

SIPPING
Internet-Draft
Intended status: Standards Track
Expires: August 28, 2008

H. Tschofenig
Nokia Siemens Networks
E. Leppanen
Individual
S. Niccolini
NEC
M. Arumaithurai
University of Goettingen
February 25, 2008

Completely Automated Public Turing Test to Tell Computers and Humans
Apart (CAPTCHA) based Robot Challenges for SIP
draft-tschofenig-sipping-captcha-01.txt

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with [Section 6 of BCP 79](#).

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/lid-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 August 28, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2008).

Abstract

A common approach to deal with unwanted communication attempts is to rely on some form of authorization policies, typically whitelists.

Internet-Draft CAPTCHA based Robot Challenges for SIP February 2008

In order to populate the entries in such an access control list it is helpful to have a way to challenge the entity willing to engage in a conversation (unless they are already pre-authorized). One reason why this is desired is to deal with robots that are aggressively distributing messages.

This document describes how "Completely Automated Public Turing Test to Tell Computers and Humans Apart" (CAPTCHA) tests, which require human interaction, are applied to SIP.

Internet-Draft CAPTCHA based Robot Challenges for SIP February 2008

Table of Contents

1.	Introduction	4
2.	Terminology	5
3.	UAC, UAS and Proxy Behavior	5
3.1.	Operation of a SIP Proxy or SIP UAS	5
3.2.	Operation of UAC	5
4.	Description of the CAPTCHA XML Document	6
4.1.	Structure of XML-Encoded CAPTCHA Challenge	6
4.2.	MIME Type for CAPTCHA Challenge Document	6
4.3.	The <challenge> Root Element	6
4.4.	The <media> Element	6
4.5.	The <uri> element	7
4.6.	The <data> element	7
4.7.	Values	7
5.	Syntax	8
6.	Example	9
7.	XML Schema	10
8.	Security Considerations	12
9.	IANA Considerations	12
9.1.	Captcha Header	12
9.2.	4xx Response	12
9.3.	Namespace	13
9.4.	Content-Type registration for 'application/captcha-challenge+xml'	13
9.5.	CAPTCHA Schema Registration	15
10.	Acknowledgments	15
11.	Alternative Solution Approaches	15
11.1.	Challenge by Proxy	15
11.1.1.	Overview	15
11.1.2.	Operation of Proxy when it issues a challenge directly	17
11.1.3.	Operation of UAC on receiving a CAPTCHA challenge from the SIP	17
11.2.	SIP request redirected by the SIP Proxy	17
11.2.1.	Overview	17

11.2.2.	Operation of Proxy when it redirects the INVITE to a CAPTCHA UA	20
11.2.3.	Operation of UAC when it receives a challenge from a CAPTCHA UA	20
11.3.	SIP Application Interaction Framework	20
12.	References	21
12.1.	Normative references	21
12.2.	Informative references	21
	Authors' Addresses	22
	Intellectual Property and Copyright Statements	24

[1.](#) Introduction

The problem of unwanted communication is an imminent challenge and only the combination of several techniques can provide some degree of protection. [\[RFC5039\]](#) provides four recommendations that should to be considered for an overall solution, namely,

- o Strong Identity
- o White Lists
- o Solve the Introduction Problem
- o Don't Wait Until its Too Late.

The human interaction required challenges are mainly used for solving the introduction problem targeting to handle requests from user agents with whom the recipient do not have former relations. For example, the challenge is initiated towards user agents that are not yet white or black-listed, or based on some other criteria.

The [\[I-D.tschofenig-sipping-framework-spit-reduction\]](#) provides a framework for dealing with unwanted communication. The policy contains rules that are applied to requests if the conditions of a given rule match. The actions of the matching rules are executed and one of the actions could be to provide a challenge that must be solved by a human before the request is forwarded to the called party triggering the corresponding user interface notifications to the user.

There are different techniques already developed for challenging user agents. "Completely Automated Public Turing Test to Tell Computers

and Humans Apart" (CAPTCHA) [[captcha](#)] typically provides a human a task either to recognize something or a question to be answered using different media types. [[Inaccessibility-of-CAPTCHA](#)] provides alternatives to visual test for allowing systems to test for human users while preserving access by users with disabilities. Hashcash challenge [[hashcash](#)] requires user agents to perform CPU-intensive computational puzzles making it difficult to send large amounts of requests. The hashcash concept has been proposed for usage with SIP in [[I-D.jennings-sip-hashcash](#)].

Using CAPTCHA techniques for SIP communication requires a mechanism for enabling user interaction to be associated with SIP requests. When a proxy or user agent server (UAS) server receives a SIP request that needs to be challenged, the proxy or UAS sends a challenge to the originator of the SIP request before continue handling of the request. After getting the answer to the challenge from the user, the user agent client (UAC) needs to provide the answer back towards the UAS in order to get the initial request passed to the recipient.

The challenge should offer multiple choices for the UACs to select depending on the capabilities of the device where the UAC is running. Also, the UAC should be able to authenticate and authorize the source of challenge. The UAC may receive the challenge via a URL or as direct media compoment(s).

The main goal is to support SIP dialog creating request such as SIP INVITE, but ideally the solution should also cover non-dialog creating requests, e.g., SIP MESSAGE.

Note that this document presents several different solution approaches, see [Section 11](#). The solution presented in the main part of the document is aligned with the work done with [[XEP-0158](#)] on CAPTCHAs for XMPP.

[2](#). Terminology

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

implementations.

This document makes also use of the vocabulary defined in [RFC 3261](#) [[RFC3261](#)].

