that allow a client to provide authorization decisions regarding watchers of their presence. The first of these usages, called permission-statements, contains statements about what permissions are to be granted to watchers of presence. The second usage, called supported-permissions, allows a client to determine what permissions are understood by the provider.

Table of Contents

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 [13], in order to learn their presence information [14]. This subscription is handed by a presence agent. In order to process the subscription, the presence agent must make a determination about whether the subscription is authorized. This authorization decision includes whether or not to accept the subscription, but also includes decisions about when the watcher should receive notifications, and when it does receive them, what the content of those notifications should be.

Typically, the authorization decision will be a combination of the authorization policies of the provider, combined with the authorization policices of the presentity. In order for the PA to compute the final authorization decision, it needs access to the presentity's authorization policies.

In order to provide this access, the XML Configuration Access Protocol (XCAP) [2] is used. XCAP allows a client to manipulate XML documents stored on a server. Those XML documents represent per-user provisioning data on how an application should operate. XCAP has the notion of an application usage, which is a definition of the XML schema used by a particular application, along with other relevant information. Each application usage is given a unique application usage ID (AUID) which identifies it. This specification makes use of three application usages.

2. Structuring Presence Authorization

This specification defines three application usages (each with their own XML schema) that, put together, present a comprehensive solution for allowing clients to specify authorization policies that a PA can use when processing a subscription. The first of these application usages has the AUID of permission-statements. This usage allows a client to make statements about which permissions are granted to which watchers. Each statement contains a definition of the watchers to whom it applies, and then contains a list of permissions which are granted to those watchers. The concept of a permission is central to this specification. A permission is an atomic statement of consent. A permission can indicate a condition under which a subscription is accepted or rejected, a condition under which a notification is or is not sent, or a piece of information which is revealed in a presence document. The overall authorization for a watcher is represented by the union of the permissions granted to that watcher.

This specification contains a basic set of primitive permissions. It is anticipated that new ones will be standardized in the future. It is also anticipated that vendors will define proprietary permissions. In order for a client to connect to a server, and achieve interoperability, it is neccesary for the client to know what permissions are supported by the server. The second application usage, supported-permissions, allows a client to read the list of permissions understood by the server.

3. 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.

4. Permission Statements

4.1 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 "permission-statements" AUID within the IETF tree, via the IANA registration in Section 6.

4.2 Structure of Permission Statements

A permission statement is an XML [3] document that MUST be well-formed and SHOULD be valid. Permission statement documents MUST be based on XML 1.0 and MUST be encoded using UTF-8. This

specification makes use of XML namespaces for identifying permission statement documents and document fragments. The namespace URI for elements defined for this purpose is a URN [5], using the namespace identifier 'ietf' defined by [7] and extended by [11]. This URN is:

urn:ietf:params:xml:ns:permission-statements

A permission statement document begins with the root element tag "permission-statements". It consists of any number of "statement" elements. Each statement element defines a set of permissions and identifies to whom they are granted.

Each "statement" element has a single attribute:

id: This is a string which serves as a way to uniquely identify statements in the document. The attribute MUST be unique amongst all statement elements in the document. This attribute is mandatory.

Each statement is composed of a single "applies-to" element and a single "permissions" element. The "permissions" element is composed of one or more elements that grant permissions.

4.2.1 Applying Statements to Watchers

The "applies-to" element defines the set of watchers to whom the statement applies. It contains one or more "uri" elements, "domain" elements, "on-list" elements or a single "any" element, followed by any number of "except" elements. The "uri" element identifies a single watcher by specifying its URI. The "domain" element says that the statement applies to all watchers from the specified domain. The "on-list" element says that the statement applies to all users on the specified presence list [17], identified with an HTTP URI that points to the list. Finally, the "any" element says that the statement

applies to all watchers. Additional elements can be added that express other ways of identifying the watchers to whom the statement applies. When unioned together, the result of the "uri", "domain", "on-list" and "any" elements define a set of users to whom the permission statement applies. This list is reduced in size by the "except" element, which removes a user or domain from the set. The "except" element contains instances of the "uri", "domain" and

"on-list" elements, which specify the users, domains, lists to be removed from the set.

The "uri", "domain", "on-list", and "any" elements all have the following attributes:

id: This is a string which serves as a way to uniquely identify an instance of this element within the enclosing "applies-to" element. The attribute MUST be unique amongst all elements of the same name within the enclosing "applies-to" element. This attribute is mandatory.

display-name: This is a string that contains a display name, suitable for rendering to a human user, the identity of the user or domain implied by the element. This attribute is optional.

lang: This attribute identifies the language used to represent the display name. It is imported from the XML namespace. This attribute is optional.

