XEP-0158: CAPTCHA Forms

Ian Paterson
mailto:ian.paterson@clientside.co.uk
xmpp:ian@zoofy.com

Peter Saint-Andre
mailto:stpeter@stpeter.im
xmpp:stpeter@jabber.org
https://stpeter.im/

2019-11-07
Version 1.0.1

Status	Type	Short Name
Draft	Standards Track	captcha

This document specifies an XMPP protocol extension that entities may use to discover whether the sender of an XML stanza is a human user or a robot.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

The appearance of large public IM services based on XMPP Core [1] and XMPP IM [2] makes it desirable to implement protocols that *discourage* the sending of large quantities of instant messaging spam (a.k.a. "spim") or, in general, abusive traffic. Abusive stanzas could be generated by XMPP clients connected to legitimate servers or by XMPP servers with virtual clients, where the malicious entities are hosted on networks of "zombie" machines. Such abusive stanzas could take many forms; a full taxonomy is outside the scope of this document.
One technique developed to combat abusive messages and behavior via non-XMPP technologies requires humans to be differentiated from bots using a "Completely Automated Public Turing Test to Tell Computers and Humans Apart" or CAPTCHA (see <http://www.captcha.net/>). These challenge techniques are easily adapted to discourage XMPP abuse. The very occasional inconvenience of responding to a CAPTCHA (e.g., when creating an IM account or sending a message to a new correspondent) is small and perfectly acceptable -- especially when compared to the countless robot-generated interruptions people might otherwise have to filter every day.
An alternative technique to CAPTCHAs requires Desktop PC clients to undertake a Hashcash [3] challenge. These are completely transparent to PC users. They require clients to perform specified CPU-intensive work, making it difficult to send large amounts of abusive traffic.
Both CAPTCHAs and hashcash have been criticized regarding their effectiveness (or lack thereof). Therefore, the challenge protocol specified herein provides a great deal of flexibility, so that challenges can include CAPTCHAs, hashcash, word puzzles, so-called kitten authentication, and any other mechanism that may be developed in the future.
The generic challenge protocol described in this document is designed for incorporation into protocols such as In-Band Registration (XEP-0077) [4], Multi-User Chat (XEP-0045) [5], Privacy Lists (XEP-0016) [6], and SPIM-Blocking Control (XEP-0159) [7].

2 Requirements

2.1 Extensibility

The CAPTCHAs in most common use today are Optical Character Recognition (OCR) challenges where an image containing deformed text is presented and the human enters the characters they can read. However, if OCR software advances more rapidly than the techniques used to disguise text from Artificial Intelligence (AI) then very different CAPTCHAs will need to be deployed. This protocol must be extensible enough to allow the incorporation of CAPTCHA

[1] RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc6120>.
[2] RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence <http://tools.ietf.org/html/rfc6121>.
[3] Hashcash <http://hashcash.org/>.
[4] XEP-0077: In-Band Registration <https://xmpp.org/extensions/xep-0077.html>.
[5] XEP-0045: Multi-User Chat <https://xmpp.org/extensions/xep-0045.html>.
[6] XEP-0016: Privacy Lists <https://xmpp.org/extensions/xep-0016.html>.
[7] XEP-0159: SPIM-Blocking Control <https://xmpp.org/extensions/xep-0159.html>.

techniques that may not have been envisaged.

2.2 Variety

Several common CAPTCHA techniques present major problems to users with disabilities (see Inaccessibility of Visually-Oriented Anti-Robot Tests [8]). Clients running in constrained environments may not be able to perform some challenges (e.g., due to the absence of audio output or a lack of CPU performance). This protocol must allow clients to be offered a choice from a variety of challenges.

3 Protocol

3.1 Simple Challenge

3.1.1 Triggering Stanza

A "triggering stanza" is an XMPP <message/>, <presence/>, or <iq/> stanza that is deemed abusive by the receiving entity (e.g., a client) or an intermediate router (e.g., a server). The entity that generates a triggering stanza is called a "sender".

Listing 1: Sender Generates Triggering Stanza

```
<message from='robot@abuser.com/zombie'
    to='innocent@victim.com'
    xml:lang='en'
    id='spam1'>
  <body>Love pills - 75% OFF</body>
  <x xmlns='jabber:x:oob'>
    <url>http://www.abuser.com/lovepills.html</url>
  </x>
</message>
```

3.1.2 Challenge Stanza

Upon receiving a triggering stanza, an entity MAY send a "challenge stanza". An entity MUST NOT send a challenge stanza under any other circumstances. The entity that generates the challenge stanza is called the "challenger".
The challenge stanza consists of an XMPP <message/> stanza containing a data form for the sender to fill out, formatted according to Data Forms (XEP-0004) [9], optionally along with a <body/> and other elements. The following rules apply to the challenge stanza.