[3.](#) UAC, UAS and Proxy Behavior

[3.1.](#) Operation of a SIP Proxy or SIP UAS

When a SIP proxy or a SIP UAS receives a SIP request from a UAC, its authorization engine may apply the policy to the SIP request, as, for example, defined in [[RFC5025](#)]. This authorization policy execution may result in the need for the proxy (or the UAS) to generate a challenge to the UAC, the proxy (or the UAS) can send the challenge directly, can send a URI of the challenge, or can redirect the request to a special CAPTCHA UA.

[3.2.](#) Operation of UAC

The UAC either receives a CAPTCHA challenge or a URI of the challenge. The UAC is expected to solve the CAPTCHA puzzle and send the answer back to the SIP proxy server or to send a token to indicate that it has successfully solved the puzzle.

[4.](#) Description of the CAPTCHA XML Document

This section describes the content of the CAPTCHA XML document. The XML schema for it can be found in [Section 7](#).

[4.1.](#) Structure of XML-Encoded CAPTCHA Challenge

A CAPTCHA challenge is an XML document [[XML](#)] that MUST be well-formed and MUST be valid according to the schema defined in this document, including extension schemas available to the validator and applicable to the XML document. The XML documents MUST be based on XML 1.0 and MUST be encoded using UTF-8.

The namespace identifier 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:captcha.

[4.2.](#) MIME Type for CAPTCHA Challenge Document

The MIME type for the XML document is 'application/captcha-challenge+xml'.

[4.3.](#) The <challenge> Root Element

The root element of the XML document is <challenge>.

The <challenge> element contains the namespace definition mentioned in [Section 4.1](#). It also contains a mandatory 'id' attribute for correlating the challenge and the answer, and the 'min-tests' attribute with the default value set to 1. With the 'min-tests' attribute, it is possible to define the minimum amount of tests that need to be solved.

The <challenge> element MUST have at least one child element. This document defines the <media> element as a child element. The <challenge> element may contain one or more <media> elements.

The <challenge> element may also be extended by XML elements or attributes defined with other namespaces.

[4.4.](#) The <media> Element

The <media> element contains one child element. This document defines the <uri> and <data> elements as child elements for allowing the CAPTCHA challenge be provided directly as content or as a reference to an external content.

The <media> element contains a mandatory 'var' attribute indicating the type of the challenge (see values from the 'var' column of Figure 1). It may also contain optional 'width' and 'height' attributes for providing the size of the content. In addition, the element may contain an 'instr' attribute which purpose is to provide instructions related to the challenge (see the 'example generic instruction' column from Figure 1). The required tests can be indicated by setting the value of the 'required' attribute to 'true'.

The <media> element may also be extended by XML elements or attributes defined with other namespaces.

[4.5.](#) The <uri> element

The <uri> element contains a mandatory 'type' attribute indicating the MIME type of the challenge. See values from the 'MIME type' column of Figure 1. The value of the <uri> element is a URL where the challenge can be fetched.

The <uri> element may also be extended by XML attributes defined with other namespaces.

[4.6.](#) The <data> element

The <data> element contains a mandatory 'type' attribute indicating the MIME type of the challenge. See typical values from the 'MIME type' column of Figure 1.

The value of the <data> element is the content of the challenge.

The <data> element may also be extended by XML attributes defined with other namespaces.

[4.7.](#) Values

The following table copied from [[XEP-0158](#)] presents typical values for the CAPTCHA challenge. The 'var' column lists values for the 'var' attribute of the <media> element. The 'MIME type' column contains values of the corresponding 'type' attribute of the <uri> or <data> elements.

'var'	Name	Media type	MIME type	Example generic instructions
ocr*	Optical Char Recognition	image	image/jpeg	Enter the code you see
picture_recog	Picture Recognition	image	image/jpeg	Describe the picture
video_recog	Video Recognition	video	video/mpeg	Describe the video
speech_recog	Speech Recognition	audio	audio/x-wav	Enter the words you hear
audio_recog	Audio Recognition	audio	audio/x-wav	Describe the sound you hear
picture_q	Picture Question	image	image/jpeg	Answer the question you see
video_q	Video Question	video	video/mpeg	Answer the question in video
speech_q	Speech Question	audio	audio/x-wav	Answer the question you hear
qa	Text Q & A	text	text/plain	Answer the question

* The image portrays random characters that humans can read but OCR software cannot. To pass the challenge, the user must simply type the characters. The correct answer SHOULD NOT depend on the language specified by the 'xml:lang' attribute of the challenge.

Figure 1: Information of CAPTCHA challenges

5. Syntax

The Captcha header field carries the solution information. It has parameters called 'id' and 'answer'. The 'id' parameter value is set to the same as the 'id' attribute of the CAPTCHA challenge sent to the UAC. The 'answer' parameter value is set to the answer of the CAPTCHA challenge.

Example:

Captcha: id="rjffe32"; answer="2";

The ABNF for the header is:

```
Captcha      = "Captcha" HCOLON captcha-param *(COMMA captcha-param)
captcha-param = captcha-id SEMI captcha-answer *(SEMI generic-param)
captcha-id    = "id" EQUAL quoted-string
captcha-answer = "answer" EQUAL quoted-string
```

This document updates the Table 2 of [[RFC3261](#)] by adding the following:

Header field -----	where -----	proxy -----	ACK ---	BYE ---	CAN ---	INV ---	OPT ---	REG ---
Captcha	R	dr	o	o	-	o	o	o
			SUB ---	NOT ---	REF ---	INF ---	UPD ---	PRA ---
			o	o	o	o	o	o

[6.](#) Example

The following XML document shows the content that is provided of a CAPTCHA the challenge message sent towards the sending party as shown in message (2) of Figure 12.