When a subscription arrives at the PA, the PA performs an authentication operation to determine the identity of the watcher. It then uses the "applies-to" element in each statement within the presentity's document, and determines the set of statements that apply to the watcher. It is possible that multiple statements can match a single subscription. In that case, the union of the permissions across those statements is applied to the subscription. It is also possible that none of the statements match, in which case the subscription is considered "pending".

For example, the following XML fragment includes two statements, one that applies to the user joe@example.com, and another that applies to example.com. When Joe subscribes, both statements match. Therefore, he is granted the union of the permissions across the two statements.

```
<?xml version="1.0" encoding="UTF-8"?>
<permission-statements>
  <statement id="as8f">
    <applies-to>
      <uri>sip:joe@example.com</uri>
    </applies-to>

    <permissions>
     <!-- permissions for joe go here -->
    </permissions>

  </statement>

  <statement id="kgg8a">
    <applies-to>
      <domain>example.com</domain>
    </applies-to>

    <permissions>
      <!-- permissions for example.com go here -->
    </permissions>
  </statement>

</permission-statements>
```

4.2.2 Specifying Permissions

The remainder of the content of the "statement" element contains specific permissions that are granted to watchers to whom the statement applies. Each permission is represented by a single XML element.

Primitive permissions can be grouped into two categories:

1. Acceptance permissions allow the watcher to subscribe. Without an acceptance permission, a subscription is rejected outright.

2. Content permissions indicate which information the watcher is permitted to see, in the event a notification is sent in the first place (based on the rule permissions).

4.2.2.1 Acceptance Permissions

Acceptance permissions grant the ability of the watcher to subscribe to the presentity. Without an acceptance permission, none of the other permissions make any sense. There are only two primitive

acceptance permissions, each of which is an XML element. These are "accept" and "accept-if". The "accept" element has no content and no attributes. It simply grants permission to the watcher to subscribe. Only one such element can be present in any statement. The "accept-if" element also grants permission to subscribe, but the granting of this permissions is predicated on some condition. The content of the "accept-if" element is a condition element. Condition elements describe characteristics of the subscription, or of the operating environment of the server, which are either true or false. If the condition within the "accept-if" element is true, an acceptance permission is granted.

The following represent the conditions which can be checked:

auth-mechanism: This element contains an enumerated type that describes authentication mechanisms. The defined values are none, digest (referring to the HTTP digest [8] mechanism used in RFC 3261 [9]), smime (referring to SIP's S/MIME authentication), tls (meaning that the watcher authenticated themself using a client certificate in a mutual TLS exchange with the server), and p-asserted-id (as defined in RFC 3325 [16]). The condition evaluates to true if the client was authenticated using the listed algorithm.

anonymous: This element contains no values. The condition evaluates to true if the watcher is anonymous. They are considered anonymous if the From header field of the request is equal to "Anonymous". Note that a user can be anonymous and also have authenticated themselves with digest. This occurs when the "anonymous" username and password, as defined in RFC 3261 [9], are used.

can-encrypt: This element contains no values. The condition evaluates to true if it is possible to encrypt, using S/MIME, notifications sent to this watcher. Generally, this can be determined when the Accept header field in the subscription indicates support for the application/pkcs7-mime [10] MIME type.

As an example, the following statement grants permission for watcher sip:joe@example.com to subscribe if he authenticates with digest:

```
<?xml version="1.0" encoding="UTF-8"?>
<permission-statements>
  <statement id="as8f">
    <applies-to>
      <uri>sip:joe@example.com</uri>
    </applies-to>

    <permissions
      <accept-if>
        <auth-mechanism>digest</auth-mechanism>
      </accept-if>
    </permissions>
  </statement>
</permission-statements>
```

4.2.2.2 Content Permissions

Content permissions specify the information that is to be sent to a watcher. Each permission specifies a piece of information that is to be sent, or to be used in general in the computation of the presence document. The defined permissions are:

all-content: This permission specifies that all presence information can be sent. The element has no attributes or value.

show-contact-element: This permission specifies that the contact component of the tuple can be sent. The element has no attributes or value.

show-note: This permission specifies that the note component of the tuple can be sent. The element has no attributes or value.

show-tuple: This permission specifies that the identified tuples can be sent to the watcher. The element has no attributes. Its content is a string that matches the "class" element present within the

tuple [12].

show-element: This permission specifies that the XML element identified by "show-element" can be presented to the watcher. The content of "show-element" is a qualified element name. When present, it that the specified element, if present in any published documents from publishers, can be used by the presence server and then distributed to watchers.

show-namespace: This permission specifies that elements and attributes in the presence document within the specified namespace can be presented to the watcher. When present, it that content

from the specified element, if present in any published documents from publishers, can be used by the presence server and then distributed to watchers.

encrypt: This permission specifies that the presence document should be sent to the watcher encrypted. It should never be present in a statement without the presence of an "accept-if" element which conditions acceptance of the subscription on the ability of the watcher to receive encrypted presence documents.