[8] Inaccessibility of Visually-Oriented Anti-Robot Tests <http://www.w3.org/TR/turingtest/>.
[9] XEP-0004: Data Forms <https://xmpp.org/extensions/xep-0004.html>.

1. The challenge stanza MUST include an 'id' attribute set to the challenge ID (i.e., a unique identifier for this challenge within the challenger's application).

2. The challenge stanza SHOULD include a <body/> element that provides an explanation of the challenge for clients that do not yet support CAPTCHA forms.

3. The challenge stanza MAY include a URL (typically a Web page with instructions) using Out-of-Band Data (XEP-0066) [10] as an alternative for clients that do not yet support CAPTCHA forms.

4. The 'xml:lang' attribute of the challenge stanza SHOULD be the same as the one received from the sender, if any.

5. The challenge stanza MUST include a CAPTCHA form, i.e., a data form of type "form" containing one or more challenges. [11]

6. The CAPTCHA form MUST include a hidden field named "FORM_TYPE" (in accordance with Field Standardization for Data Forms (XEP-0068) [12]) whose value MUST be "urn:xmpp:captcha".

7. The CAPTCHA form MUST include a hidden field named "challenge" set to the challenge ID.

8. The CAPTCHA form MUST include a hidden field named "from" set to the value of the 'to' attribute from the triggering stanza.

9. If the triggering stanza included an 'id' attribute, then the CAPTCHA form MUST include a hidden field named "sid" set to that value.

10. Each of the CAPTCHA form's non-hidden <field/> elements MAY contain a different challenge.

11. Each CAPTCHA field MAY contain a media element (see Data Forms Media Element (XEP-0221) [13]) that in turn contains a pointer to media that the sender shall use in solving puzzles, performing optical character recognition, identifying audio or video samples, etc. When the sender replies to a media element via a data form of type "submit", the field type SHOULD be "text-single" (which is the default for data form fields) but MAY in turn include a media element if acceptable to the challenger application.

Listing 2: Challenger Offers a Choice of Challenges to Sender

```
<message from='victim.com'
```

[10] XEP-0066: Out of Band Data <https://xmpp.org/extensions/xep-0066.html>.

[11] Inclusion of a CAPTCHA form not only makes it possible to flexibly support or require a large number of challenge types, but also enables constrained clients to respond to challenges (e.g., mobile phone clients that cannot present web pages, or clients on XMPP-only networks).

[12] XEP-0068: Field Data Standardization for Data Forms <https://xmpp.org/extensions/xep-0068.html>.

[13] XEP-0221: Data Forms Media Element <https://xmpp.org/extensions/xep-0221.html>.

```
  to='robot@abuser.com/zombie'
  xml:lang='en'
  id='F3A6292C'>
<body>
  Your messages to innocent@victim.com are being blocked. To unblock
  them, visit http://www.victim.com/challenge.html?F3A6292C
</body>
<x xmlns='jabber:x:oob'>
  <url>http://www.victim.com/challenge.html?F3A6292C</url>
</x>
<captcha xmlns='urn:xmpp:captcha'>
  <x xmlns='jabber:x:data' type='form'>
    <field type='hidden' var='FORM_TYPE'>
      <value>urn:xmpp:captcha</value>
    </field>
    <field type='hidden' var='from'><value>innocent@victim.com</
       value></field>
    <field type='hidden' var='challenge'><value>F3A6292C</value></
       field>
    <field type='hidden' var='sid'><value>spam1</value></field>
    <field var='ocr' label='Enter␣the␣text␣you␣see'>
      <media xmlns='urn:xmpp:media-element'
             height='80'
             width='290'>
        <uri type='image/jpeg'>
          http://www.victim.com/challenges/ocr.jpeg?F3A6292C
        </uri>
        <uri type='image/jpeg'>
          cid:sha1+f24030b8d91d233bac14777be5ab531ca3b9f102@bob.xmpp
             .org
        </uri>
      </media>
    </field>
    <field var='picture_recog' label='Identify␣the␣picture'>
      <media xmlns='urn:xmpp:media-element'
             height='150'
             width='150'>
        <uri type='image/jpeg'>
          http://www.victim.com/challenges/picture.jpeg?F3A6292C
        </uri>
        <uri type='image/jpeg'>
          cid:sha1+f2377f3a3287eac81028243079e1aa9905c466bc@bob.xmpp
             .org
        </uri>
      </media>
    </field>
    <field var='speech_recog' label='Enter␣the␣words␣you␣hear'>
      <media xmlns='urn:xmpp:media-element'>
        <uri type='audio/x-wav'>
```

```
            http://www.victim.com/challenges/speech.wav?F3A6292C
          </uri>
          <uri type='audio/ogg-speex'>
            http://www.victim.com/challenges/speech.ogg?F3A6292C
          </uri>
        </media>
      </field>
      <field var='video_recog' label='Identity␣the␣video'>
        <media xmlns='urn:xmpp:media-element'
               height='150'
               width='150'>
          <uri type='video/mpeg'>
            http://www.victim.com/challenges/video.mpeg?F3A6292C
          </uri>
        </media>
      </field>
      <field label='Type␣the␣color␣of␣a␣stop␣light' type='text-single'
          var='qa'/>
      <field label='93C7A' type='text-single' var='SHA-256'/>
    </x>
  </captcha>
</message>
```

The sender then would retrieve the media data via HTTP or (for the cid: URIs) via XMPP as described in Bits of Binary (XEP-0231) [14].