Internet-Draft CAPTCHA based Robot Challenges for SIP February 2008

```
<?xml version="1.0" encoding="UTF-8"?>
<challenge xmlns="urn:ietf:params:xml:ns:captcha"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  id="73DE28A2">

  <media var="urn:ietf:params:xml:ns:captcha:ocr"
    width="290" height="80">
    <uri type="image/jpeg">
      http://www.example.com/challenges/ocr.jpeg?F3A6292C
    </uri>
  </media>

  <media var="urn:ietf:params:xml:ns:captcha:audio_recog">
    <uri type="audio/x-wav">
      http://www.example.com/challenges/audio.wav?F3A6292C
    </uri>
  </media>

  <media var="urn:ietf:params:xml:ns:captcha:qa">
    <data type="text/plain">Type the color of a stop light</data>
  </media>

</challenge>
```

7. XML Schema

This document defines the XML Schema based on the schema defined in Section 12 of [[XEP-0158](#)].

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  targetNamespace="urn:ietf:params:xml:ns:captcha"
  xmlns="urn:ietf:params:xml:ns:captcha"
  elementFormDefault="qualified">

  <xs:element name="challenge" type="challengeType"/>
```

```

<xs:complexType name="challengeType">
  <xs:complexContent>
    <xs:restriction base="xs:anyType">
      <xs:sequence>
        <xs:element ref="media"
          minOccurs="1" maxOccurs="unbounded"/>
        <xs:any namespace="##other"
          minOccurs="0" processContents="lax"/>

```

```

      </xs:sequence>
      <xs:attribute name="id"
        use="required" type="xs:string"/>
      <xs:attribute name="min_tests" type="xs:unsignedInt"
        default="1" use="optional" />

    </xs:restriction>
  </xs:complexContent>
</xs:complexType>

<xs:element name="media" type="mediaType"/>

<xs:complexType name="mediaType">
  <xs:complexContent>
    <xs:restriction base="xs:anyType">
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element ref="uri"
          minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="data"
          minOccurs="0" maxOccurs="unbounded"/>
        <xs:any namespace="##other" minOccurs="0"
          processContents="lax"/>
      </xs:choice>
      <xs:attribute name="var"
        use="required" type="xs:anyURI"/>
      <xs:attribute name="required" type="xs:boolean"
        default="false" use="optional"/>
      <xs:attribute name="height"
        type="xs:string" use="optional"/>
      <xs:attribute name="width"
        type="xs:string" use="optional"/>
      <xs:attribute name="instr"

```

```

        type="xs:string" use="optional"/>
        <xs:anyAttribute namespace="##any"
            processContents="lax"/>
    </xs:restriction>
</xs:complexContent>
</xs:complexType>

<xs:element name="uri">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="type" use="required"/>
                <xs:anyAttribute namespace="##any"
                    processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:element>

```

```

    </xs:complexType>
</xs:element>

<xs:element name="data">
    <xs:complexType>
        <xs:simpleContent>
            <xs:extension base="xs:string">
                <xs:attribute name="type" use="required"/>
                <xs:anyAttribute namespace="##any"
                    processContents="lax"/>
            </xs:extension>
        </xs:simpleContent>
    </xs:complexType>
</xs:element>

</xs:schema>

```

[8.](#) Security Considerations

[Editor's Note: A future version of this document will describe security considerations.]

[9.](#) IANA Considerations

This specification registers a new header and a new response code. IANA is requested to make the following updates in the registry at: <http://www.iana.org/assignments/sip-parameters>. It also registers a new namespace and a content type.

[9.1.](#) Captcha Header

Add the following entry to the header sub-registry.

Header Name	compact	Reference
-----	-----	-----
Captcha		[RFC-XXXX]

[9.2.](#) 4xx Response

Add the following entry to the response code sub-registry under the "Request Failure 4xx" heading.

4xx CAPTCHA required [RFC-XXXX]

[9.3.](#) Namespace

This section registers a new XML namespace per the procedures in [\[RFC3688\]](#).

URI: urn:ietf:params:xml:ns:captcha

Registrant Contact: IETF SIPPING Working Group, Hannes Tschofenig
(hannes.tschofenig@nsn.com).

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>Namespace for CAPTCHA Challenge</title>
</head>
<body>
  <h1>Namespace for providing CAPTCHA challenge</h1>
  <h2>urn:ietf:params:xml:ns:captcha</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
```

[9.4.](#) Content-Type registration for 'application/captcha-challenge+xml'

This specification requests the registration of a new MIME type according to the procedures of [RFC 2048](#) [[RFC2048](#)] and guidelines in [RFC 3023](#) [[RFC3023](#)].

MIME media type name: application

MIME subtype name: captcha-challenge+xml

Mandatory parameters: none

Optional parameters: charset

Indicates the character encoding of enclosed XML. Default is UTF-8.
Encoding considerations:

Uses XML, which can employ 8-bit characters, depending on the character encoding used. See [RFC 3023](#) <xref target="[RFC3023](#)">, [Section 3.2](#).

Security considerations:

This content type is designed to carry challenges for the user agent clients to solve in order to give a proof of being a human behind the generated request. This action is a part of a spam preventing mechanism. Appropriate precautions should be adopted to limit disclosure of this information. Please refer to RFCXXXX [NOTE TO IANA/RFC-EDITOR: Please replace XXXX with the RFC number of this specification.] Security Considerations section for more information.