4.2.2.2.1 Examples

The following example specifies that a watcher is only allowed to see baseline pidf information:

```
<?xml version="1.0" encoding="UTF-8"?>
<permission-statements
  xmlns:pidf="urn:ietf:params:xml:ns:pidf">
  <statement id="as8f">
    <applies-to>
      <uri>sip:joe@example.com</uri>
    </applies-to>

    <permissions>
      <accept/>
      <show-namespace>urn:ietf:params:xml:ns:pidf</show-namespace>
    </permissions>
```

```
  </statement>
</permission-statements>
```

The following example shows that the watcher is allowed to see PIDF information along with the placetype element from RPIDS:


```
<?xml version="1.0" encoding="UTF-8"?>
<permission-statements
  xmlns:pidf="urn:ietf:params:xml:ns:pidf"
  xmlns:rpids="urn:ietf:params:xml:ns:sip-rpids">
  <statement id="as8f">
    <applies-to>
      <uri>sip:joe@example.com</uri>
    </applies-to>

    <permissions>
      <accept/>
      <show-namespace>urn:ietf:params:xml:ns:pidf</show-namespace>
      <show-element>rpids:placetype</show-element>
    </permissions>

  </statement>
</permission-statements>
```

4.3 Additional Constraints

The following are additional constraints not described by the schema:

- The content of a "show-element" element indicates the name of an XML element, and may be fully qualified (i.e., prefixed with a namespace identifier followed by a colon).

- The value of the "domain" element MUST be compliant to the BNF for "host" as defined in RFC 3261 [9].

- The value of the "on-list" element MUST be a valid HTTP URI that represents a presence list, as defined in [17].

- TODO: Complete this list.

4.4 Naming Conventions

When a presence agent receives a subscription for some user foo within a domain, it will look for all documents within http://[xcap root services uri]/permission-statements/users/foo, and use all documents found beneath that point to guide authorization policy.

4.5 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 priveleged users to modify documents

that they don't own, but the establishment and indication of such policies is outside the scope of this document.

4.6 XML Schema

]]>

TODOS: need to add points of extensibility.

5. Supported Permissions

Supported permissions allow a presentity to determine what the capabilities of the PA are, in terms of expressing authorization policy. This capability is expressed as a list of primitive permissions, primitive conditions, and compound permissions. When a client starts up, it reads this set of permissions from a well known URI (see Section 5.3). It then knows which permissions, both

primitive and compound, that it can include in its permission statements.

5.1 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 "supported-permissions" AUID within the IETF tree, via the IANA registration in Section 6.

5.2 Structure of Supported Permissions

A supported permission is an XML [3] document that MUST be well-formed and SHOULD be valid. Supported permission documents MUST be based on XML 1.0 and MUST be encoded using UTF-8. This specification makes use of XML namespaces for identifying supported permission documents and document fragments. The namespace URI for elements defined for this purpose is a URN [5], using the namespace identifier 'ietf' defined by [7] and extended by [11]. This URN is:

urn:ietf:params:xml:ns:supported-permissions

A supported permission document begins with the root element tag "supported-permissions". It consists of one "primitive-permissions" element, and zero or one "conditions" elements.

The "primitive-permissions" element has, for its content, a "permissions" element. This element contains a valid permission statement which purposefully includes all primitive permissions that are supported by the server. All PA's which allow for xcap-based configuration of authorization MUST support, at a minimum, the "accept", and "all-content" primitive permissions.

The "conditions" element contains a sequence of conditions which can be used within the "accept-if" element. Clearly, the "conditions" element will not be present if "accept-if" is not listed as a supported permission. There is no minimum requirement for a PA in terms of the conditions that need to be supported.

5.3 Naming Conventions

When a client starts, it can fetch the permissions understood by the server in one of two places. If the server capabilities differ on a user by user basis, the supported permissions for user foo can be found in http://[xcap root services uri]/supported-permissions/users/foo/sp.xml. A client SHOULD check this file first. If this document doesn't exist, the client should next check for the system wide permissions by checking http://[xcap root services uri]/supported-permissions/global/sp.xml.

5.4 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 priveleged users to modify documents that they don't own, but the establishment and indication of such policies is outside the scope of this document.