3.1.3 Response Stanza

The sender's client SHOULD ignore the challenge stanza in either of the following cases:

- If it has not recently sent (e.g., in the last two minutes) a stanza to the JID specified in the 'from' field of the form with the 'id' specified in the 'sid' field (or with no 'id' if no 'sid' field is included). [15]
- If the 'from' attribute of the challenge stanza does not match the 'from' field of the form. (If the values are different, then they still match if the bare JIDs are the same, or if the 'from' attribute is the domain of the other JID.)

Otherwise, if the challenger provided a URL using Out-of-Band Data, then the sender's client MAY present the URL to the sender, instead of responding to the CAPTCHA form, in any of the following cases:

[14] XEP-0231: Bits of Binary <https://xmpp.org/extensions/xep-0231.html>.

[15] Otherwise the user's presence would be disclosed, or a robot might dupe the user into providing answers to other people's challenges!

- if it does not understand the CAPTCHA form
- if it does not support all of the *required* challenges (see Multiple Challenges)
- if it does not support enough of the challenges (see Multiple Challenges)

Otherwise, the sender's client MUST respond to the challenge.
The sender's client MUST respond with a <not-acceptable/> error in any of the following cases:

- if it does not support all of the required challenges (see Multiple Challenges)
- if it does not support enough of the challenges (see Multiple Challenges)
- if the sender declines the challenge

Listing 3: Sender Reports Challenge Not Acceptable

```
<message type='error'
    from='robot@abuser.com/zombie'
    to='victim.com'
    xml:lang='en'
    id='F3A6292C'>
  <error type='modify'>
    <not-acceptable xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</message>
```

Otherwise, it MUST select one challenge according to the sender's preferences and submit the sender's response form to the challenger.

Listing 4: Sender Sends One Response to Challenger

```
<iq type='set'
    from='robot@abuser.com/zombie'
    to='victim.com'
    xml:lang='en'
    id='z140r0s'>
  <captcha xmlns='urn:xmpp:captcha'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:captcha</value>
      </field>
      <field var='from'><value>innocent@victim.com</value></field>
      <field var='challenge'><value>F3A6292C</value></field>
      <field var='sid'><value>spam1</value></field>
      <field var='ocr'><value>7nHL3</value></field>
    </x>
  </captcha>
</iq>
```

3.1.4 Result Stanza

The challenger SHOULD send a <service-unavailable/> error to the sender if:

- The challenger did not send the specified challenge. [16]
- The sender already submitted its response to this challenge.
- The sender took too long to submit its response.

Note: This error MAY be sent even in cases where the challenge became unnecessary while the challenger was waiting for the response.

Listing 5: Challenger Indicates Challenge Not Found

```
<iq type='error'
    from='victim.com'
    to='robot@abuser.com/zombie'
    id='z140r0s'>
  <error type='cancel'>
    <service-unavailable xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

After receiving a correct response to its challenge, the challenger SHOULD inform the sender that it was successful.

Listing 6: Challenger Tells Sender it Passed

```
<iq type='result'
    from='victim.com'
    to='robot@abuser.com/zombie'
    id='z140r0s'/>
```

However, if the sender submits an incorrect response the challenger SHOULD send it a <not-acceptable/> error with type "cancel": [17]

Listing 7: Challenger Tells Sender it Failed

```
<iq type='error'
    from='victim.com'
```

[16] If the challenger is a client then it SHOULD be careful not to leak information about the presence of the sender and reply to potentially bogus challenge responses with exactly the same XML that its server would send if the sender were offline.

[17] If a large proportion of the responses a server is receiving from another IP are incorrect then it SHOULD inform the administrator of the other server using the protocol specified in Abuse Reporting (XEP-0161) [18] or Abuse Reporting (XEP-0236) [19]. It SHOULD also automatically block all stanzas from the abusive user, users, server or IP.

