Abstract

This document defines a new primitive for the "Sieve" language that tests for the occurrence of one or more strings in the body of an e-mail message.

1. Introduction

The proposed "body" test checks for the occurrence of one or more strings in the body of an e-mail message. Such a test was initially discussed for the [SIEVE] base document, but was subsequently removed because it was thought to be too costly to implement.

Nevertheless, several server vendors have implemented some form of the "body" test.

This document reintroduces the "body" test as an extension, and specifies it syntax and semantics.

2. Conventions used.
Conventions for notations are as in [SIEVE] section 1.1, including use of [KEYWORDS] and "Syntax:" label for the definition of action and tagged arguments syntax.

The capability string associated with extension defined in this document is "body".

3. Test body

Syntax:

"body" [COMPARATOR] [MATCH-TYPE] [BODY-TRANSFORM]
<key-list: string-list>

The body test matches text in the body of an e-mail message, that is, anything following the first empty line after the header. (The empty line itself, if present, is not considered to be part of the body.)

The COMPARATOR and MATCH-TYPE keyword parameters are defined in [SIEVE]. The BODY-TRANSFORM is a keyword parameter discussed in section 4, below.

If a message consists of a header only, not followed by an empty line, all "body" tests fail, including that for an empty string.

If a message consists of a header followed only by an empty line with no body lines following it, the message is considered to have an empty string as a body.

4. Body Transform

Prior to matching text in a message body, "transformations" can be applied that filter and decode certain parts of the body. These transformations are selected by a "BODY-TRANSFORM" keyword parameter.

Syntax:

":raw"
/ ":content" <content-types: string-list>
/ ":text"

The default transformation is :text.

4.1 Body Transform ":raw"

The ":raw" transform is intended to match against the undecoded body of a message.

If the specified body-transform is ":raw", the [MIME] structure of the body is irrelevant. The implementation MUST NOT remove any transfer encoding from the message, MUST NOT refuse to filter messages with syntactic errors (unless the environment it is part of rejects them outright), and MUST NOT interpret or skip MIME headers of enclosed body parts.

Example:

require "body";
# This will match a message containing the words "MAKE MONEY FAST"
# in body or MIME headers other than the outermost RFC 822 header,
# but will not match a message containing the words in a
# content-transfer-encoded body.

```plaintext
if body :raw :contains "MAKE MONEY FAST" {
    reject;
}
```

## 4.2 Body Transform ":content"

If the body transform is ":content", only MIME parts that have
the specified content-types are selected for matching.

If an individual content type contains a ‘/’ (slash), it
specifies a full <type>/<subtype> pair, and matches only
that specific content type. If it is the empty string, all
MIME content types are matched. Otherwise, it specifies a
<type> only, and any subtype of that type matches it.

The search for MIME parts matching the :content specification is
recursive and automatically descends into multipart and
message/rfc822 MIME parts. Once a MIME part has been identified
as suitable for searching, only its direct contents are searched
for the key strings.

For example, a document with "multipart" major content type only
directly contains the text in its epilogue and prologue section;
all the user-visible data inside it is directly contained in
documents with MIME types other than multipart.

(Nevertheless, matches against container types with an empty
match string can be useful as tests for the existence of such
document parts.)

MIME parts encoded in "quoted-printable" or "base64" content
transfer encodings MUST be decoded to prior to the match.
MIME parts in other transfer encodings MAY be decoded, omitted
from the test, or processed as raw data.

MIME parts identified as using charsets other than UTF-8 as
defined in [UTF-8] SHOULD be converted to UTF-8 prior to the match.
A conversion from US-ASCII to UTF-8 MUST be supported.
If an implementation does not support conversion of a given
charset to UTF-8, it MAY compare against the US-ASCII subset
of the transfer-decoded character data instead. Characters from
documents tagged with charsets that the local implementation
cannot convert to UTF-8 and text from mistagged documents MAY
be omitted or processed according to local conventions.

Search expressions MUST NOT match across MIME part boundaries.
MIME headers of the containing text MUST NOT be included in the
data.

Example:
```
require ["body", "fileinto"];
```

# Save any message with any text MIME part that contains the
# worlds "missile" or "coordinates" in the "secrets" folder.

if body :content "text" :contains ["missile", "coordinates"] {  
  fileinto "secrets";
}

# Save any message with an audio/mp3 MIME part in
# the "jukebox" folder.

if body :content "audio/mp3" :contains "" {  
  fileinto "jukebox";
}

4.3 Body Transform ":text"

The ":text" body transform matches against the results of an implementation’s best effort at extracting UTF-8 encoded text from a message.

In simple implementations, :text MAY be treated the same as :content "text".

Sophisticated implementations MAY strip mark-up from the text prior to matching, and MAY convert media types other than text to text prior to matching.

(For example, they may be able to convert proprietary text editor formats to text or apply optical character recognition algorithms to image data.)

5. Interaction with Other Sieve Extensions

Any extension that extends the grammar for the COMPARATOR or MATCH-TYPE nonterminals will also affect the implementation of "body".

The [REGEX] extension can place a considerable load on a system when applied to whole bodies of messages, especially when implemented naively or used maliciously.

Regular and wildcard expressions used with "body" are exempt from the side effects described in [VARIABLES]. That is, they do not set numbered variables ${1}, ${2}... to the input values corresponding to wild card sequences in the matched pattern. However, variable references in the pattern string are evaluated as described in the draft, if the extension is present.

6. IANA Considerations

The following template specifies the IANA registration of the Sieve extension specified in this document:

To: iana@iana.org
Subject: Registration of new Sieve extension

Capability name: body
Capability keyword: body
Capability arguments: N/A
Standards Track/IESG-approved experimental RFC number: this RFC
Person and email address to contact for further information:

Jutta Degener
jutta@sendmail.com

This information should be added to the list of sieve extensions given on http://www.iana.org/assignments/sieve-extensions.

7. Security Considerations

The system MUST be sized and restricted in such a manner that even malicious use of body matching does not deny service to other users of the host system.

Filters relying on string matches in the raw body of an e-mail message may be more general than intended. Text matches are no replacement for a virus or spam filtering system.

8. Acknowledgments

This document has been revised in part based on comments and discussions that took place on and off the SIEVE mailing list. Thanks to Cyrus Daboo, Ned Freed, Simon Josefsson, Mark E. Mallet, Chris Markle, Greg Shapiro, Tim Showalter, Nigel Swinson, and Dowson Tong for reviews and suggestions.

9. Author’s Address

Jutta Degener
Sendmail, Inc.
6425 Christie Ave, 4th Floor
Emeryville, CA 94608

Email: jutta@sendmail.com

10. Discussion

This section will be removed when this document leaves the Internet-Draft stage.

This draft is intended as an extension to the Sieve mail filtering language. Sieve extensions are discussed on the MTA Filters mailing list at <ietf-mta-filters@imc.org>. Subscription requests can be sent to <ietf-mta-filters-request@imc.org> (send an email message with the word "subscribe" in the body).

More information on the mailing list along with a WWW archive of back messages is available at <http://www.imc.org/ietf-mta-filters/>.

10.1 Changes from the previous version

Made "body" exempt from variable-setting side effects in the presence of the "variables" extension and wild cards. It’s too hard to implement.
Removed :binary. It’s uglier and less useful than it needs to be to bother.

Added IANA section.

Appendices

Appendix A. References


Appendix B. Copyright Statement

Copyright (C) The Internet Society (2004). 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 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.