5.5 XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema
 targetNamespace="urn:ietf:params:xml:ns:supported-permissions"
 xmlns:ps="urn:ietf:params:xml:ns:permission-statements"
 xmlns:xs="http://www.w3.org/2001/XMLSchema"
 elementFormDefault="qualified"
 attributeFormDefault="unqualified">
  <xs:import namespace="urn:ietf:params:xml:ns:permission-statements"/>
  <xs:element name="supported-permissions">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="primitive-permissions">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="permissions" type="ps:permissionsType"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
        <xs:element name="conditions" minOccurs="0">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="condition" type="xs:string"
                maxOccurs="unbounded"/>
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

5.6 Example Document

This example document describes a PA that allows very simple primitive types. Instead, it defines several compound ones that are the preferred way for clients to express permissions.

```
<?xml version="1.0" encoding="UTF-8"?>
<supported-permissions
  xmlns:ps="urn:ietf:params:xml:ns:permission-statements"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

<primitive-permissions>
  <permissions>
    <accept/>
    <all-content/>
    <show-element>example</show-element>
  </permissions>
</primitive-permissions>
</supported-permissions>
```

6. IANA Considerations

There are several IANA considerations associated with this specification.

6.1 XCAP Application Usage IDs

This section registers three XCAP Application Usage IDs (AUID) according to the IANA procedures defined in [2].

6.1.1 Permission Statements

Name of the AUID: permission-statements

Description: Permission-statements are documents that describe the permissions that a presentity [13] has granted to users that seek to watch their presence.

6.1.2 Supported Permissions

Name of the AUID: supported-permissions

Description: Supported permissions are documents that describe the types of permissions which are supported by a presence agent [14]. These permissions specify the information that watchers [13] of presence are allowed to see.

6.2 URN Sub-Namespace Registrations

This section registers several new XML namespaces, as per the

guidelines in [11]

6.2.1 urn:ietf:params:xml:ns:permission-statements

URI: The URI for this namespace is
urn:ietf:params:xml:ns:permission-statements.

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>Permission Statements Namespace</title>
</head>
<body>
  <h1>Namespace for Permission Statements</h1>
  <h2>urn:ietf:params:xml:ns:permission-statements</h2>
  <p>See <a href="[[[URL of published RFC]]]">RFCXXXX</a>.</p>
</body>
</html>
END
```

6.2.2 urn:ietf:params:xml:ns:supported-permissions

URI: The URI for this namespace is
urn:ietf:params:xml:ns:supported-permissions.

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>Supported Permissions Namespace</title>
</head>
<body>
  <h1>Namespace for Supported Permissions</h1>
  <h2>urn:ietf:params:xml:ns:supported-permissions</h2>
  <p>See <a href="[[[URL of published RFC]]]">RFCXXXX</a>.</p>
</body>
</html>
END
```


6.3 XML Schema Registrations

This section registers three XML schemas as per the procedures in [11].

6.3.1 Permissions Statements

URI: please assign.

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 4.6.

6.3.2 Supported Permissions

URI: please assign.

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 5.5.

Normative References

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

[2] Rosenberg, J., "The Extensible Markup Language (XML) Configuration Access Protocol (XCAP)", draft-ietf-simple-xcap-00 (work in progress), June 2003.

[3] Bray, T., Paoli, J., Sperberg-McQueen, C. and E. Maler, "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C

REC REC-xml-20001006, October 2000.

[4] Clark, J. and S. DeRose, "XML Path Language (XPath) Version 1.0", W3C REC REC-xpath-19991116, November 1999.

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

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

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

[8] Franks, J., Hallam-Baker, P., Hostetler, J., Lawrence, S., Leach, P., Luotonen, A. and L. Stewart, "HTTP Authentication: Basic and Digest Access Authentication", RFC 2617, June 1999.

[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] Ramsdell, B., "S/MIME Version 3 Message Specification", RFC 2633, June 1999.

[11] Mealling, M., "The IETF XML Registry", draft-mealling-iana-xmlns-registry-05 (work in progress), June 2003.

[12] Schulzrinne, H., "RPID -- Rich Presence Information Data Format", draft-ietf-simple-rpid-00 (work in progress), July 2003.

Informative References

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

[14] Rosenberg, J., "A Presence Event Package for the Session Initiation Protocol (SIP)", draft-ietf-simple-presence-10 (work in progress), January 2003.

[15] Sugano, H. and S. Fujimoto, "Presence Information Data Format (PIDF)", draft-ietf-impp-cpim-pidf-08 (work in progress), May 2003.

[16] Jennings, C., Peterson, J. and M. Watson, "Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks", RFC 3325, November 2002.

[17] Rosenberg, J., "An Extensible Markup Language (XML) Configuration Access Protocol (XCAP) Usage for Presence Lists", draft-ietf-simple-xcap-list-usage-00 (work in progress), June 2003.

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.