```
    to='robot@abuser.com/zombie'
    id='z140r0s'>
  <error type='cancel'>
    <not-acceptable xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
  </error>
</iq>
```

3.2 Multiple Challenges

The challenger MAY demand responses to more than one of the challenges it is offering; this is done by including an 'answers' <field/> element in the form, which specifies how many answers the sender needs to include. The challenger also MAY require responses to particular challenges; this is done by including a <required/> element in the compulsory fields.

Listing 8: Challenger Sets Multiple Challenges

```
<message from='victim.com'
    to='robot@abuser.com/zombie'
    xml:lang='en'
    id='73DE28A2'>
  <body>Your messages to innocent@victim.com are being blocked.
    To unblock them, ask innocent@victim.com to send you a message.</
       body>
  <captcha xmlns='urn:xmpp:captcha'>
    <x xmlns='jabber:x:data' type='form'>
      <field type='hidden' var='FORM_TYPE'>
        <value>urn:xmpp:captcha</value>
      </field>
      <field type='hidden' var='from'><value>innocent@victim.com</
         value></field>
      <field type='hidden' var='challenge'><value>73DE28A2</value></
         field>
      <field type='hidden' var='sid'><value>spam2</value></field>
      <field type='hidden' var='answers'><value>2</value></field>
      <field var='ocr' label='Enter␣the␣text␣you␣see'>
        <media xmlns='urn:xmpp:media-element'
               height='80'
               width='290'>
          <uri type='image/jpeg'>
            http://www.victim.com/challenges/ocr.jpeg?F3A6292C
          </uri>
          <uri type='image/jpeg'>
            cid:sha1+f24030b8d91d233bac14777be5ab531ca3b9f102@bob.xmpp
               .org
          </uri>
        </media>
      </field>
      <field var='audio_recog' label='Describe␣the␣sound␣you␣hear'>
```

```
        <media xmlns='urn:xmpp:media-element'>
          <uri type='audio/x-wav'>
            http://www.victim.com/challenges/audio.wav?F3A6292C
          </uri>
        </media>
      </field>
      <field label='Type␣the␣color␣of␣a␣stop␣light' type='text-single'
          var='qa'>
        <required/>
      </field>
      <field label='e03d7' type='text-single' var='SHA-256'/>
    </x>
  </captcha>
</message>
```

If the sender finds the request acceptable, it MUST answer all challenges that include a <required/> element. If the total number of answers was specified and it is greater than the number of <required/> elements then the sender MUST also answer one or more of the challenges without a <required/> element. In the example above, the sender should respond to the 'qa' challenge *and* one of the other challenges ('ocr', 'audio_recog' or 'SHA-256').

Listing 9: Sender Sends Multiple Responses to the Challenger

```
<iq type='set'
    from='robot@abuser.com/zombie'
    to='victim.com'
    xml:lang='en'
    id='73DE28A2'>
  <captcha xmlns='urn:xmpp:captcha'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>urn:xmpp:captcha</value>
      </field>
      <field var='from'><value>innocent@victim.com</value></field>
      <field var='challenge'><value>73DE28A2</value></field>
      <field var='sid'><value>spam2</value></field>
      <field var='answers'><value>2</value></field>
      <field var='qa'><value>red</value></field>
      <field var='SHA-256'><value>innocent@victim.com2450F06C173B05E3<
          /value></field>
    </x>
  </captcha>
</iq>
```

The challenger MAY decide the sender has passed a challenge even if the responses are not all perfectly correct.

4 Extended In-Band Registration

This section shows how challenges SHOULD be combined with the existing In-Band Registration protocol according to the rules defined in the Extensibility section of In-Band Registration (XEP-0077) [20].
Note: The <challenge/> wrapper element is not included, because In-Band Registration (XEP-0077) [21] specifies that data forms shall be contained as the direct children of the <query/> element.

Listing 10: Entity Requests Registration Fields from Host

```
<iq type='get' xml:lang='en' id='reg1'>
  <query xmlns='jabber:iq:register'/>
</iq>
```

Note that the CAPTCHA form MUST be inside the <query/> element, and the server's challenge ID is specified within the form:

Listing 11: Host Returns Registration and Challenge Fields to Entity

```
<iq type='result' xml:lang='en' id='reg1'>
  <query xmlns='jabber:iq:register'>
    <x xmlns='jabber:x:data' type='form'>
      <field type='hidden' var='FORM_TYPE'>
        <value>jabber:iq:register</value>
      </field>
      <field type='hidden' var='challenge'><value>F3A6292C</value></
         field>
      <field type='hidden' var='sid'><value>reg1</value></field>
      <field type='hidden' var='answers'><value>3</value></field>
      <field var='ocr' label='Enter␣the␣text␣you␣see'>
        <media xmlns='urn:xmpp:media-element'
               height='80'
               width='290'>
          <uri type='image/jpeg'>
            http://www.victim.com/challenges/ocr.jpeg?F3A6292C
          </uri>
        </media>
      </field>
      <field label='93C7A' type='text-single' var='SHA-256'/>
      <field type='text-single' var='username'>
        <required/>
      </field>
      <field type='text-private' var='password'>
        <required/>
      </field>
```

[20] XEP-0077: In-Band Registration <https://xmpp.org/extensions/xep-0077.html>.
[21] XEP-0077: In-Band Registration <https://xmpp.org/extensions/xep-0077.html>.