Interoperability considerations: none

Published specification: RFCXXXX [NOTE TO IANA/RFC-EDITOR: Please replace XXXX with the RFC number of this specification.] this document

Applications which use this media type: SIP applications

Additional information:

Magic Number: None
File Extension: .xml
Macintosh file type code: 'TEXT'

Personal and email address for further information: Hannes
Tschofenig, Hannes.Tschofenig@nsn.com
Intended usage: LIMITED USE

Author/Change controller:

This specification is a work item of the IETF SIPPING working group, with mailing list address <xxxxx@ietf.org>.

[9.5.](#) CAPTCHA Schema Registration

URI: urn:ietf:params:xml:schema:captcha
Registrant Contact: IETF SIPPING Working Group, Hannes Tschofenig

(Hannes.Tschofenig@nsn.com).

XML: The XML schema to be registered is contained in [Section 7](#). Its first line is

```
<?xml version="1.0" encoding="UTF-8"?>
```

and its last line is

```
</xs:schema>
```

[10](#). Acknowledgments

Years ago CAPTCHAs have been introduced for XMPP, see 'XEP-0158: Robot Challenges' [[XEP-0158](#)]. The authors of this document believe that there is value in re-using it for SIP for Spam prevention. Hence, the authors would like to thank the XMPP community for their work on this subject. In particular, all credits go to Ian Paterson (ian.paterson@clientside.co.uk), the author of [[XEP-0158](#)].

We would like to thank Jonathan Rosenberg for his feedback to this draft.

[11](#). Alternative Solution Approaches

This section shows alternative solution approaches that can be used by a proxy to perform CAPTCHA tests.

[11.1](#). Challenge by Proxy

[11.1.1](#). Overview

Figure 11 and Figure 12 present high level messages flows for conveying a challenge (e.g., CAPTCHA) to the SIP UAC that initiated a dialog forming SIP request. In Figure 11 the challenge is included in the body of the SIP 4xx response while Figure 12 describes a case when the challenge is fetched via an URL that was provided with the response. After the user has managed to solve the challenge the UAC re-issues the request with the solution. The proxy removes the solution before forwarding the request to the SIP UAS.