```
    </x>
    <instructions>
      To register, visit http://www.victim.com/register.html
    </instructions>
    <x xmlns='jabber:x:oob'>
      <url>http://www.victim.com/register.html</url>
    </x>
  </query>
</iq>
```

The server MAY include an <instructions/> element and a URL using Out-of-Band Data (e.g., a web page) in the <query/> element (see example above). In-Band Registration recommends that the challenger SHOULD submit the completed x:data form, however if it does not understand the form, then it MAY present the instructions and the included URL to the user instead of providing the required information in-band.

Listing 12: Entity Provides Required Information In-Band

```
<iq type='set' xml:lang='en' id='reg2'>
  <query xmlns='jabber:iq:register'>
    <x xmlns='jabber:x:data' type='submit'>
      <field var='FORM_TYPE'>
        <value>jabber:iq:register</value>
      </field>
      <field var='challenge'><value>F3A6292C</value></field>
      <field var='sid'><value>reg1</value></field>
      <field var='answers'><value>3</value></field>
      <field var='ocr'><value>7nHL3</value></field>
      <field var='username'><value>bill</value></field>
      <field var='password'><value>Calliope</value></field>
    </x>
  </query>
</iq>
```

5 Multi-User Chat

A service that hosts multi-user chat rooms in accordance with Multi-User Chat (XEP-0045) [22] MAY challenge unknown entities that seek to join such rooms or that send messages in such rooms.

Listing 13: Sender Attempts to Join Chat Room

```
<presence from='robot@abuser.com/zombie'
          to='friendly-chat@muc.victim.com/robot101'>
  <x xmlns='http://jabber.org/protocol/muc'/>
```

[22] XEP-0045: Multi-User Chat <https://xmpp.org/extensions/xep-0045.html>.

```
</presence>
```

Listing 14: Challenger Offers a Choice of Challenges to Sender

```
<message from='friendly-chat@muc.victim.com'
         to='robot@abuser.com/zombie'
         id='A4C7303D'>
  <body>
    Your messages to friendly-chat@muc.victim.com are being blocked.
       To unblock
    them, visit http://www.victim.com/challenge.html?A4C7303D
  </body>
  <x xmlns='jabber:x:oob'>
    <url>http://www.victim.com/challenge.html?A4C7303D</url>
  </x>
  <captcha xmlns='urn:xmpp:captcha'>
    <x xmlns='jabber:x:data' type='form'>
      <field type='hidden' var='FORM_TYPE'>
        <value>urn:xmpp:captcha</value>
      </field>
      <field type='hidden' var='from'><value>friendly-chat@muc.victim.
         com</value></field>
      <field type='hidden' var='challenge'><value>A4C7303D</value></
         field>
      <field var='ocr' label='Enter␣the␣text␣you␣see'>
        <media xmlns='urn:xmpp:media-element'
               height='80'
               width='290'>
          <uri type='image/jpeg'>
            http://www.victim.com/challenges/ocr.jpeg?A4C7303D
          </uri>
        </media>
      </field>
      <field var='picture_recog' label='Identify␣the␣picture'>
        <media xmlns='urn:xmpp:media-element'
               height='150'
               width='150'>
          <uri type='image/jpeg'>
            http://www.victim.com/challenges/picture.jpeg?A4C7303D
          </uri>
          <uri type='image/jpeg'>
            cid:sha1+f2377f3a3287eac81028243079e1aa9905c466bc@bob.xmpp
               .org
          </uri>
        </media>
      </field>
      <field var='speech_recog' label='Enter␣the␣words␣you␣hear'>
        <media xmlns='urn:xmpp:media-element'>
          <uri type='audio/x-wav'>
            http://www.victim.com/challenges/speech.wav?A4C7303D
```

```
          </uri>
          <uri type='audio/ogg-speex'>
            http://www.victim.com/challenges/speech.ogg?A4C7303D
          </uri>
        </media>
      </field>
      <field var='video_recog' label='Identity␣the␣video'>
        <media xmlns='urn:xmpp:media-element'
               height='150'
               width='150'>
          <uri type='video/mpeg'>
            http://www.victim.com/challenges/video.mpeg?A4C7303D
          </uri>
        </media>
      </field>
      <field label='Type␣the␣color␣of␣a␣stop␣light' type='text-single'
          var='qa'/>
      <field label='93C7A' type='text-single' var='SHA-256'/>
    </x>
  </captcha>
</message>
```

6 Challenge Types

6.1 Introduction

Entities MUST address the needs of disabled people and CPU-constrained clients by offering senders a reasonable choice of different types of challenges.
Desktop clients running on modern PCs will typically be configured to automatically perform a specified 'SHA-256' Hashcash challenge (see below) whenever it is below a certain level of difficulty, with the result that many people may not even notice challenges most of the time. However, people using CPU-constrained clients (e.g. Web or mobile clients) would notice the performance hit. They might prefer to take a CAPTCHA challenge instead. [23]
Visually disabled people using a CPU-constrained client could configure their client to always present them with an audio CAPTCHA challenge.
Most of the challenges below are language sensitive. However, the evaluation of the OCR and Hashcash responses does not depend on the language the sender is using.
Challenge types are distinguished by the 'var' attribute of each <field/> element. Several types of challenges are described below. More challenges MAY be documented elsewhere and registered with the XMPP Registrar (see Field Standardization).

[23] A CPU-constrained client could ask a faster computer (e.g., its server) to perform a Hashcash challenge for it.

6.2 SHA-256 Hashcash

The SHA-256 Hashcash challenge is transparent to average PC users. It is indicated when the value of the 'var' attribute is 'SHA-256'. It forces clients to perform CPU-intensive work, making it difficult to send large amounts of abusive traffic. This significantly reduces abusive traffic, but alone it will not completely stop abusive stanzas from being sent through large collections of 'zombie' computers. [24]
The challenger MUST set the 'label' attribute of the <field/> element to a hexadecimal random number containing a configured number of bits (e.g., $2^{20} \leq$ label $< 2^{21}$).
To pass the test, the sender MUST return a text string that starts with the JID the sender sent the first stanza to (i.e., the stanza that triggered the challenge). The least significant bits of the SHA-256 hash (see SHA [25]) of the string MUST equal the hexadecimal value specified by the challenger (in the 'label' attribute of the <field/> element). For example, if the 'label' attribute is the 20-bit value 'e03d7' then the following string would be correct:

```
innocent@victim.com2450F06C173B05E3
```

Note: When configuring the number of bits to be specified by a challenger in the 'label' attribute values, administrators MUST balance the need to make mass abuse as difficult as possible, with the inconvenience that may be caused to the users of less powerful computers. (Most clients will be challenged only very occasionally, so the consumption of 70% of a typical desktop CPU for 4 seconds might be considered appropriate.) Administrators SHOULD increment the configured number of bits from time to time to match increases in the performance of typical desktop PCs. If an administrator notices that abusive robots never attempt the Hashcash challenge, then he SHOULD consider reducing the number of bits, to avoid inconveniencing people unnecessarily.

6.3 CAPTCHAs

Note: It may be profitable to send abusive stanzas even if less than one percent of CAPTCHA responses are successful. The effectiveness of a CAPTCHA challenge needs to be close to perfect, unless it is used in combination with other anti-abuse techniques.
If a media type is specified (see table below) then the <field/> element MUST contain a <media/> element that includes a <uri/> element of that type. Clients that support the CAPTCHA type MUST be able to play or render the specified MIME-types (see table below). They MAY also support other formats. [26]

[24] The hope is that the extra CPU usage will often be noticed by the owners of the zombie machines, who will be more likely to fix them.
[25] Secure Hash Standard: Federal Information Processing Standards Publication 180-2 <http://csrc.nist.gov/publications/fips/fips180-2/fips186-2withchangenotice.pdf>.
[26] Audio CAPTCHAs typically require challengers to provide at least the 'audio/x-wav' MIME-type (with the PCM codec) because more efficient patent-free formats are often not supported by constrained clients. It is RECOMMENDED that challengers provide more compact formats (like Ogg Speex or MP3) too.

The 'type' attribute of the <field/> element SHOULD be 'text-single', 'text-private', or 'text-multi' (if no 'type' is specified, the default is 'text-single'). [27] The response MUST be provided in the language specified by the 'xml:lang' attribute of the challenge stanza.

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

* See the Internationalization Considerations section of this document.
* The image portrays random characters that humans can read but OCR software cannot. [28] To pass the challenge, the sender must simply type the characters. The correct answer SHOULD NOT depend on the language specified by the 'xml:lang' attribute of the challenge stanza.

7 Question and Answer for Legacy Clients

A challenger MAY provide a text question in the <body/> element of a challenge stanza for clients that do not support CAPTCHA forms. Entities that cannot serve Out-of-Band Data URLs

[27] The 'boolean' and 'list-single' field types would make it trivial for a robot to provide a correct response at least some of the time.
[28] See PWNtcha <http://sam.zoy.org/pwntcha/> for some example OCR CAPTCHA images.

MAY use this option to challenge legacy clients.
Note: Robots always attempt the easiest challenge they are offered. So the question MUST be at least as difficult for a robot as the CAPTCHA form.
Note: Even if it provides a text question in the <body/> element, a challenger MUST always provide a CAPTCHA form.

Listing 15: Challenger Includes a Legacy Challenge