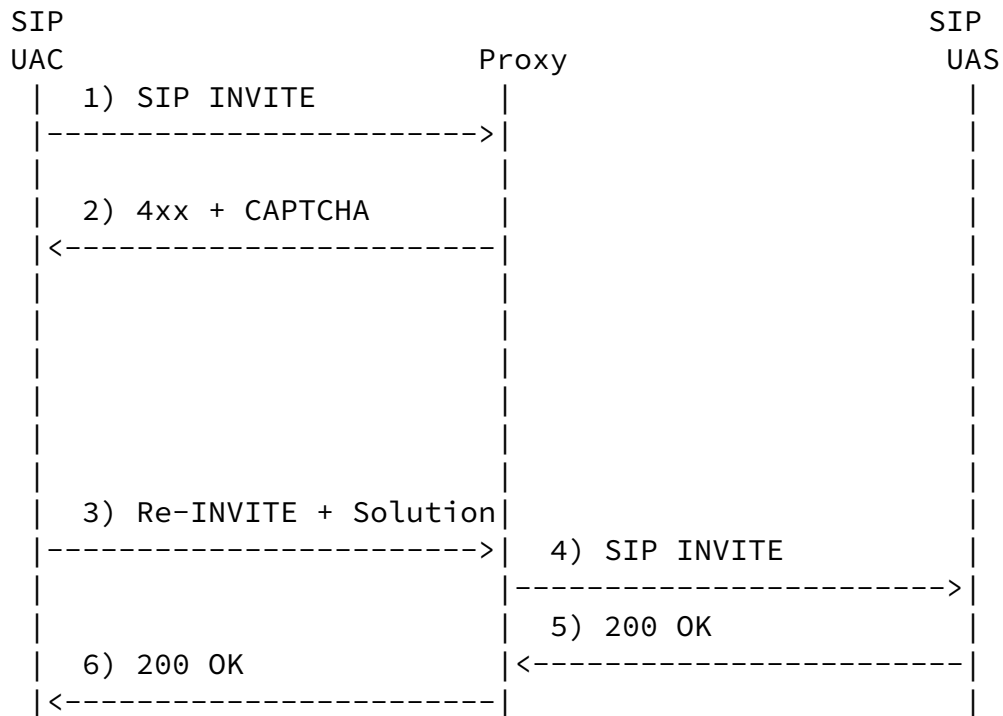


Figure 11: Proxy returns the CAPTCHA directly with the response

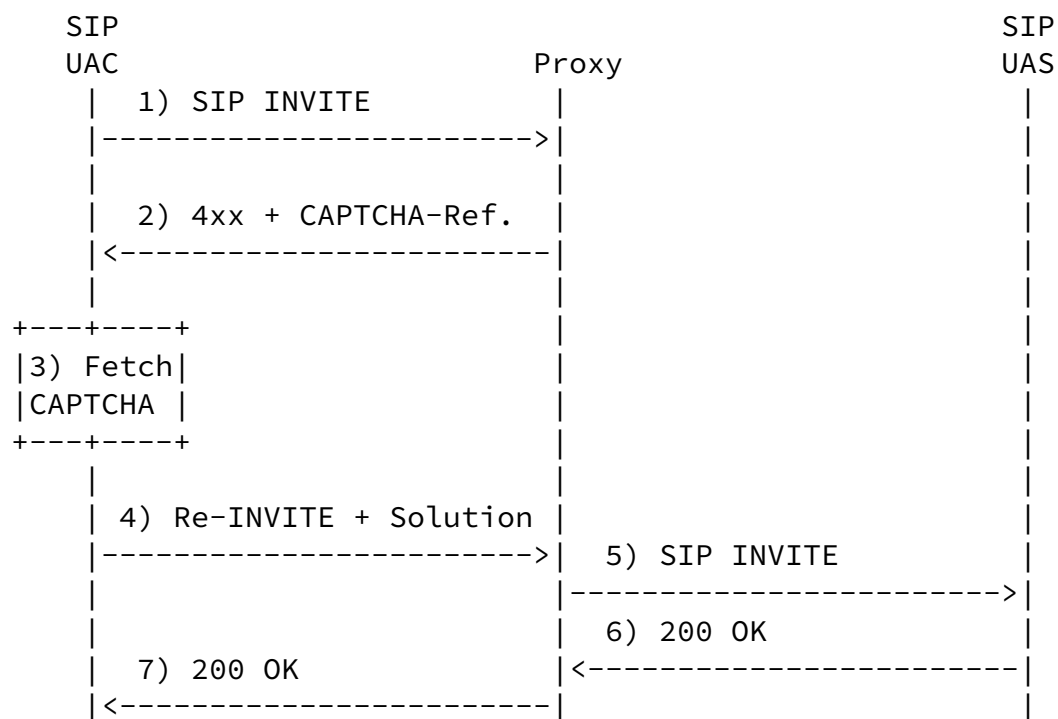


Figure 12: Proxy returns URL to the CAPTCHA

[11.1.2.](#) Operation of Proxy when it issues a challenge directly

The proxy sends a 4xx response with an XML document containing the challenge in the body. The Content-Type used for the XML document is 'application/captcha-challenge+xml'.

When the proxy receives a re-issued SIP request from the UAC, it validates the answer provided by the UAC in the CAPTCHA header field. In case the answer and other possible policies allow the request to get proxied further to the UAS, the proxy removes the CAPTCHA header. Depending on the policies and functionality of the proxy, the proxy may update the authorization policy according to the decision, e.g., insert the AoR of the user of the UAC to a white or black list. In case the answer was not satisfactory, the UAS acts according to a defined policy, e.g., rejects the request.

[11.1.3.](#) Operation of UAC on receiving a CAPTCHA challenge from the SIP

When the UAC receives a 4xx response with a MIME type 'application/captcha-challenge+xml' in the body to be solved, the UAC first authenticates and authorizes the sender of the challenge.

The UAC selects the challenges marked as mandatory and possibly some additional ones for UAC's execution or to be rendered to the user based on, e.g., the device capabilities. The UAC may also need to fetch the challenges from which URL links were provided. When the challenge gets solved, the UAC provides an answer in the CAPTCHA header field by re-issuing the SIP request, e.g., by sending a SIP re-INVITE.

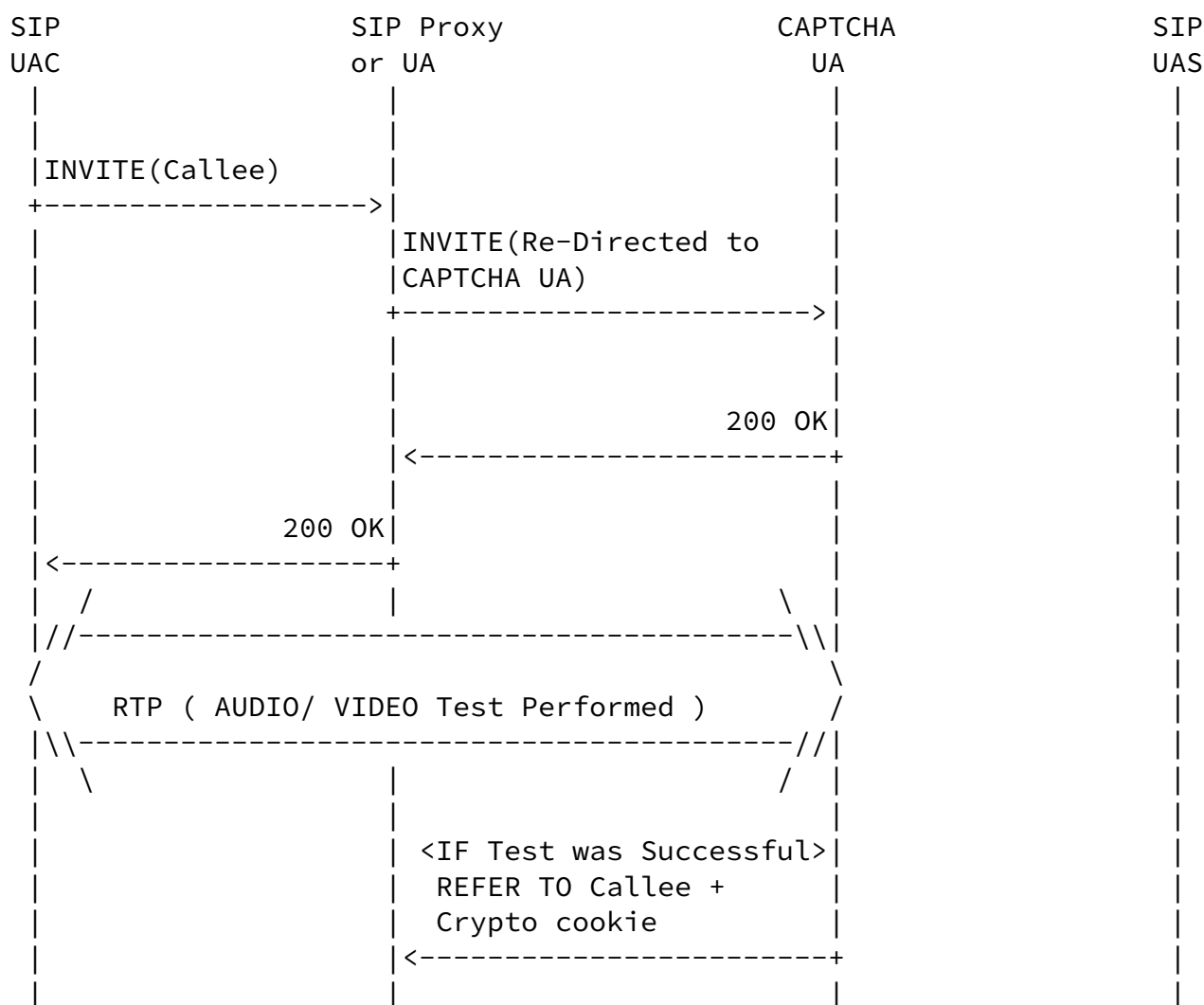
[11.2.](#) SIP request redirected by the SIP Proxy

[11.2.1.](#) Overview

In this case, the SIP proxy redirects the INVITE from a SIP UAC to a CAPTCHA UAS. The CAPTCHA UA acknowledges the request for service and then, contacts the SIP UAC directly to issue the challenge. On performing the CAPTCHA tests, it intimates the SIP server of the result.

The redirect of the INVITE by the SIP server to a CAPTCHA UA is a simple call redirect, negotiation of the parameters for the CAPTCHA is done using the standard SDP negotiation. From the caller point of view this is just a call setup, the caller will be presented the CAPTCHA test depending on the media it supports (audio, video, text). In this way there is no need for additional signaling that would reveal the caller that a CAPTCHA needs to be solved.

Figure 13 presents a high level message flow showing a successful CAPTCHA test and Figure 14 presents a high level message flow conveying a unsuccessful CAPTCHA challenge by a UA.



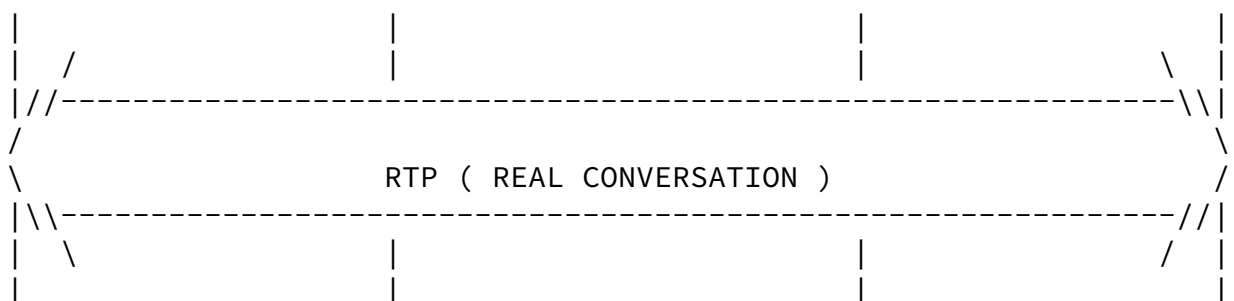
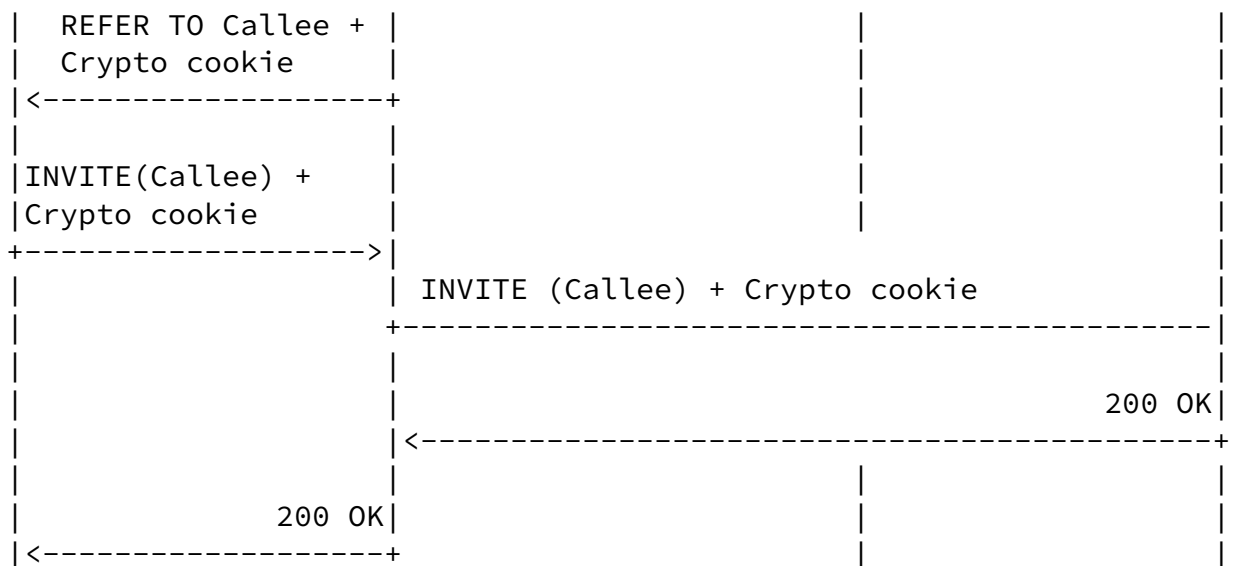
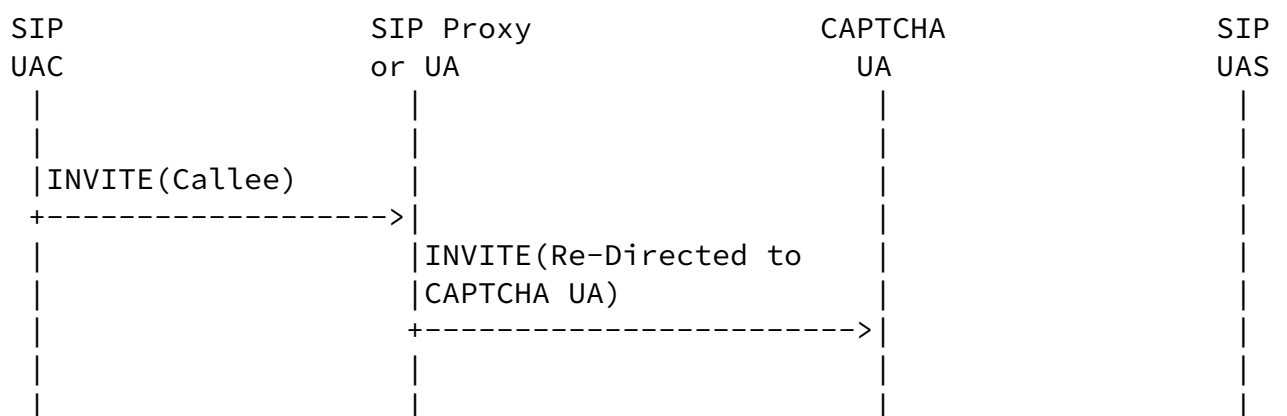