```
<message from='innocent@victim.com/pda'
    to='robot@abuser.com/zombie'
    xml:lang='en'
    id='F3A6292C'>
  <body>Your messages to me are being blocked. To unblock them,
    reply with the color of a stop light followed by 'F3A6292C'.</body
      >
  <captcha xmlns='urn:xmpp:captcha'>
    <x xmlns='jabber:x:data' type='form'>
      <field type='hidden' var='FORM_TYPE'>
        <value>urn:xmpp:captcha</value>
      </field>
      <field type='hidden' var='from'><value>innocent@victim.com</
         value></field>
      <field type='hidden' var='challenge'><value>F3A6292C</value></
         field>
      <field type='hidden' var='sid'><value>spam1</value></field>
      <field label='Type␣the␣color␣of␣a␣stop␣light' type='text-single'
          var='qa'/>
      <field label='93C7A' type='text-single' var='SHA-256'/>
    </x>
  </captcha>
</message>
```

Legacy clients respond to the challenger using a <message/> stanza (not an <iq/>).

Listing 16: Legacy Sender Responds

```
<message from='robot@abuser.com/zombie' to='innocent@victim.com/pda'>
  <body>red F3A6292C</body>
</message>
```

The challenger SHOULD treat the stanza as a normal message (instead of as a response to its challenge) if the legacy client either takes too long to submit it or has already responded to the challenge. The challenger MAY treat the response as a normal message even in cases where the challenge became unnecessary while the challenger was waiting for the response. Otherwise the challenger MUST report the result of the challenge to the legacy client using a <message/> stanza (not an <iq/>).

Listing 17: Challenger Tells Legacy Sender it Passed

```
<message from='innocent@victim.com/pda' to='robot@abuser.com/zombie'>
  <body>Your message was delivered. Your messages
        to me are no longer being blocked.</body>
</message>
```

Listing 18: Challenger Tells Legacy Sender it Failed

```
<message type='error'
         from='innocent@victim.com/pda'
         to='robot@abuser.com/zombie'>
  <error type='cancel'>
    <not-acceptable xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'/>
    <text xmlns='urn:ietf:params:xml:ns:xmpp-stanzas'>
      Your message to me was not delivered.
    </text>
  </error>
</message>
```

8 Discontinuation Policy

It is RECOMMENDED that entities employ other techniques to combat abusive stanzas in addition to those described in this document (e.g., see Abuse Reporting (XEP-0161) [29] and Best Practices to Discourage Denial of Service Attacks (XEP-0205) [30]).
It is expected that this protocol will be an important and successful tool for discouraging abusive traffic. However, much of its success is dependent on the quality of the CAPTCHAs and other puzzles employed by a particular implementation.
The administrator of an application that functions as a challenger SHOULD discontinue the use of CAPTCHA forms under the following circumstances:

- If he realises that the challenger's challenges are largely ineffective in combating abusive traffic, and that the reduction in abuse does not compensate for the inconvenience to humans of responding to the challenger's challenges.
- If other, *more transparent*, techniques being employed by the challenger are so successful that challenges are offering only negligible additional protection against abusive traffic.
- If the challenger needs no protection at all because it receives only a negligible amount of abusive traffic.

[29] XEP-0161: Abuse Reporting <https://xmpp.org/extensions/xep-0161.html>.
[30] XEP-0205: Best Practices to Discourage Denial of Service Attacks <https://xmpp.org/extensions/xep-0205.html>.

9 Internationalization Considerations

Each form field SHOULD include a 'label' attribute. If the sender did not include an 'xml:lang' attribute, then the challenger may not know the correct language for the labels. Therefore, depending on user preferences, the client that receives a challenge MAY present generic but localized text instead of label text that would not be understood by the user. Suggested generic text (to be suitably localized) is provided by Table 1 in the CAPTCHAs section of this document.

10 Security Considerations

The use of CAPTCHAs is not a panacea, and should be combined with other anti-abuse mechanisms, such as those described in Abuse Reporting (XEP-0161) [31] and XEP-0205. For example, the task of finding solutions to CAPTCHAs and other computational puzzles is becoming easier for computer programs, and in any case can be farmed out to third parties. Therefore challengers should limit the number of triggering stanzas (e.g., registration attempts, subscription requests, or chatroom joins) allowed per JabberID or IP address during any given time period, and may simply refuse repeated stanzas by terminating an XML stream with a <policy-violation/> stream error or returning a <not-acceptable/> stanza error as appropriate. In addition, a challenger should feel free to deploy additional anti-abuse mechanisms as needed.

11 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [32].

12 XMPP Registrar Considerations

12.1 Protocol Namespaces

The XMPP Registrar [33] includes "urn:xmpp:captcha" in its registry of protocol namespaces (see <https://xmpp.org/registrar/namespaces.html>).

[31] XEP-0161: Abuse Reporting <https://xmpp.org/extensions/xep-0161.html>.

[32] The Internet Assigned Numbers Authority (IANA) is the central coordinator for the assignment of unique parameter values for Internet protocols, such as port numbers and URI schemes. For further information, see <http://www.iana.org/>.

[33] The XMPP Registrar maintains a list of reserved protocol namespaces as well as registries of parameters used in the context of XMPP extension protocols approved by the XMPP Standards Foundation. For further information, see <https://xmpp.org/registrar/>.

12.2 Field Standardization

12.2.1 CAPTCHA FORM_TYPE

The XMPP Registrar registers following FORM_TYPE. Additional fields might be defined in future submissions.