Figure 13: A case where the Proxy redirects the INVITE to a CAPTCHA UA and gets a SUCCESS response



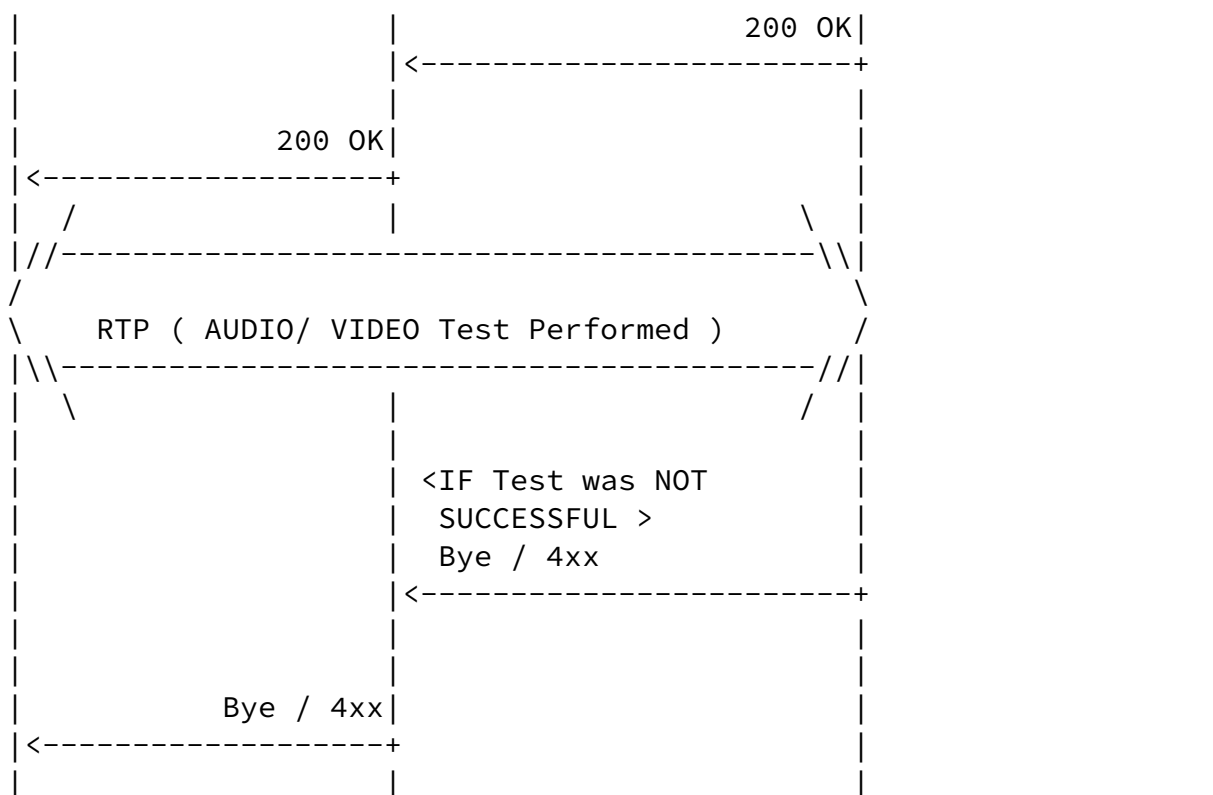


Figure 14: A case where the Proxy redirects the INVITE to a CAPTCHA UA and gets a NOT SUCCESS response

[11.2.2.](#) Operation of Proxy when it redirects the INVITE to a CAPTCHA UA

The SIP server redirects the INVITE to a CAPTCHA UA. The CAPTCHA UA, acknowledges the request for service by sending a "200 OK" message. The CAPTCHA UA, then proceeds to issue the CAPTCHA challenge to the user.

If the user is successful in solving the CAPTCHA challenge, the CAPTCHA UA issues a reference to the Callee along with crypto cookie to ensure that a replay attack isn't possible. The SIP server passes this information to the SIP UAC. The SIP UAC issues a new INVITE along with the obtained crypto cookie. Figure 13 presents the message flow.

If the user is not successful in solving the CAPTCHA challenge, the CAPTCHA UA issues a Bye message or a 4xx RESPONSE with an appropriate

error message. Figure 14 presents the message flow.

[11.2.3.](#) Operation of UAC when it receives a challenge from a CAPTCHA UA

When the UAC receives a challenge from a CAPTCHA UA, the UAC selects the challenges marked as mandatory and possibly some additional ones for UAC's execution or to be rendered to the user based on e.g. the device capabilities. When the challenge gets solved, the UAC provides an answer to the CAPTCHA UA.

[11.3.](#) SIP Application Interaction Framework

[I-D.ietf-sipping-app-interaction-framework] defines a framework for interaction between users and SIP based applications. The framework covers both the "presentation capable" and "presentation free" user interfaces (UI) having different solutions to both. The user interaction with the presentation capable UI is handled by using SIP REFER and HTTP while the presentation free UI case utilize SIP events [[RFC3265](#)] (SIP SUBSCRIBE and NOTIFY). Since there are different solutions for different cases, the UAC needs to indicate the supported application user interaction mechanisms when issuing a SIP request.

[12.](#) References

[12.1.](#) Normative references

- [RFC2048] Freed, N., Klensin, J., and J. Postel, "Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures", [BCP 13](#), [RFC 2048](#), November 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC2141] Moats, R., "URN Syntax", [RFC 2141](#), May 1997.
- [RFC2648] Moats, R., "A URN Namespace for IETF Documents", [RFC 2648](#),

August 1999.

- [RFC3023] Murata, M., St. Laurent, S., and D. Kohn, "XML Media Types", [RFC 3023](#), January 2001.
- [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.
- [RFC3688] Mealling, M., "The IETF XML Registry", [BCP 81](#), [RFC 3688](#), January 2004.
- [XML] Bray, T., "Extensible Markup Language (XML) 1.0 (Second Edition)", W3C CR CR-xml11-20011006, October 2000.

[12.2.](#) Informative references

- [I-D.ietf-sipping-app-interaction-framework]
Rosenberg, J., "A Framework for Application Interaction in the Session Initiation Protocol (SIP)",
[draft-ietf-sipping-app-interaction-framework-05](#) (work in progress), July 2005.
- [I-D.jennings-sip-hashcash]
Jennings, C., "Computational Puzzles for SPAM Reduction in SIP", [draft-jennings-sip-hashcash-06](#) (work in progress), July 2007.
- [I-D.tschofenig-sipping-framework-spit-reduction]
Tschofenig, H., Schulzrinne, H., Wing, D., Rosenberg, J., and D. Schwartz, "A Framework to tackle Spam and Unwanted Communication for Internet Telephony",
[draft-tschofenig-sipping-framework-spit-reduction-02](#) (work in progress), November 2007.

- [Inaccessibility-of-CAPTCHA]
May, M., "Inaccessibility of CAPTCHA; Alternatives to Visual Turing Tests on the Web",
html <http://www.w3.org/TR/turingtest/>, November 2005.

- [RFC3265] Roach, A., "SIP-Specific Event Notification", [RFC 3265](#), June 2002.
- [RFC5025] Rosenberg, J., "Presence Authorization Rules", [RFC 5025](#), December 2007.
- [RFC5039] Rosenberg, J. and C. Jennings, "The Session Initiation Protocol (SIP) and Spam", [RFC 5039](#), January 2008.
- [XEP-0158] Paterson, I., "XEP-0158: Robot Challenges", [html http://wiki.jabber.org/index.php/Robot](http://wiki.jabber.org/index.php/Robot) Challenges (XEP-0158), October 2006.
- [captcha] von Ahn, L., Blum, M., and J. Langford, "Telling Humans and Computers Apart Automatically", [html http://www.captcha.net](http://www.captcha.net), February 2004.
- [hashcash] Back, A., "Hashcash - A Denial of Service Counter-Measure", [html http://hashcash.org](http://hashcash.org), August 2002.

Authors' Addresses

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@nsn.com
URI: <http://www.tschofenig.com>

Eva Leppanen
Individual
Finland

Email: eva.leppanen@hukassa.com

Saverio Niccolini
NEC Laboratories Europe, NEC Europe Ltd.
Kurfuersten-Anlage 36
Heidelberg 69115
Germany

Phone: +49 (0) 6221 4342 118
Email: saverio.niccolini@nw.neclab.eu
URI: <http://www.nw.neclab.eu>

Mayutan Arumaithurai
University of Goettingen

Email: mayutan.arumaithurai@gmail.com
URI: <http://www.mayutan.org>

Internet-Draft CAPTCHA based Robot Challenges for SIP February 2008

Full Copyright Statement

Copyright (C) The IETF Trust (2008).

This document is subject to the rights, licenses and restrictions contained in [BCP 78](#), and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM 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.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights 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; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in [BCP 78](#) and [BCP 79](#).

Copies of IPR disclosures made to the IETF Secretariat 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 implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).