```
<form_type>
  <name>urn:xmpp:captcha</name>
  <doc>XEP-0158</doc>
  <desc>Forms enabling the use of CAPTCHAs.</desc>
  <field
      var='answers'
      type='hidden'
      label='number␣of␣answers␣required'/>
  <field
      var='audio_recog'
      type='text-single'
      label='text␣associated␣with␣a␣sound'/>
  <field
      var='challenge'
      type='hidden'
      label='challenge␣ID'/>
  <field
      var='from'
      type='hidden'
      label='to␣attribute␣of␣stanza␣that␣triggered␣challenge'/>
  <field
      var='ocr'
      type='text-single'
      label='code␣appearing␣in␣an␣image'/>
  <field
      var='picture_q'
      type='text-single'
      label='answer␣associated␣with␣a␣picture'/>
  <field
      var='picture_recog'
      type='text-single'
      label='text␣associated␣with␣a␣picture'/>
  <field
      var='qa'
      type='text-single'
      label='answer␣to␣a␣question'/>
  <field
      var='SHA-256'
      type='text-single'
      label='least␣significant␣bits␣of␣SHA-256␣hash␣of␣text␣should␣
         equal␣hexadecimal␣label'/>
  <field
```

```
      var='sid'
      type='hidden'
      label='stanza␣ID'/>
  <field
      var='speech_q'
      type='text-single'
      label='answer␣associated␣with␣speech'/>
  <field
      var='speech_recog'
      type='text-single'
      label='text␣associated␣with␣speech'/>
  <field
      var='video_q'
      type='text-single'
      label='answer␣associated␣with␣a␣video'/>
  <field
      var='video_recog'
      type='text-single'
      label='text␣associated␣with␣a␣video'/>
</form_type>
```

12.2.2 jabber:iq:register FORM_TYPE

The XMPP Registrar registers the following fields for the existing jabber:iq:register FORM_-TYPE. Additional fields might be defined in future submissions.

```
<form_type>
  <name>jabber:iq:register</name>
  <doc>XEP-0077</doc>
  <field
      var='answers'
      type='hidden'
      label='number␣of␣answers␣required'/>
  <field
      var='audio_recog'
      type='text-single'
      label='text␣associated␣with␣a␣sound'/>
  <field
      var='challenge'
      type='hidden'
      label='challenge␣ID'/>
  <field
      var='ocr'
      type='text-single'
      label='code␣appearing␣in␣an␣image'/>
  <field
      var='picture_q'
      type='text-single'
```

```
      label='answer associated with a picture'/>
  <field
      var='picture_recog'
      type='text-single'
      label='text associated with a picture'/>
  <field
      var='qa'
      type='text-single'
      label='answer to a question'/>
  <field
      var='SHA-256'
      type='text-single'
      label='least significant bits of SHA-256 hash of text should
         equal hexadecimal label'/>
  <field
      var='sid'
      type='hidden'
      label='stanza ID'/>
  <field
      var='speech_q'
      type='text-single'
      label='answer associated with speech'/>
  <field
      var='speech_recog'
      type='text-single'
      label='text associated with speech'/>
  <field
      var='video_q'
      type='text-single'
      label='answer associated with a video'/>
  <field
      var='video_recog'
      type='text-single'
      label='text associated with a video'/>
</form_type>
```

13 XML Schema

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

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='urn:xmpp:captcha'
    xmlns='urn:xmpp:captcha'
    elementFormDefault='qualified'>

  <xs:annotation>
```

```
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-0158: http://xmpp.org/extensions/xep-0158.html
    </xs:documentation>
  </xs:annotation>

  <xs:import namespace='jabber:x:data'
             schemaLocation='http://xmpp.org/schemas/x-data.xsd'/>

  <xs:element name='captcha'>
    <xs:complexType>
      <xs:sequence xmlns:xdata='jabber:x:data'>
        <xs:element ref='xdata:x' minOccurs='1' maxOccurs='1'/>
      </xs:sequence>
    </xs:complexType>
  </xs:element>

</xs:schema>
```

14 Open Issues

Another protocol could allow users to edit the challenges their server will make on their behalf. For example, the number of SHA-256 bits, a personal or original question and answer, a picture, a video, or a sound recording. Of course Aunt Tillie would typically use this feature only if she was plagued by abusive traffic.